Computer Networks 215 (2022) 109200

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Dynamic UPF placement and chaining reconfiguration in 5G networks

Irian Leyva-Pupo [a,*], Cristina Cervelló-Pastor [a,*], Christos Anagnostopoulos [b], Dimitrios P. Pezaros [b]

[a] Department of Network Engineering, Universitat Politècnica de Catalunya (UPC), 08860, Castelldefels, Spain
[b] School of Computing Science, University of Glasgow (UoG), G12 8QQ Glasgow, UK

ARTICLE INFO

Keywords:
5G
Dynamic reconfiguration
Integer linear programming (ILP)
Network function virtualization (NFV)
Optimal stopping theory (OST)
Service function chain (SFC)
User plane function (UPF)

ABSTRACT

Network function virtualization (NFV) and multi-access edge computing (MEC) have become two crucial pillars in developing 5G and beyond networks. NFV promises cost-saving and fast revenue generation through dynamic instantiation and the scaling of virtual network functions (VNFs) according to time-varying service demands. Additionally, MEC provides considerable reductions in network response time and backhaul traffic since network functions and server applications can be deployed close to users. Nevertheless, the placement and chaining of VNFs at the network edge is challenging due to numerous aspects and attendant trade-offs. This paper addresses the problem of dynamic user plane function placement and chaining reconfiguration (UPCR) in a MEC environment to cope with user mobility while guaranteeing cost reductions and acceptable quality of service (QoS). The problem is formalized as a multi-objective integer linear programming model to minimize multiple cost components involved in the UPCR procedure. We propose a heuristic algorithm called dynamic priority and cautious UPCR (DPC-UPCR) to reduce the solution time complexity. Additionally, we devise a scheduler mechanism based on optimal stopping theory to determine the best reconfiguration time according to instantaneous values of latency violations and a pre-established QoS threshold. Our detailed simulation results evidence the efficiency of the proposed approaches. Specifically, the DPC-UPCR provides near-optimal solutions, within 15% of the optimum in the worst case, in significantly shorter times than the mathematical model. Moreover, the proposed scheduling method outperforms two scheduler baseline solutions regarding the number of reconfiguration events and QoS levels.

1. Introduction

The fifth generation (5G) of mobile networks will likely mark a turning point not only in the telecommunications field but in almost all spheres of society (e.g., health care, education, and industry). 5G technology is expected to bring tremendous growth in connectivity, mobile traffic capacity, and new capabilities that enhance performance by providing greater throughput, lower latency, ultra-high reliability, higher connectivity density, and an expanded range of mobility [1]. These features demand unprecedented transformations regarding how services are created and operated, how users communicate, and how networks are designed and managed.

In the path toward 5G and beyond networks, technologies such as network function virtualization (NFV) [2] and multi-access edge computing (MEC) [3] are two fundamental enablers [1,4]. NFV allows coping with user demand variations through flexible and dynamic deployment and scaling of virtual network functions (VNFs). Additionally, MEC brings computing, storage, and networking capabilities close to users, at the network edge.

The implementation of user plane functions (UPFs) [5] and applications at the network edge will reduce the end-to-end (E2E) network response time and backhaul bandwidth consumption, thus avoiding traffic congestion in the core network. Furthermore, the virtualization of UPFs provides more efficient usage of MEC node resources since UPFs can be scaled and deployed on demand. Nonetheless, the VNF placement, especially 5G UPFs, at the network edge poses extra challenges due to numerous aspects and attendant trade-offs. In 5G networks, packet data unit (PDU) sessions can be served by multiple UPFs, chained together, and realizing diverse tasks to enhance efficiency. Numerous UPFs performing different functionalities in a service data path add extra complexity to the problem since we must consider their order and inter-dependency requirements under strict service latency demands and limited resources at edge locations.

Moreover, fulfilling the 5G data plane's stringent latency requirements (less than 1 ms) demands additional levels of user plane distribution, moving from up to 100 sites to a maximum of 1000 sites [6]. The latter will require higher deployment and operational costs as more

* Corresponding authors.
E-mail addresses: irian.leyva@entel.upc.edu (I. Leyva-Pupo), cristina@entel.upc.edu (C. Cervelló-Pastor).

https://doi.org/10.1016/j.comnet.2022.109200
Received 2 December 2021; Received in revised form 16 May 2022; Accepted 15 July 2022
Available online 20 July 2022
1389-1286/© 2022 The Author(s). Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

edge nodes and UPF instances need to be deployed. Additionally, in a MEC ecosystem, frequent session relocations are most likely due to the presence of highly mobile users and UPFs' smaller service areas.

On the other hand, expenditures and session relocations can be reduced by decreasing the number of UPF instances and consolidating their deployment on a small subset of edge nodes. However, this may cause the quality of service (QoS) to deteriorate and routing costs to increase. Moreover, dynamic UPF placement and chaining reconfiguration (UPCR) may be required to cope with user mobility and ensure QoS. Nevertheless, these reconfiguration events may produce temporal service interruptions, extra delays along and additional costs. Thus, this study addresses problems related to determining the best UPCR that allows cost reductions while ensuring 5G service requirements are fulfilled. Moreover, the optimal time to readjust the UPCR needs to be determined to reduce reconfiguration effects. Overall, our work addresses the question of *how* and *when* the UPCR should be performed. In this regard, our main contributions can be summarized as follows:

- We formulate the UPCR problem as an integer linear programming (ILP) model under service function chain (SFC) and UPF specific constraints. The model's primary goal is to minimize capital and operational costs related to reconfiguration events while ensuring service requirements.
- We design a heuristic algorithm based on the SFCR-mapping procedure [7], referred to as the Dynamic Priority and Cautions UPCR (DPC-UPCR), to efficiently remap SFC requests (SFCRs) and readjust UPF placement in online scenarios. Extensive simulation results evidence that the DPC-UPCR algorithm outperforms two greedy-based baselines and provides near-optimal reconfiguration costs in a significantly shorter time than the baselines.
- We provide a decision-making mechanism based on optimal stopping theory (OST), referred to as the Optimal Scheduler Reconfiguration (OSR), to determine the optimal time to adjust the UPF placement and chaining (UPC) configuration. The OSR mechanism decides the reconfiguration time according to instantaneous values of sessions with latency violations, an upper QoS threshold, and expected reconfiguration costs. The conducted experiments reveal that the conceived mechanism improves the QoS while decreasing reconfiguration events by at least 30% compared with other state-of-the-art approaches.

The remainder of this paper is organized as follows. Section 2 reviews related work to the UPF placement and dynamic reconfiguration of SFCs. Section 3 introduces the system model, used notation, and proposed solutions for the UPCR problem. Following this, Section 4 introduces and describes the scheduling mechanism, while Section 5 discusses the simulation results. Finally, Section 6 concludes our work and outlines directions for future research.

2. Related work

This section briefly reviews the literature related to the UPF placement problem and the dynamic reconfiguration of SFCs. Special attention is paid to their solution approaches (i.e., static or dynamic placement and reactive or proactive reconfiguration strategies).

2.1. User plane function placement

Peters et al. [8] introduce the concept of anticipatory user plane management for 5G networks to reduce the user plane configuration latency during handovers. In this vein, decisions regarding intermediate UPF (I-UPF) placement are made based on individual user mobility patterns prediction (e.g., target access point). Additionally, in [9], they provide a blueprint of their proposed approach for the I-UPF placement. Their solution selects the best location by evaluating a cost function that ranks feasible candidates according to the 5G use case requirements.

Authors in [10,11] address the joint placement of edge servers and anchor UPFs (A-UPFs) intending to improve QoS. In [10], two solution approaches are presented (an ILP model and an algorithm) with the main goal of minimizing UPF deployment costs according to user mobility patterns. By contrast, in [11], the main aim is to minimize latency subject to cost limitations. In both works, service demands are assigned per traffic generators (access nodes) and not per session or user basics.

Subramaya et al. [12] apply machine learning (ML) techniques for the proactive auto-scale of user plane SFC placement. To this aim, they propose two ML models to predict the required number of VNF based on traffic traces. Moreover, they also address the joint user association and SFC placement problem in a hierarchical MEC environment. The problem is formulated as an ILP model targeted at minimizing E2E service delay. To overcome scalability limitations, a heuristic is proposed in which SFCs are mapped according to latency requirements. Nevertheless, no placement constraints associated with UPF requirements or VNF-order are considered. In [7], the 5G UPC problem is formalized as an ILP model aimed at optimizing provisioning cost and service response time. The mathematical model encompasses several aspects: UPF-specific requirements, VNF order, and chain topology. Moreover, two heuristic-based approaches are presented to improve its solution efficiency.

A common limitation of the aforementioned works is that they overlook aspects associated with reconfiguration events such as session relocation or UPF migration since static approaches are adopted. In this regard, an ILP model for the placement reconfiguration of A-UPFs is presented in [13] which does consider multiple cost components as well as requirements during the UPF placement readjustment.

2.2. Dynamic SFC placement

The dynamic reconfiguration of VNF placement and chaining (VNFPC) has been widely addressed in the literature through either reactive [14–19] or proactive approaches [20,21].

Liu et al. [14] study the problem of joint optimization of new and in-service SFCs provisioning. Their main objective is to maximize operator revenue by maximizing the acceptance requests' profit and reducing deployment costs. The problem is formulated as ILP and a column generator-based solution is presented to reduce time complexity. However, service latency requirements are not considered and the proposed solution still presents scalability limitations.

In [15], the authors introduce a multi-objective ILP model to minimize SFC reconfiguration costs of value-added services in Content Delivery Networks. Several cost components are considered (i.e., routing, migration, and VNF hosting and instantiation) along with resource capacity, service latency, and mapping constraints. Similar objectives are considered in [16], where a framework for adaptive VNFPC called NFV-PEAR is presented. This framework has an optimization modulo that runs periodically to readjust the VNFPC due to demand fluctuations. Apart from network infrastructure resource optimization, the proposed ILP model also seeks to reduce changes in VNFs and flow mapping as well as SFC reassignments.

Liu et al. [17] address the SFC dynamic reconfiguration problem to balance between the service provider's revenue and reconfiguration cost. They formulate the problem as an ILP model and propose a heuristic solution that combines Tabu search and Fuzzy C-means. The reconfiguration trigger condition is based on the substrate network utilization thresholds (i.e., lower and upper bounds). In [18], the VNF infrastructure migration and SFC reconfiguration problem are formalized as an ILP model and a heuristic is proposed to solve it in polynomial time. Its main objective is to minimize delay along with network imbalance. The migration trigger conditions are physical node failure and network resource overload.

Chen and Liao [19] investigate the impact of user mobility on embedding SFCs in a MEC environment. They formulate the problem as an

ILP to optimize user satisfaction by minimizing the effects of handovers in the SFC migration delay and the service downtime. They consider resource limitations in MEC servers and link bandwidth along with service propagation delay requirement and migration time. They show the problem to be NP-hard and propose a heuristic solution. However, they adopt a user-based reactive approach since the migration decision is taken upon user handover.

Gu et al. [20] propose an online learning solution to predict SFC flows to take VNF scaling decisions. Their main objective is to minimize service provider operational expenditure by scaling and provisioning VNF in a proactive way. They formulate the problem as a multi-objective MILP model and present an algorithm for its online solution. Wang et al. [21] propose a user-managed framework for the online orchestration of SFCs at the edge in which each user is responsible for managing and allocating resources to their service based on learned information. Their objective is to minimize the E2E delay by leveraging contextual information (i.e., user demand and mobility). The problem is formulated as MIP subject to affinity, link capacity, and path-related constraints, and a bandit-based algorithm is provided. However, the effects of users' decisions on the network operating costs are not considered.

2.3. Optimal stopping time

The principles of OST have been widely adopted to solve optimization problems. Cziva et al. in [22] propose an ILP model to minimize users' E2E service latency, s.t. latency and capacity constraints. Additionally, a dynamic placement scheduler is presented to forecast when the placement needs to be readjusted. Their main objective is to guarantee the established QoS levels (i.e., the cumulative sum of sessions with latency violations), whereas frequent placement recalculations are avoided. They apply the principles of OST to dynamically determine the optimal reconfiguration time. Likewise, authors in [13] propose a scheduling method based on OST to decide the UPF placement reconfiguration time according to instantaneous values of latency violations and pre-established QoS thresholds.

In [23], a dynamic service migration strategy is presented to optimize the energy consumption of the MEC platform. Anagnostopoulos and Kolomvatsos [24] propose a model based on OST to determine the right time to take a mitigation action in ENs (e.g., upgrade the current services/resources or offload tasks). The optimal time to take these actions is identified upon sequentially observed values of a specified QoS metric. Yan et al. [25] study the joint optimization of model placement and online model splitting with the goal of minimizing the energy-and-time cost of device-edge co-inference. They formulate the optimal model splitting point selection problem as an OSP and propose a one-stage look-ahead (1-SLA) stopping rule to reduce solution complexity.

Unlike the existing literature, our work investigates the problem of dynamic UPF placement and chaining reconfiguration. To this aim, PDU sessions are modeled as SFCRs, which may have different topologies (single and multiple branches) and numbers and types of constituent UPFs. In this regard, the proposed solutions consider several aspects such as UPFs' roles and specificities (e.g., anti-affinity, interdependency, and order) and multiple cost components associated with the reconfiguration process. In addition, a scheduling mechanism based on OST is introduced to determine the best reconfiguration time according to instantaneous values of sessions with latency violation, QoS threshold, and expected reconfiguration cost.

3. Optimal UPCR

This section presents a brief background on 5G UPFs, followed by the system model and used notation. We then formulate the UPCR problem as a multi-objective ILP model to optimize deployment and operational expenditures associated with reconfiguration events. Lastly, we describe the proposed heuristic referred to as DPC-UPC and analyze its complexity.

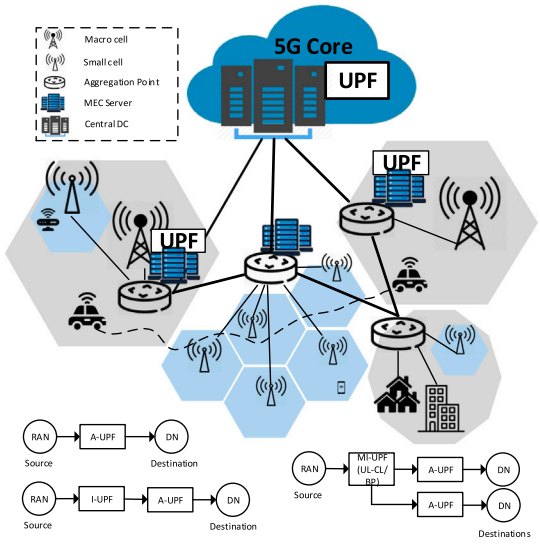


Fig. 1. Example of UPF placement and SFC topologies in a MEC ecosystem.

3.1. 5G user plane functions

The 5G user plane is formed by the UPFs [5], which combine functionalities of traditional serving and packet gateways (SGW and PGW). The UPF's primary function is to process data plane packets between the access and data networks (AN and DN). Nevertheless, it may perform other functionalities such as packet routing and forwarding, traffic steering, QoS handling, packet inspection, and lawful intersection. These functionalities do not need to be supported by a unique UPF instance, and several UPFs with different roles can be instantiated as needed. This approach provides faster response time along with higher flexibility and scalability of the user plane. In 5G standards, unlike traditional networks, a PDU session may be served by multiple UPFs chained together.

Though UPFs can play many roles, the most common are the PDU session anchor (PSA or A-UPF), intermediate-UPF, uplink classifier (UL-CL), and branching point (BP) are the most common. The A-UPFs are responsible for IP anchoring as well as terminating PDU sessions at the DN end through the N6 interface. I-UPFs forward traffic between the AN and the A-UPF through the interfaces N3 and N9, respectively. They are used for redundant transmission based on two N3 and N9 tunnels (required for URLLC services) or to ensure service continuity. Additionally, UL-CL and BPs are inserted between the AN and PSAs when more than one PSA (i.e., multihoming) is required. These functionalities allow selective traffic routing, which is highly useful for MEC environments where the traffic may be routed or steered to applications in the local DN [26]. Moreover, these UPFs are used to guarantee session and service continuity during user handovers as well as load balancing.

As UPFs are unaware of PDU session necessities, they need to be told what to do. The entity responsible for this is the session management function (SMF), located in the control plane. Basically, SMFs are responsible for selecting, instructing, and managing the UPFs associated with PDU sessions during their lifetimes. Moreover, PDU sessions can have different characteristics and demands according to their topology and service demands. This paper considers three basic topologies for UPF SFCs. However, these topologies can be extended by including other VNF types and UPF roles to create more complex structures. Fig. 1 illustrates these topologies as well as the deployment of UPFs in a MEC ecosystem.

Table 1
Physical and virtual networks notation.

Notation	Description
Sets	
N	Set of all nodes
N_r	Set of access nodes
N_c	Set of candidate locations (e.g., MEC servers)
N_a	Set of aggregation points
E	Set of physical links
P	Set of paths between all network nodes
$P_{n,m}$	Set of paths between nodes n and m
T	Set of all types of available VNFs
F	Set of already deployed VNF instances
Parameter	
C_c	Resource capacity at candidate node $n \in N_c$
C_t	Resource capacity of VNF of type $t \in T$
$\beta_{u,v}$	Bandwidth capacity of link (u,v)
$d_{u,v}$	Latency associated with link (u,v)
d_p	Latency associated with path $p \in P$
d_t	Processing delay of VNF of type $t \in T$
I_t	Maximum number of instances of type $t \in T$
Ψ	Cost component

3.2. System model and notation

The 5G network model is described as a directed graph G(N, E), where N and E denote the sets of physical nodes and edges, respectively. The set of nodes is formed by three subsets: access nodes (N_r), aggregation points (N_a) and server nodes (N_c). Server nodes are analyzed as VNF candidate locations since they provide virtualization resources required to host VNF instances. The amount of resources (e.g., memory and CPU) available in a candidate node is represented by C_c. A physical link $(u, v) \in E$ is characterized by its bandwidth capacity ($\beta_{u,v}$) and latency ($d_{u,v}$). The latter includes the propagation delay and the processing time of transmission nodes. Moreover, a set of pre-calculated paths (P) is also included. Each path $p \in P$ is identified by its two endpoints (n, m) and an id (h) which helps to differentiate different paths between the same pair of nodes. The indicator $H^p_{u,v}$ specifies if path p is formed by link (u, v) or not.

The set of available VNF types is denoted by T, where $t \in T$ represents a specific type (e.g., t = 1: A-UPF, t = 2: MI-UPF, and t = 3: I-UPF). For each VNF type, a processing capacity (C_t), a processing time (d_t), and a maximum number of instances (I_t) are stipulated. Additionally, the set F represents those VNF instances that are running at the reconfiguration time, where (i, t) identifies the *i-th* instance of type t in F. These VNFs were deployed as a result of an initial placement and mapping configuration or due to a previous reconfiguration event. Table 1 provides the notation related to physical and virtual networks.

Let us consider the following scenario: a set of mobile devices are connected to the access network through their nearest access node ($n^s_r \in N^s_r$) and requesting PDU sessions (S). Each PDU session $s \in S$ is compound by an ordered set of VNF services (F_s) and requires a certain processing capacity (C_s), bandwidth (B_s), and E2E delay (L_s) to be served. The properties of a given SFCR $s \in S$ are represented by a tuple $(n^s_r, F_s, C_s, \beta_s, L_s, B_s, T^{f,t}_s, O^{f,g,b}_s, Q^{f,b}_s)$, where parameters B_s, $T^{f,t}_s$, $O^{f,g,b}_s$ and $Q^{f,b}_s$ denote the number of branches in an SFCR, the type of each VNF, as well as their order and presence in each branch, respectively. Unlike other related studies, our model does not include the destination nodes of an SFCR since we assume that they are DNs co-located with A-UPFs. The used notation for SFCRs and decision variables is summarized in Table 2 and Table 3, respectively.

3.3. Optimal UPCR

In this subsection, we formulate the UPCR problem as an ILP model aimed at minimizing deployment and operational costs associated with the UPF placement and chaining configuration during reconfiguration events. To perform this, we consider multiple cost components:

Table 2
Service Function Chain Request notation.

Notation	Description
S	Set of PDU sessions (SFCRs)
N^s_r	Set of access nodes (N_r) per PDU sessions
n^s_r	Access node ($n_r \in N_r$) of SFCR $s \in S$
F_s	Set of VNFs forming SFCR $s \in S$
$\|F_s\|$	Number of VNFs forming SFCR $s \in S$
C_s	Computing resources required by SFCR $s \in S$
β_s	Bandwidth capacity required by SFCR $s \in S$
L_s	E2E latency requirement of SFCR $s \in S$
B_s	Number of A-UPFs (branches) in SFCR $s \in S$
$T^{f,t}_s$	1 if VNF $f \in F_s$ forming SFCR $s \in S$ is of type $t \in T$
$O^{f,g,b}_s$	1 if VNF f goes just before VNF g in branch $b \in B_s$ of SFCR $s \in S$
$Q^{f,b}_s$	1 if VNF $f \in F_s$ is present in branch $b \in B_s$ of SFCR $s \in S$

Table 3
Decision variables and indicators.

Notation	Description
Binary variables	
w_n	1 if candidate node $n \in N_c$ is open.
r_s	1 if PDU session $s \in S$ was reassigned during the reconfiguration.
$v_{i,t}$	1 if as a result of the reconfiguration there is a new instance $i \in I_t$ of VNF type $t \in T$
$x_{i,t,n}$	1 if instance $i \in I_t$ of VNF type $t \in T$ is deployed on node $n \in N_c$
$m_{i,t,n',n}$	1 if instance $i \in I_t$ of VNF type $t \in T$ was migrated from node n' to node n ($n', n \in N_c$)
$z^{f,s}_{i,t,n}$	1 if VNF $f \in F_s$ of SFCR $s \in S$ is mapped to instance $i \in I_t$ of VNF type $t \in T$ located at node $n \in N_c$
$a^{f,s}_n$	1 if VNF $f \in F_s$ of SFCR $s \in S$ is assigned to node $n \in N_c$
$y^{f,g,s}_p$	1 if path $p \in P$ is used to route traffic between VNFs f and g $(f, g \in F^+_s)$ of SFCR $s \in S$
$\delta^{f,g,s}_{n,m}$	1 if VNFs f and g $(f, g \in F_s)$ of SFCR $s \in S$ are mapped on nodes n and m, resp.
ξ_t	1 if there is at least one new VNF instance of type $t \in T$ deployed in the network
Binary indicators	
$\tilde{X}_{i,t,n}$	1 if instance $i \in I_t$ of VNF type $t \in T$ was placed on node $n \in N_c$ before the reconfiguration
$\tilde{A}^{f,s}_n$	1 if VNF $f \in F_s$ of SFCR $s \in S$ is hosted in node $n \in N_c$
$H^p_{u,v}$	1 if path $p \in P$ is mapped to link $(u, v) \in E$
V^t_n	1 if node $n \in N$ supports VNFs of type $t \in T$

- Node activation cost (C_{act}): It represents the cost associated with the activation of servers (e.g., energy consumption). This cost is produced when empty candidate nodes (candidates with no VNF deployed) host at least one VNF instance after the reconfiguration.

$$C_{act} = \sum_{n \in N_c} \Psi^n_a \cdot w_n \tag{1}$$

- VNF deployment cost (C_{dep}): It is the cost related to instantiating new VNFs (e.g., software license cost).

$$C_{dep} = \sum_{i \in I_t} \sum_{t \in T} \Psi^t_d \cdot v_{i,t} \tag{2}$$

- VNFs running cost (C_{run}): It deals with the cost (e.g., power consumption) of running VNF instances. Thus, it is expressed in terms of the overall number of deployed VNFs.

$$C_{run} = \sum_{i \in I_t} \sum_{t \in T} \sum_{n \in N_c} \Psi^{t,n}_r \cdot x_{i,t,n} \tag{3}$$

- VNF migration cost (C_{mig}): It is the cost for migrating an already deployed VNF instance ($f \in F$) from one location to another. Thus, it is expressed in terms of the number of deployed VNFs at the reconfiguration moment.

$$C_{mig} = \sum_{(i,t)\in F} \sum_{n'\in N_c} \sum_{n\in N_c} \Psi_m^{t,n',n} \cdot m_{n',n}^{i,t} \tag{4}$$

- Routing cost (C_{rou}): It expresses the cost of delivering traffic among VNF services forming the SFCRs. Note that this expression can also improve network response time since it includes the propagation delay of the virtual links that form the SFC data paths.

$$C_{rou} = \sum_{f,g\in F_s^+} \sum_{s\in S} \sum_{p\in P} \Psi^{s,p} \cdot d_p \cdot y_p^{f,g,s} \tag{5}$$

- Session reassignment cost (C_{rea}): It is the cost for reassigning PDU sessions during the UPC recalculation. This cost is measured as a penalty that the service provider must pay for interrupting user sessions or exceeding their service delay requirement. We consider a session has been reassigned if at least one of its constituent VNF services has been re-located from its previously assigned server.

$$C_{rea} = \sum_{s\in S} \Psi^s \cdot r_s \tag{6}$$

Given a UPF placement and chaining setup, determined upon either an initial placement or a previous reevaluation event, the goal of the UPCR problem is to determine the optimal UPF placement and chaining rearrangement that minimizes the total costs incurred during reconfiguration events. We express the objective function as a linear combination of the aforementioned cost components to achieve this aim. Additionally, we introduce weight factors (α_i) to reflect the relative importance of each component in the objective function. Moreover, to avoid the influence of one term over the others, we need to normalize their magnitudes. Thus, the UPCR problem can be formulated as follows:

$$\text{Min } C_{tot} = \alpha_1 \cdot C_{act} + \alpha_2 \cdot C_{dep} + \alpha_3 \cdot C_{run} + \alpha_4 \cdot C_{mig} + \alpha_5 \cdot C_{rou} + \alpha_6 \cdot C_{rea} \tag{7}$$

A set of constraints must be satisfied to generate valid solutions to the problem. We group these constraints into the following seven categories:

Infrastructure resources constraints: Constraints (8), (9), and (10) refer to resource limitations in the MEC servers, deployed VNF instances and physical links, respectively. More specifically, constraint (8) ensures that the instances deployed in the candidate nodes do not exceed their available resources (e.g., CPU and memory). Similarly, inequalities (9) and (10) enforce the mapping of SFCRs to those VNF instances and physical links with sufficient processing capacity and bandwidth to satisfy their demands.

$$\sum_{i\in I_t} \sum_{t\in T} C_t \cdot x_{i,t,n} \le C_n \qquad \forall n \in N_c \tag{8}$$

$$\sum_{f\in F_s} \sum_{s\in S} C_s \cdot z_{i,t,n}^{f,s} \le C_t \qquad \forall i \in I_t, \forall t \in T, \forall n \in N_c \tag{9}$$

$$\sum_{f,g\in F_s^+} \sum_{s\in S} \sum_{p\in P} \beta_s \cdot y_p^{f,g,s} \cdot H_{u,v}^p \le \beta_{u,v} \qquad \forall (u,v) \in E \tag{10}$$

VNF placement constraints: Inequality (11) forces server nodes that are not hosting VNF instances to be closed, whereas constraint (12) ensures the deployment of VNF instances in the candidate nodes that are open and support the requested VNF type. Expression (13) guarantees that each VNF is hosted by only one server. Additionally, constraint (14) restricts the maximum number of instances of a given type that can be deployed.

$$w_n \le \sum_{i\in I_t} \sum_{\forall t\in T} x_{i,t,n} \qquad \forall n \in N_c \tag{11}$$

$$x_{i,t,n} \le w_n \cdot V_n^t \qquad \forall i \in I_t, \forall t \in T, \forall n \in N_c \tag{12}$$

$$\sum_{n\in N_c} x_{i,t,n} \le 1 \qquad \forall i \in I_t, \forall t \in T \tag{13}$$

$$\sum_{i\in I_t} \sum_{n\in N_c} x_{i,t,n} \le I_t \qquad \forall t \in T \tag{14}$$

VNF assignment constraints: Inequality (15) expresses that a VNF service requested by an SFC can only be assigned to a VNF if this instance has already been deployed and is of the same type. Constraint (16) avoids the deployment of empty VNFs by ensuring that all VNF instances have been assigned at least one SFCR. Moreover, expression (17) ensures that only one VNF instance is selected to serve a VNF service ($f \in F_s$) request. Expressions (18) and (19) establish the relationship between the variables $a_n^{f,s}$ and $z_{i,t,n}^{f,s}$. Specifically, these expressions state that if a VNF service request is mapped to a node, this is because it has been assigned to a VNF instance deployed on that node.

$$z_{i,t,n}^{f,s} \le x_{i,t,n} \cdot T_s^{f,t} \qquad \forall f \in F_s, \forall s \in S, \forall i \in I_t, \forall t \in T, \forall n \in N_c \tag{15}$$

$$x_{i,t,n} \le \sum_{s\in S} \sum_{f\in F_s} z_{i,t,n}^{f,s} \qquad \forall i \in I_t, \forall t \in T, \forall n \in N_c \tag{16}$$

$$\sum_{i\in I_t} \sum_{t\in T} \sum_{n\in N} z_{i,t,n}^{f,s} = 1 \qquad \forall f \in F_s, \forall s \in S \tag{17}$$

$$a_n^{f,s} \le \sum_{t\in T} \sum_{i\in I_t} z_{i,t,n}^{f,s} \qquad \forall f \in F_s, \forall s \in S, \forall n \in N_c \tag{18}$$

$$a_n^{f,s} \ge z_{i,t,n}^{f,s} \qquad \forall f \in F_s, \forall s \in S, \forall i \in I_t, \forall t \in T, \forall n \in N_c \tag{19}$$

Path mapping constraints: Constraint (20) ensures that there is a path in the specified direction (i.e., $f \to g$) between two consecutive VNFs in a branch of an SFCR. In addition, inequality (21) restricts the maximum number of selected paths between any pair of VNFs to one to avoid loops. It should be noted that for these two constraints, the set of VNFs forming an SFC has been extended ($F_s^+ = F_s \cup n_r^s$) to include the (R)AN, so as the path between the (R)AN and the first VNF in the chain is also mapped. Furthermore, expression (22) guarantees the mapping of consecutive VNF pairs to the endpoints of the selected path. Finally, inequality (23) is an adaptation of (22) to include paths with the (R)AN in one of their endpoints.

$$\sum_{p\in P} y_p^{f,g,s} \ge O_s^{f,g,b} \qquad \forall f,g \in F_s^+, \forall b \in B_s, \forall s \in S \tag{20}$$

$$\sum_{p\in P} y_p^{f,g,s} \le 1 \qquad \forall f,g \in F_s^+, \forall s \in S \tag{21}$$

$$\sum_{p\in P_{n,m}} y_p^{f,g,s} \le a_n^{f,s} \cdot a_m^{s,g} \qquad \forall f,g \in F_s, \forall s \in S, \forall n,m \in N_c \tag{22}$$

$$\sum_{p\in P_{n_r^s,m}} y_p^{n_r^s,g,s} \le a_m^{s,g} \qquad \forall g \in F_s, \forall s \in S, n_r^s = N_r^s[s], \forall m \in N_c \tag{23}$$

Constraint (22) is non-linear since it implies the product of two variables. However, it can be replaced by the following set of linear expressions:

$$\delta_{n,m}^{f,g,s} \le a_n^{f,s} \qquad \forall f,g \in F_s, \forall s \in S, \forall n,m \in N_c$$

$$\delta_{n,m}^{f,g,s} \le a_m^{s,g} \qquad \forall f,g \in F_s, \forall s \in S, \forall n,m \in N_c$$

$$\delta_{n,m}^{f,g,s} \ge a_n^{f,s} + a_m^{s,g} - 1 \qquad \forall f,g \in F_s, \forall s \in S, \forall n,m \in N_c$$

$$\sum_{p\in P_{n,m}} y_p^{f,g,s} \le \delta_{n,m}^{f,g,s} \qquad \forall f,g \in F_s, \forall s \in S, \forall n,m \in N_c$$

UPF constraints: Constraints (24) and (25) are UPF specific. Inequality (24) defines the anti-affinity property for VNFs (e.g., UPFs) of the same type. More specifically, it enforces the deployment of VNF instances of the same type, serving the same PDU session, on different servers. Similarly, expression (25) guarantees that PSA and I-UPFs without multi-homing functions that serve the same PDU session are also mapped to different locations.

$$\sum_{f\in F_s} \sum_{i\in I_t} z_{i,t,n}^{f,s} \le 1 \qquad \forall s \in S, \forall t \in T, \forall n \in N_c \tag{24}$$

$$\sum_{f\in F_s}\sum_{i\in I_t} z^{f,s}_{1,i,n} + \sum_{f\in F_s}\sum_{i\in I_t} z^{f,s}_{3,i,n} \le 1 \qquad \forall s\in S, \forall n\in N_c \quad (25)$$

Reconfiguration constraints: Expressions (26)–(29) are related to reconfiguration and help determine the type of changes produced during this event. Constraint (26) indicates that a VNF instance has been migrated when its location before and after the reconfiguration is different. Note that this constraint is linear, as $X_{i,t,n}$ is a parameter and not a variable. We added expressions (27) and (28) to promote VNF migration over new deployments when the deployment component is not considered in (7). Otherwise, new instances can be deployed when optimizing migration effects. Mainly, they express the instantiation of new VNFs of a given type, as the activation of new VNFs' IDs and the incremental in the number of its deployed instances w.r.t. the previous placement configuration.

$$m^{i,t}_{n',n} = x_{i,t,n}\cdot \bar{X}_{i,t,n'} \qquad \forall (i,t)\in F, \forall n,n'\in N_c; n\neq n' \quad (26)$$

$$v_{i,t} = \Big[\sum_{n\in N_c} x_{i,t,n} - \sum_{n\in N_c} \bar{X}_{i,t,n}\Big]^+ \qquad \forall i\in I_t, \forall t\in T \quad (27)$$

$$\sum_{i\in I_t} v_{i,t} = \Big[\sum_{n\in N_c}\sum_{i\in I_t} x_{i,t,n} - \sum_{n\in N_c}\sum_{i\in I_t} \bar{X}_{i,t,n}\Big]^+ \qquad \forall t\in T \quad (28)$$

$$r_s = 1 \Leftrightarrow |F_s| - \sum_{f\in F_s}\sum_{n\in N_c} a^{f,s}_n \cdot \bar{A}^{f,s}_n \ge 1 \qquad \forall s\in S \quad (29)$$

Constraint (29) guarantees that if a session is reassigned, this is because at least one of the VNFs forming the SFC has been re-located. We assume that the reassignment of VNF service requests (e.g., $f\in F_s$) inside the same server does not affect the session/service continuity. However, for more restrictive considerations it could be modified by expressing r_s in terms of the variable $z^{f,s}_{i,t,n}$ instead of $a^{f,s}_n$. Please note that expressions (27)–(29) are not linear. Thus, they can be reformulated in a linear form as follows:

$$(27)\Leftrightarrow \begin{cases} v_{i,t} \ge \sum_{n\in N_c} x_{i,t,n} - \sum_{n\in N_c} \bar{X}_{i,t,n} & \forall i\in I_t, \forall t\in T \\ 2\cdot v_{i,t} \le 1 + \sum_{n\in N_c} x_{i,t,n} - \sum_{n\in N_c} \bar{X}_{i,t,n} & \forall i\in I_t, \forall t\in T \end{cases}$$

$$(28)\Leftrightarrow \begin{cases} \sum_{i\in I_t} v_{i,t} = \xi_t \cdot (\sum_{n\in N_c}\sum_{i\in I_t} x_{i,t,n} - \sum_{n\in N_c}\sum_{i\in I_t} \bar{X}_{i,t,n}) & \forall t\in T \\ \xi_t \ge (\sum_{n\in N_c}\sum_{i\in I_t} x_{i,t,n} - \sum_{n\in N_c}\sum_{i\in I_t} \bar{X}_{i,t,n})/I_t & \forall t\in T \\ \xi_t \le (I_t + \sum_{n\in N_c}\sum_{i\in I_t} x_{i,t,n} - \sum_{n\in N_c}\sum_{i\in I_t} \bar{X}_{i,t,n})/(I_t+1) & \forall t\in T \end{cases}$$

$$(29)\Leftrightarrow \begin{cases} \sum_{f\in F_s}\sum_{n\in N_c} a^{f,s}_n \cdot \bar{A}^{f,s}_n \ge |F_s|(1-r_s) & \forall s\in S \\ \sum_{f\in F_s}\sum_{n\in N_c} a^{f,s}_n \cdot \bar{A}^{f,s}_n \le |F_s\cdot| - r_s & \forall s\in S \end{cases}$$

QoS constraints: The QoS is measured in terms of service latency which is expressed as the combination of the VNF processing time and the propagation delays between consecutive VNFs in a branch. In this regard, constraint (30) guarantees that the E2E delay, in the round-trip-time (RTT), does not violate the service latency requirement.

$$2\cdot(\sum_{f\in F^+_s}\sum_{t\in T}\sum_{n\in N_c} d_t \cdot Q^{f,b}_s \cdot T^{f,t}_s + \sum_{f,g\in F^+_s}\sum_{p\in P} d_p \cdot O^{f,g,b}_s \cdot y^{f,g,s}_p) + d_{DN} \le L_s$$
$$\forall b\in B_s, \forall s\in S \quad (30)$$

The VNF placement and chaining problem, as well as its dynamic reconfiguration, have been proven to be NP-hard problems [14,17,18,27,28]. The complexity of these problems derives from the requirements and constraints (e.g., node capacity, latency, and anti-affinity) that need to be satisfied during their solution. Evaluating these restrictions becomes intractable for large networks where the number of possible combinations increases with the network size (number of nodes and links) and users (SFCRs). Given that UPFs are a specific VNF type with similar requirements to these two problems as well as additional ones due to UPF's different roles and SFC topologies (linear and non-linear), we can deduce that the UPCR problem is also NP-hard, and no polynomial-time complexity algorithm exists to solve it unless $P = NP$. To overcome this limitation, the design of heuristic solutions to solve the UPC problem for large-scale scenarios in polynomial time is mandatory.

3.4. Dynamic priority and cautious UPCR

Scalability limitations associated with ILP models cause the latter to become computationally intractable as the solution space increases (e.g., the number of MEC servers, SFCRs and VNFs). For this reason, we design a heuristic algorithm capable of solving the UPCR problem in polynomial time. The pseudo-code of the proposed solution (i.e., DPC-UPCR) is shown in Algorithm 1.

The DPC-UPCR algorithm comprises four main stages. In the first phase, the algorithm starts by gathering data related to the current placement configuration and SFC mapping (line: 2). For example, the algorithm collects information about the location of each VNF instance, its available capacity, and infrastructure resource utilization. This data is later used to determine the SFCR mapping and placement locations that imply fewer transformations and have the lowest impact on the overall reconfiguration cost. Following this, the algorithm creates the set of sessions that are remapped during the placement reconfiguration (line: 4). This set can encompass either all the active sessions in the system or a subset of them, as indicated by the parameter P_r. In the latter case, the sessions with the worst QoS are prioritized (i.e., sessions with latency violations or sessions which have almost exceeded their service latency requirement). Afterward, the algorithm proceeds to release resources assigned to the selected SFCRs (line: 5). As part of this step, sessions assigned to underutilized VNF instances can also be unmapped, which implies an update of the set of sessions to be remapped (line: 6). As a result of this step, VNF instances can be removed, and active servers can be closed.

The second phase of the algorithm aims to determine the set of available candidates for each SFCR. This phase starts by determining the set of available candidates to be used during the SFC remapping procedure (line: 9). A server is considered available when it disposes of enough resources to instantiate a new VNF or its deployed VNFs can serve at least one SFCR. Once this step has been executed, the algorithm proceeds to find the possible candidates for each SFC according to the service latency requirements (lines: 10–12). This information helps to determine whether an SFCR is in a critical stage. An SFCR is considered critical if its number of possible candidates is lower or equal to the minimum number required to map its constituent VNFs.

The third part of the algorithm involves the SFC mapping procedure. First, the selected SFCRs (S_{remap}) are sorted according to the established criteria (line: 13). More specifically, they are sorted according to their criticism, latency requirement, SFC length, access node location, and the number of available candidates. This combination of parameters increases the possibilities of successfully mapping the most demanding and critical sessions. After having established the initial mapping order of the selected sessions, the algorithm starts looping over them by always choosing the most demanding one (line: 15). The algorithm uses the SFCR-remapping procedure (line: 16) to map a session.

The SFCR-remapping method is based on the SFCR-mapping procedure proposed in [7], with some minor modifications. The main difference between both procedures is the function used to evaluate the mapping cost of a VNF instance. More specifically, the SFCR-remapping procedure computes the mapping cost according to (7). Additionally, this procedure does not consider the popularity of candidates.

For each unmapped VNF instance that forms the selected SFC, the algorithm determines all its available candidates. This step is required given that the latency budget and source nodes change with the mapping of previous VNFs in the chain. These candidates are then classified as feasible or infeasible locations according to whether they satisfy or not the UPCR constraints (e.g., latency and anti-affinity). Additionally, the algorithm obtains the shortest virtual path with enough bandwidth

Algorithm 1: Dynamic Priority & Cautious UPCR

Input: flag_improve, No. of improvement attempts (F_i), Percentage of additional SFCs to remap (P_r)

```
// Phase 1: Prepare variables and indicators
Initialize output variables and parameters
Gather information about previous placement and mapping configuration
if P_r! = 100 then
    S_remap ← Select sessions (s ∈ S) to be remapped based on QoS and P_r
    Release resources assigned to selected sessions
    Update S_remap if required
else
    S_remap ← S
// Phase 2: Find possible candidates and classify SFCRs
Select candidates with available capacity
forall s ∈ S_remap do
    Determine possible candidates
    Classify SFCR s as critical or not according to its number of available candidates
// Phase 3: Map SFCRs
Sort s ∈ S_remap according to established criteria (sort_c)
while S_remap ≠ ∅ do
    s ← S_remap[0]
    SFCR-remapping procedure(s)
    S_remap ← S_remap − s
    if mapping_success then
        S_map ← S_map ∪ s
        Update network and infrastructure resources
        if available servers changed then
            Update available candidates and determine criticism level for each SFCR
            Sort selected SFCRs according to sort_c
    else
        S_unmap ← S_unmap ∪ s
// Phase 4: Improve solution and apply configuration
if S_unmap = ∅ then
    Determine cause of VNF with temporal IDs if any
    if flag_improve then
        cost_best ← Compute solution's cost
        while F_i ≠ 0 do
            Make a copy of current best solution
            f_remap ← Randomly select a VNF different from the previous one
            S_remap ← Sessions assigned to f_remap
            Release resources assigned to selected sessions
            sort_c ← Randomly select a sorting criteria
            Repeat steps 9 - 25
            if S_unmap = ∅ then
                Determine cause of temporal VNF IDs
                cost ← Compute solution's cost
                if cost < cost_best then
                    cost_best ← cost
                    Update best solution
            F_i ← F_i − 1
    if vnf_tmp_IDs then
        Update VNFs IDs
    Apply mapping and placement configuration
```

to support the requested traffic flow for each feasible candidate. Afterward, the algorithm evaluates the impact of the remapping cost on the set of feasible candidates. Since mapping a VNF instance to a different host may incur additional costs in terms of VNF migrations and SFC relocations, this procedure attempts to maintain a similar configuration to the previous placement for as long as possible.

When the selection of a node implies the relocation of a VNF service request, the mapping procedure first checks if this node has available instances of the required type from the previous placement. If this is the case, one of these instances may be reused, and only the operation cost is incurred related to UPF costs (operation, migration, and new deployment). Otherwise, the migration or the new deployment cost component is also present. Like the ILP model, the algorithm considers a new deployment when the current number of instances of a given type is greater than before the reconfiguration event. Otherwise, the algorithm assumes that a VNF instance has been migrated. Since VNF instances are shared by several SFCs, and this decision is based on the mapping of the current PDU session, further analysis is required. To this end, the algorithm assigns temporal IDs to these instances, which helps determine the real cause of a VNF deployment (i.e., new instantiation or migration).

At the end of the evaluation process, the algorithm maps a VNF instance to the candidate location with the lowest cost. This process continues until all the VNFs in the chain have been evaluated or no feasible candidate is left. The latter implies the interruption of the mapping procedure and its classification as failed (lines: 24–25). Once the SFCR-remapping procedure has finished, the algorithm removes the selected session from the remapped SFCR set (S_{remap}), and the session is classified as mapped or rejected (line: 17). The successful mapping of an SFC requires an update of the network and the infrastructure's available resources (lines: 18–23). Moreover, if the number of available candidates changes, the algorithm updates the subset of available candidates and the critical level of each SFCR. A change in the number of available candidates requires that the remaining SFCRs be reordered (line: 23).

The execution of the last phase of the algorithm depends on the successful mapping of all the selected SFCRs (lines: 26–46). The main objectives of this phase are to improve the quality of the obtained solution and apply the obtained mapping and placement configuration. This phase begins by determining the best combination of migration and new deployments for instances with temporal IDs if there are any (line: 27). This action is required because the algorithm assumes the cause of a change (i.e., migration or new deployment) during the SFCR mapping phase based on the mapping of a given session.

The improvement procedure is optional, and it is executed by an input indicator referred to as *flag_improve* (lines: 28–43). This phase's main objective is to enhance the quality of the generated solution by introducing some modifications. It starts by determining the cost of the current placement configuration and setting this cost as the current best solution. Following this, it performs F_i improvement attempts. Each attempt starts by randomly selecting a VNF instance from the set of VNFs ($f_{remap} \in F$) and unmapping its assigned SFCRs. The only condition for selecting this VNF is that it is different from the one previously analyzed. This is recommended to reduce the chances of obtaining consecutive improvement attempts with similar outcomes.

For each improvement attempt, the sorting criteria for the SFCRs' mapping are randomly chosen to promote diversification in the generated solutions (lines: 35). In this regard, our algorithm offers three sorting strategies. One is the previously mentioned strategy in phase 3, which is used to generate the initial reconfiguration solution. Another is a variant of the aforementioned sorting criteria in which the order of the number of candidates and access node location parameters have been exchanged. The final strategy only considers the service latency requirement. It should also be noted that different approaches can be defined as desired by the service provider. Following the sorting, the algorithm generates a new solution by remapping the sessions assigned to the selected VNF instance (S_{remap}). Similar to the initial solution, the cause of temporal IDs in the feasible generated solutions (i.e., $S_{unmap} = \emptyset$) is investigated before determining the solution cost. The algorithm compares the obtained cost with the best cost found so far, and an update process occurs if a better solution is found. At the end of this phase, the algorithm removes any temporal VNF IDs and applies the mapping and placement configuration.

3.5. Time complexity

The time complexity of the DPC-UPCR heuristic is mainly determined by the third and last phases of Algorithm 1, as these phases involve more iterative processes than the first two. Specifically, part three requires performing S iterations to remap the selected SFCRs when considering the worst-case scenario (i.e., $S_{remap} = S$). The runtime of the mapping process is dominated by the SFCR-remapping procedure, which is an adaptation of the SFC-mapping approach proposed in [7]. In the worst case, the time complexity of this phase is at the level of $\mathcal{O}(|S| \cdot |N_c|^2 \cdot |M| \cdot |F_s| \cdot |B_s|)$, where $|S|$, $|N_c|$, $|F_s|$, and $|B_s|$ indicates the sizes of the sets of SFCRs, available candidates, VNFs and branches forming a service chain, respectively. In addition, M represents the runtime complexity associated with the candidate evaluation process. It should be noted that M cannot be determined beforehand, as it depends on several factors such as servers and VNF capacity, UPF specific constraints, and available paths. Hence, the complexity of this phase is formulated based on M.

The complexity of the last stage of Algorithm 1 in terms of runtime depends on the improvement procedure. Each improvement attempt implies the analysis of the sessions assigned to the randomly selected VNF instance (S_{remap}). For this set of sessions, its mapped resources must be released and phases 2 and 3 must be executed. Given that this phase can perform F_i number of improvement attempts, in the worst scenario, its complexity can be expressed as $O(|F_i| \cdot |S| \cdot |N_c|^2 \cdot |M| \cdot |F_s| \cdot |B_s|)$. It should be noted that the number of sessions assigned to a VNF instance is typically much lower than the size of the overall set of SFCRs (i.e., $|S_{remap}| \ll |S|$). In contrast, when no improvement attempt is considered (i.e., $F_i = 0$), this phase's complexity is as simple as $O(|F_{temp}|)$, where $|F_{temp}|$ indicates the size of the VNF set with temporal IDs. In the latter case, the complexity of this algorithm stage is negligible compared to phase three.

The maximum runtime of DPC-UPCR depends on the number of iterations over phase three (F_i+1). In this regard, the overall complexity of the proposed heuristic can be formulated as $O(|F_i| \cdot |S| \cdot |N_c|^2 \cdot |M| \cdot |F_s| \cdot |B_s|)$. However, the number of improvement attempts can be omitted in large-scale scenarios where the numbers of edge nodes (N_c) and PDU sessions are higher (S). In these scenarios, F_i does not contribute as much to the problem size as these other parameters (i.e., $F_i \ll N_c$ and $F_i \ll S$). Similarly, the numbers of VNFs and branches ($|F_s|$ and $|B_s|$) forming the SFC topologies are, generally, significantly smaller parameters than S and N_c. Thus, the overall complexity of Algorithm 1 can be reduced to $O(|S| \cdot |N_c|^2 \cdot |M|)$. From this expression, we note that the time complexity of the DPC-UPCR is strongly dependent on the size of the considered set of SFCRs (S) and candidate locations (N_c). For this reason, the envisioned DPC-UPCR heuristic is a polynomial-time algorithm.

4. Dynamic UPCR

This section introduces a mechanism to schedule the dynamic readjustment of the UPF placement and chaining configuration presented in Section 3. The proposed mechanism is called the Optimal Scheduler Reconfiguration (OSR), and it is based on the principles of OST.

4.1. Problem formulation

Given a UPF placement and chaining configuration obtained as a result of an initial deployment or a reconfiguration event, in which all PDU sessions were mapped according to their service requirements (e.g., latency), we need to consider some degradation in the QoS over time. For instance, the QoS may deteriorate in dynamic environments where users' locations and points of attachment to the network change over time. One cause might be the distance increase with their assigned VNFs (UPFs) due to the absence of nearer VNFs with enough available capacity to attend to the users at their new points of attachment, which implies higher propagation delays. More specifically, a service latency violation in the user plane occurs when the network response time exceeds the service latency requirement (i.e., $L_{ser} \geq L_s$). We define $I_t^s \in [0;1]$ as a random variable that indicates whether the service latency of a PDU session s is affected or not at time t.

$$L_t^s = \begin{cases} 1 & \text{if } L_{ser} \geq L_s \\ 0 & \text{otherwise} \end{cases} \tag{31}$$

Thereby, the total number of sessions with latency violations at a given time t (L_t) can be defined as follows:

$$L_t = \sum_{s \in N_s} I_t^s \tag{32}$$

We also consider that at each time instant t, an agent measures the QoS offered by the service provider, which in this case has been defined in terms of the number of sessions with latency violations (L_t). This allows us to obtain a sequence of observations $(L_1, L_2, \ldots, L_t)$ over time. The values of this metric are desired to be as small as possible, being the QoS optimal when the latency requirement of all the sessions is satisfied (i.e., $L_t = 0$). When the UPF placement and chaining are no longer optimal ($L_t \neq 0$), it must be readjusted to reestablish the system's QoS levels. However, these events consume significant resources and may create additional delays in the user plane and service interruption. Furthermore, reconfiguration events may produce additional costs due to the opening of new locations (servers), the deployment of extra VNF instances, the migration of existing UPFs, and the reassignment of PDU sessions. Consequently, unnecessary and frequent re-computation events must be avoided as much as possible and can only be triggered when strictly necessary. Here, we deal with the problem of determining the optimal time to initiate a reconfiguration procedure and minimize its adverse effects.

Moreover, we also assume that the system can tolerate a maximum number of sessions with latency violations, defined as θ where $\theta > 0$, without needing to trigger a reconfiguration event. In other words, if the value of the selected metric (L_t) is below the established threshold, the offered QoS is considered acceptable, and no reconfiguration is needed. However, once this upper bound is exceeded, the QoS is considered degraded, and the UPF placement and chaining configuration need to be readjusted. The service provider can define the θ parameter to satisfy service-level agreements related to application types or subscriber profiles.

The reconfiguration time must be selected so that the value of the QoS metric is as close as possible to the pre-established upper bound without exceeding it. In this way, UPF placement and chaining reconfiguration events can be delayed or avoided, and the QoS is kept at acceptable levels. In other words, we need to determine when the system is about to exceed the established threshold to activate a reconfiguration event in advance and avoid QoS deterioration. To facilitate the decision process, we present the following cost function:

$$Y_t(L_t) = \begin{cases} \Psi \cdot (\mathbb{E}[S_r] - L_t) & \text{if } L_t \leq \theta \\ \lambda \cdot \mathbb{E}[C_{rec}] & \text{if } L_t > \theta \end{cases} \tag{33}$$

Here, Ψ is a cost component and $\mathbb{E}[S_r]$ denotes the expected number of relocated sessions, while $\mathbb{E}[C_{rec}]$ is the expected reconfiguration cost, and λ is a weight factor to adjust the importance of the reconfiguration cost.

We define the loss function in (33) as a function of the number of sessions with latency violations and the maximum QoS tolerance threshold θ. Specifically, if the number of sessions with latency violations is below the established threshold, no readjustment is needed, and the service provider avoids paying its users an amount of money, $\Psi \cdot (\mathbb{E}[S_r] - L_t)$, proportional to the expected number of users with good QoS that will be affected due to the reconfiguration. In contrast, when the θ threshold is exceeded, the placement and chaining configuration must be readjusted, thus incurring an expected reconfiguration cost $\mathbb{E}[C_{rec}]$.

Our objective is to determine the time instance t where we can stop observing the QoS parameter and proceed to reconfigure the UPF placement and chaining configuration. In other words, we need to find the optimal stopping rule that minimizes the expected loss function (33).

Problem 1. Given a sequence of observations defined by L_t, an upper bound θ of the accepted tolerance on the QoS and an expected reconfiguration cost $\mathbb{E}[C_{rec}]$, determine the optimal decision epoch t^*, which minimizes the cost function Y_t:

$$\inf_{t\geq 0} \mathbb{E}[Y_t(L_t)] \tag{34}$$

4.2. An optimal scheduling reconfiguration rule

Optimal stopping theory is concerned with the problem of choosing a time to take a given action based on sequentially observed random variables to maximize an expected payoff or to minimize an expected cost [29]. OSPs are characterized by a sequence of observations $X_1, X_2, \ldots, X_t$ whose joint distribution is assumed to be known and a sequence of reward/cost functions $Y_1, Y_2, \ldots, Y_t$, where $Y_t = y_t(x_1, x_2, \ldots, x_t)$. Namely, an optimal stopping problem can be defined as follows. A decision maker or agent observes a sequence of random variables $(X_1, X_2, \ldots, X_t)$ and at each time epoch t, it must decide whether to stop observing and get the known reward Y_t or continue and observe the next variable X_{t+1}. The objective is to stop observing at the best time instant t^* for which the expected reward is maximized or the expected cost is minimized.

Problem 1 can be modeled as an OSP with infinite horizon where at each time instant or decision epoch t, one of the following decisions must be taken:

(i) stop and readjust the UPF placement and chaining configuration or,
(ii) continue to the next time interval $(t + 1)$ and keep the current configuration.

One approach widely used to solve OSPs due to its simplicity and efficiency is the 1-SLA rule, also called the myopic rule. The 1-SLA rule indicates at each decision epoch t whether to stop or continue according to the expected value of the cost function in the next time instant $(t + 1)$. Specifically, it calls for stopping at the first time t for which the cost Y_t for doing it is no more than the expected loss for continuing one stage and then stopping. Thus, the decision-making at each stage only depends on the current value of the random variable and the expected cost in the next stage.

Definition 1. For stopping problems, aimed at minimizing an expected cost, the 1-SLA rule is described by the stopping time

$$t^* = \inf \{t \geq 0 : Y_t \leq \mathbb{E}[Y_{t+1}|\mathbb{F}_t]\} \tag{35}$$

where $\mathbb{F}_t$ is the σ-fields generated by the observations $X_1, X_2, \ldots, X_t$. Specifically, it represents the knowledge of the random variable X_t up to time t.

An essential condition for which the 1-SLA stopping rule is optimal is that the problem is monotone.

Definition 2. Let A_t denote the event $\{Y_t \leq \mathbb{E}[Y_{t+1}|\mathbb{F}_t]\}$. The stopping problem is monotone if the sets A_t are monotone non-decreasing, i.e., $A_0 \subset A_1 \subset A_2 \ldots$ almost surely.

Namely, a monotone stopping rule problem can be described as follows: If the 1-SLA rule calls for stopping at stage t due to event A_t, then it will also call for stopping at all the future stages (e.g., $t + 1$, $t + 2$,...) regardless of the value of the future observations, since $A_t \subset A_{t+1} \subset A_{t+2} \ldots$

Theorem 1. *In monotone stopping rule problems, the 1-SLA rule is optimal.*

Proof. Refer to Chapter 5 of [29]. □

In the following, we derive an optimal stopping rule based on the 1-SLA rule to determine the optimal time at which is worst stopping observing and decide to reconfigure the UPF placement and chaining configuration.

Theorem 2. *Given an upper bound (θ) upon which QoS is degraded and a sequence of latency violations $L_1, \ldots, L_t$ w.r.t. the last optimal UPF placement ($L_0 = 0$), the optimal stopping time (t^*) for the problem stated in (34) is:*

$$\begin{aligned} t^* = & inf\{t \geq 0 : \Psi \cdot (\mathbb{E}[S_r] - L_t) \leq \lambda \cdot \mathbb{E}[C_{rec}] + \\ & \left(\Psi \cdot \mathbb{E}[S_r] - \lambda \cdot \mathbb{E}[C_{rec}]\right) \cdot \sum_{l=0}^{\theta} P(L = l)\} - \Psi \cdot \sum_{l=0}^{\theta} l \cdot P(L = l) \end{aligned} \tag{36}$$

Proof. Given a time interval t where $L_t \leq \theta$, the conditional expectation of Y_{t+1} at the next stage is given by:

$$\begin{aligned} \mathbb{E}[Y_{t+1}|L_t \leq \theta] = & \mathbb{E}[\Psi \cdot (\mathbb{E}[S_r] - L_{t+1})|L_t \leq \theta, L_{t+1} \leq \theta] \cdot P(L_{t+1} \leq \theta) + \\ & \mathbb{E}[\lambda \cdot \mathbb{E}[C_{rec}]|L_t \leq \theta, L_{t+1} > \theta] \cdot P(L_{t+1} > \theta) \\ = & \mathbb{E}[\Psi \cdot (\mathbb{E}[S_r] - L_{t+1})|L_{t+1} \leq \theta] \cdot P(L_{t+1} \leq \theta) + \\ & \mathbb{E}[\lambda \cdot \mathbb{E}[C_{rec}]|L_{t+1} > \theta] \cdot (1 - P(L_{t+1} \leq \theta)) \\ = & \Psi \cdot \left(\mathbb{E}[S_r] - \mathbb{E}[L_{t+1}|L_{t+1} \leq \theta]\right) \cdot P(L_{t+1} \leq \theta) + \\ & \lambda \cdot \mathbb{E}[C_{rec}] \cdot (1 - P(L_{t+1} \leq \theta)) \\ = & \left(\Psi \cdot \mathbb{E}[S_r] - \lambda \cdot \mathbb{E}[C_{rec}]\right) \cdot \sum_{l=0}^{\theta} P(L = l) + \lambda \cdot \mathbb{E}[C_{rec}] \\ & - \Psi \cdot \sum_{l=0}^{\theta} l \cdot P(L = l) \end{aligned}$$

Thus, by comparing the current cost, $Y_t(L_t) = \Psi(\mathbb{E}[S_r] - L_t)$, with the one expected at the next stage, we obtain that the UPF placement and chaining must be readjusted at the first time instance t such that $\Psi(\mathbb{E}[S_r] - L_t) \leq \mathbb{E}[Y_{t+1}|L_t \leq \theta]$. In other words, it is optimal to stop when the current loss is equal or less than the expected cost at the next stage. □

According to Theorem 1, the stopping rule presented in (36) must be monotone to guarantee the optimality of the 1-SLA rule.

Theorem 3. *The 1-SLA rule defined in (36) is optimal and minimizes the expected cost defined in (33).*

Proof. In order the problem be monotone, the difference $\mathbb{E}[Y_{t+1}|L_t \leq \theta] - Y_t(L_t)$ needs to be non-decreasing with L_t. This condition is satisfied since the right side of inequality (36) remains constant and its left side is non-increasing over L_t as long as L_t is below the predefined QoS threshold ($L_t \leq \theta$). Thus, the problem is monotone and the 1-SLA rule proposed in (36) is optimal. Please note that when $L_t > \theta$, we stop immediately and the UPF placement and chaining configuration is readjusted. □

5. Performance evaluation

This section outlines different experiments we conducted to investigate the performance of the proposed solutions. We first introduce some aspects of the simulation environment. We then evaluate the effectiveness of the UPC reconfiguration solutions by tuning different parameters. Finally, this section analyzes the performance of the scheduling mechanism.

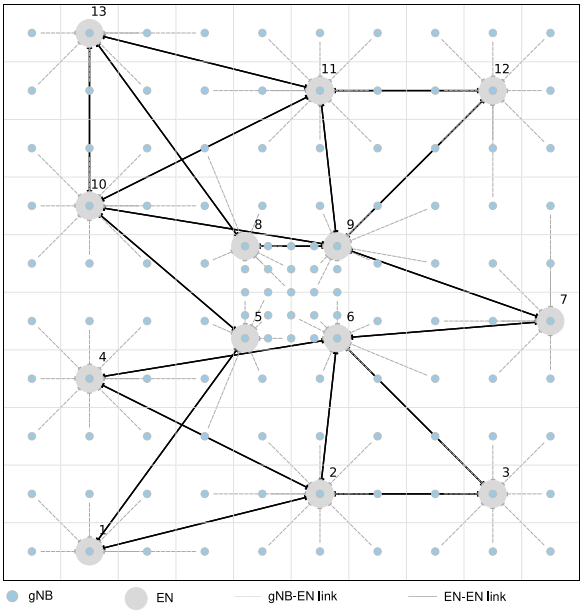


Fig. 2. 5G access network topology in a MEC ecosystem.

5.1. Simulation setup

We worked with a network topology representing a 5G medium-scale scenario (see Fig. 2). This scenario comprises 121 access nodes and 13 aggregation points (APs), and MEC servers. The gNBs are connected through APs, which are co-located with MEC servers. The EN service areas have a radius of 1 km, and the gNB inter-site distances are 500 m and 200 m for gNBs located in urban and dense urban areas, respectively. Some of these servers have already deployed UPF instances. We determined the initial UPF placement and chaining configuration through the PC-UPC heuristic proposed in [7]. We executed the PC-UPC algorithm by placing more importance on the node activation and deployment costs than on the routing term (i.e., $\alpha = 0.4$, $\beta = 0.4$, $\gamma = 0.2$).

For the service demand, we considered three types of SFCR, formed by one to three UPFs. We randomly generated the PDU session requirements, such as service latency, processing demand, and bandwidth, according to the parameters specified in Table 4. We also varied the number of active PDU sessions in the interval of [5-1000]. Table 4 summarizes the simulation parameters values.

We coded all the proposed solutions (i.e., ILP model and heuristic) using Python programming language 3.7. The optimal UPCR (O-UPCR) model was implemented in the Python-based package Pyomo [30], and Gurobi [31] was selected as its underlying solver. Additionally, we selected the CityMob [32] mobility pattern generator with a Manhattan model to simulate user mobility. Moreover, we developed a program to manage user location and assigned gNB and UPFs over the entire simulation time. All the experiments were performed on a computer with 64 GB of RAM and a 3.30 GHz Intel Core-i9 processor.

5.2. UPC reconfiguration solutions

We solved the UPC reconfiguration problem by considering two sets of weight factors. The first one (weight_set_1) assumes similar weights for the cost components but omits the reassignment term from the objective function. The second set (weight_set_2) assumes that all the cost components need to be optimized and are equally important. Additionally, each sample contains ten reconfiguration events performed by a periodic scheduling mechanism with a reconfiguration time of 30 min.

Table 4
Simulation parameters.

Notation	Description	Value
PDU sessions		
S	Number of PDU sessions	[5–1000]
β_s	Bandwidth requirement (Mbps)	[1, 10, 50]
C_s	CPU processing demand	[0.01, 0.05, 0.1]
L_s	Latency requirement (ms)	[0.95, 1]
UPF placement		
N_r	No. of access nodes	121
N_c	No. of candidate locations	13
E	Number of links	172
P	Number of shortest paths	1742
$\beta_{u,v}$	Link bandwidth capacity (Gbps)	10
C_c	Server processing capacity (CPU)	40
C_t	VNF processing capacity (CPU)	2
I_t	Maximum number of instances	+3[a]
d_r	RTT delay in the RAN (μs)	500
d_t	Processing time of UPFs (μs)	50
T_{ap}	Processing time of AP (μs)	5
d_{DN}	Processing time of DN (μs)	100
–	Prop. delay of links (μs/km)	5
CityMob		
m	Mobility model	2[b]
n	Number of users	1000
t	Simulation time (s)	36000
s	Maximum speed of users (m/s)	15
d	Distance between streets (m)	100
wxd	Dimensions of the grid (km^2)	5 × 5
a	Number of accidents	0

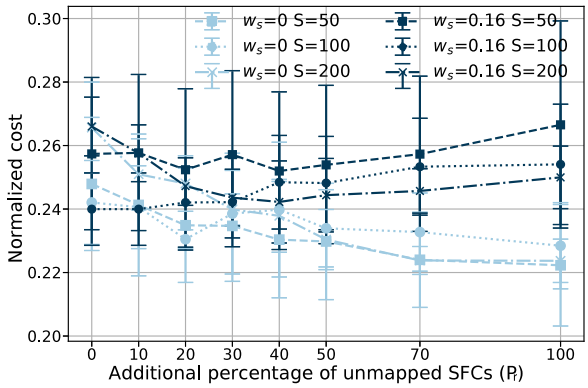
[a]For each VNF type the maximum number of instances was determined by adding three extra instances to the minimum number of VNFs required to meet SFCR's demands.
[b]The m parameter takes numeric values to indicate the mobility model (e.g., m = 2 for the Manhattan model).

5.2.1. DPC-UPCR performance

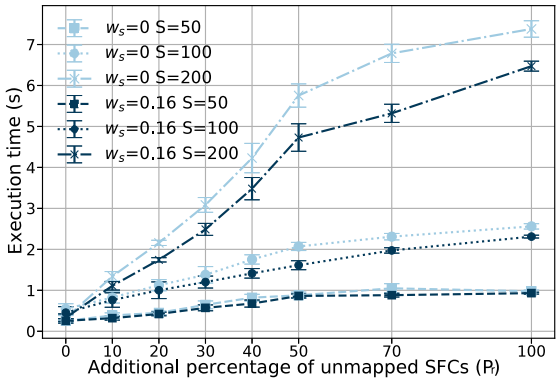
The following content analyzes the effects of the partial SFC unmapping and the improvement phase on the performance of the proposed heuristic, with a particular focus on the average reconfiguration cost and computation time. To achieve this aim, we considered different numbers of active PDU sessions (i.e., $S = 50$, $S = 100$, and $S = 200$).

Partial unmapping: We performed these experiments by unmapping an additional percentage of SFCs (P_r) apart from those with latency violations at the reconfiguration moment. This additional set of sections was formed by SFCs closer to exceeding their service latency budget. Moreover, we also unmapped extra sessions assigned to UPF instances with a capacity utilization below 20%. Moreover, we did not consider any improvement phase for the heuristic solution for these experiments.

Fig. 3(a) illustrates the average reconfiguration cost for different percentages of additional unmapped sessions for both sets of weight factors. This figure highlights how the behavior of the cost function changed notably according to the considered cost components. More specifically, the reconfiguration cost tended to decrease with the percentage of unmapped sessions for the first set of weights that omitted the reassignment term from (7) (represented in a light blue color). For all the analyzed groups of SFCs, the highest reconfiguration costs were obtained when only the sessions with latency violations at the reconfiguration moment were unmapped. In contrast, the lowest costs were obtained when unmapping the entire set of SFCRs. This behavior was expected given that there are more combinations, and therefore possibilities, of finding better UPF locations and mapping when the number of unmapped sessions is greater. The latter is sustained by the behavior of the number of relocated sessions, which increased in value with the P_r parameter (see Fig. 4(a)). It should be noted that the relocation cost is not reflected in the overall reconfiguration cost since this term was omitted from the objective function. Moreover, greater values of P_r also allowed further improvements in the average system response time. This occurred despite significant reductions in the total number of deployed UPFs (see Figs. 4(b) and 4(c)).



(a) Average reconfiguration cost.



(b) Average execution time.

Fig. 3. Average reconfiguration cost and computing time for the proposed heuristic vs. different percentage of partially unmapped SFCs (P_r). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

For the second set of weights, represented in a dark blue color, the behavior of the cost function is a bit more complex, as it does not immediately seem to follow a clear trend related to the P_r parameter (see Fig. 3(a)). Contrary to our initial thoughts, a partial unmapping resulted in lower reconfiguration costs than a full one in some cases (i.e., $S = 50$ and $S = 100$). The reason for this is that the number of relocated sessions increased with the value of the percentage of unmapped sessions (see Fig. 4(a)). Additionally, for the group of 200 PDU sessions, the cost function decreased for values of $P_r \leq 40$. In contrast, there was a slight increase for $P_r \geq 50$ due to higher variations in the number of reassigned sessions. Nevertheless, unlike the other two groups of sessions, the cost obtained for a complete unmapping was much lower than when only SFCRs with poor QoS were reassigned ($P_r = 0$). By more closely examining the relocation and cost functions, we see that both graphics had similar behavior when the number of sessions was small ($S = 50$ and $S = 100$). This is because the impact of a relocated session on the normalized cost function is greater when the number of sessions is small rather than large. This behavior was more noticeable for S = 100, which experienced scarce improvements in its total number of deployed instances.

Fig. 4(c) shows that the second set of weight factors presented higher average latency than the first one. For this set, in particular, the metric values rose with the P_r parameter. This behavior was because the number of UPF instances usually decreased with P_r and that, unlike the results obtained for the first set, a considerably lower number of sessions were relocated.

Regarding the computing time, Fig. 3(b) shows that both sets of weights increased with the percentage of unmapped SFCRs. Thus, requiring the highest execution time when a complete UPF placement and chaining readjustment (i.e., $P_r = 100$) was considered. Moreover, it should be noted that the value of this metric is greater for the second rather than the first set of weights since it required the evaluation of an extra parameter (i.e., the relocation of an SFCR) when analyzing the candidate locations.

These results show that a partial rather than complete unmapping is a more appropriate option when the relocation cost component is considered and the number of SFCRs is small. More concretely, we obtained a relatively strong cost performance when considering partial unmapping with 30%–50% of the sections. In contrast, when this cost component is omitted, a complete rather than partial unmapping provides higher reductions in the reconfiguration cost. Nevertheless, the complete unmapping still produces higher execution times. For both cases, a partial unmapping of the set of SFCRs may be implemented when short running times during the UPCR are desired.

Improvement phase:

Regarding the evaluation of the improvement phase on the performance of the proposed heuristic, we ran several iterations (i.e., five) to reflex a more stable behavior. This is recommended given that each iteration produced a different configuration due to the random selection of the VNF instances and sorting criteria. Additionally, all the components in the objective function were assumed to have the same importance (i.e., weight_set_2).

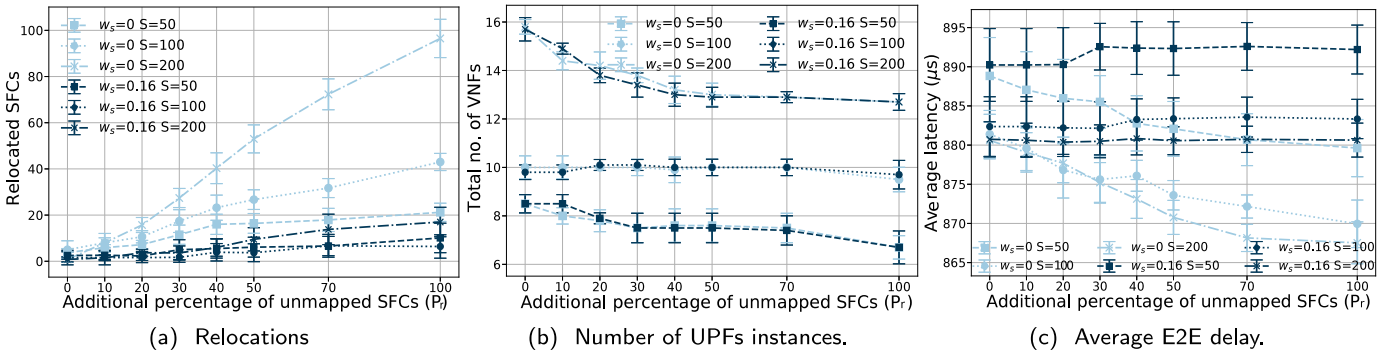
Fig. 5 depicts the behavior of the average reconfiguration cost and execution time for different numbers of improvement attempts (F_i). It also includes the results for the heuristic when no improvement (i.e., $F_i = 0$) was considered. Fig. 5(a) shows that the cost of the solutions was reduced, even in the simplest case, with only one improvement attempt in each reconfiguration event. These reductions in the average reconfiguration cost incremented with the number of improvement attempts. The latter was more remarkable for the group formed by 50 SFCRs, which experienced decrements between 1% and 5.6% for the analyzed values of the F_i parameter compared with the version of the DPC-UPCR with no improvement. For the other two groups of SFCRs, the reductions were not as significant as in the case of $S = 50$ SFCRs because the quality of their reconfiguration solutions was closer to the optimal one, with an optimality gap of 5%. Please refer to 5.2.2 for more details in this regard.

However, these improvements occurred at the expense of higher execution times for the reconfiguration events, which incremented with F_i (see Fig. 5(b)). For instance, the execution times for $F_i = 15$ in relation to the non-improved reconfiguration solutions ($F_i = 0$) were between four and seven times greater than those for the selected SFC groups.

5.2.2. UPCR solutions performance

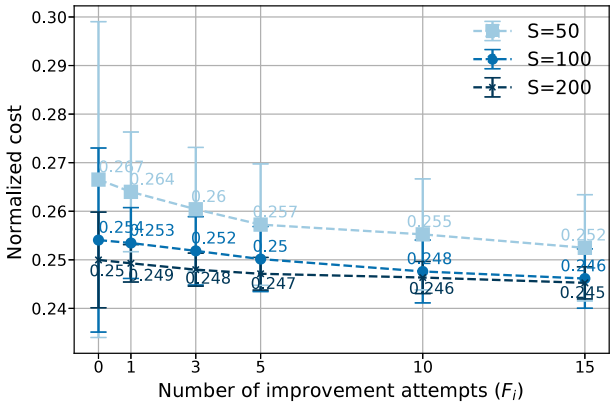
This subsection evaluates the performance of the conceived ILP model (O-UPCR) and DPC-UPCR heuristic. For the heuristic, we defined two variants depending on whether or not the improvement phase was considered. When no improvement attempt is contemplated ($F_i = 0$), we refer to this solution as a basic version of the DPC-UPCR (BDPC-UPCR). Otherwise, we call it the improved DPC-UPCR heuristic (IDPC-UPCR). These names are only for notation convenience, but both solutions derive from the proposed heuristic. For the IDPC-UPCR, 15 improvement attempts were defined ($F_i = 15$). Additionally, we determined the solutions to the ILP model with a zero optimality gap.

We compared the BDPC-UPCR with two baselines referred to as Greedy- and Sorted Greedy-UPCR (i.e., G-UPCR and SG-UPCR). These baselines are greedy-based approaches that map each SFCR to the VNFs and candidate locations that minimize the objective function of the UPCR problem in (7). The G-UPCR does not implement any sorting criteria to map SFCRs, whereas the SG-UPCR follows the same criteria as the proposed heuristic. Moreover, they do not consider the effects of a VNF mapping decision on the next VNFs that form the SFC, which

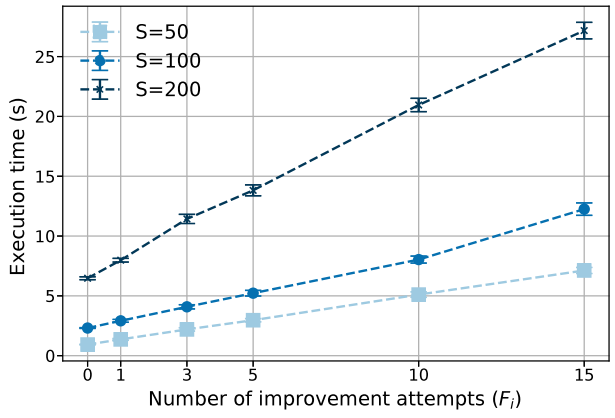


(a) Relocations (b) Number of UPFs instances. (c) Average E2E delay.

Fig. 4. Performance of the proposed heuristic with partial unmapping in terms of relocated sessions, deployed VNF instances and average latency.



(a) Average reconfiguration cost.



(b) Average execution time.

Fig. 5. Average reconfiguration cost and computing time of the proposed heuristic vs. different number of improvement attempts.

may result in the rejections of some SFCRs during their mapping procedure. We extended both baselines to include a reassignment procedure to avoid rejections and ensure similar comparison conditions. This procedure moves back a step in the SFCR mapping process by selecting a different candidate to serve the last mapped VNF in the chain. The reassignment procedure is executed when a VNF cannot be mapped due to the absence of feasible candidates, and it is repeated until the current VNF has been successfully mapped or no feasible candidate is left to remap the previous VNF.

Table 5
Solutions summary.

Name	Solution approach	P[a]	B[b]
O-UPCR	ILP model	×	
BDPC-UPCR	DPC-UPCR heuristic with $F_i = 0$	×	
IDPC-UPCR	DPC-UPCR heuristic with $F_i = 15$	×	
G-UPCR	Greedy-based heuristic		×
SG-UPCR	Greedy-based heuristic		×

[a]P: proposed solution.
[b]B: baseline solution.

Table 5 summarizes all the solutions understudy in this subsection by specifying their solution approach and whether they are part of our proposal or a baseline. Please note that all these solutions consider a full unmapping of the SFCR set and seek to minimize the objective function (7). Furthermore, we investigated their performance in terms of average reconfiguration time and overall normalized reconfiguration cost for different demands of PDU sessions. We also analyzed the number of reassignment events when comparing the BDPC-UPCR with the baselines. For these experiments, all the cost components in the objective function were considered to be equally important (i.e., weight_set_2).

Fig. 6 represents the normalized reconfiguration cost, number of reassignment events, and computing time for the BDPC-UPCR and both greedy approaches. These results indicate that the proposed heuristic outperformed the baselines for all the analyzed metrics. The BDPC-UPC solution always provided the lowest reconfiguration cost with significant differences in relation to the benchmarks. In the best case, it achieved reductions up to 24% for GH-UPCR and 21% for SGH-UPCR. In addition, BDPC-UPC was always able to map all the SFCRs without requiring any reassignment procedure, while the number of reassignment events triggered by the baselines increased with the number of PDU session requests. Additionally, Fig. 6(c) shows that the benchmarks had similar running times. Moreover, their differences with our proposed heuristic were more remarkable as the size of the SFCR set increased because they were required to perform more reassignment events to avoid SFCR rejections.

Fig. 7(a) depicts the average reconfiguration cost of the O-UPCR, BDPC-UPCR, and IDPC-UPCR solutions for increasing demands of SFCRs. This figure demonstrates how the performance of the heuristic in its simplest version (i.e., BDPC-UPC) was always within 15% of the optimum, with an average optimality gap of 7.27%. This difference was further narrowed by incorporating the improvement phase and the selected value of the F_i parameter (i.e., $F_i = 15$). More specifically, the average optimality gap for this approach was 4.25% of the optimum, with a difference of 8.62% in the worst case (i.e., $S = 50$). It should be noted that higher reductions were achieved for values of $S \geq 100$

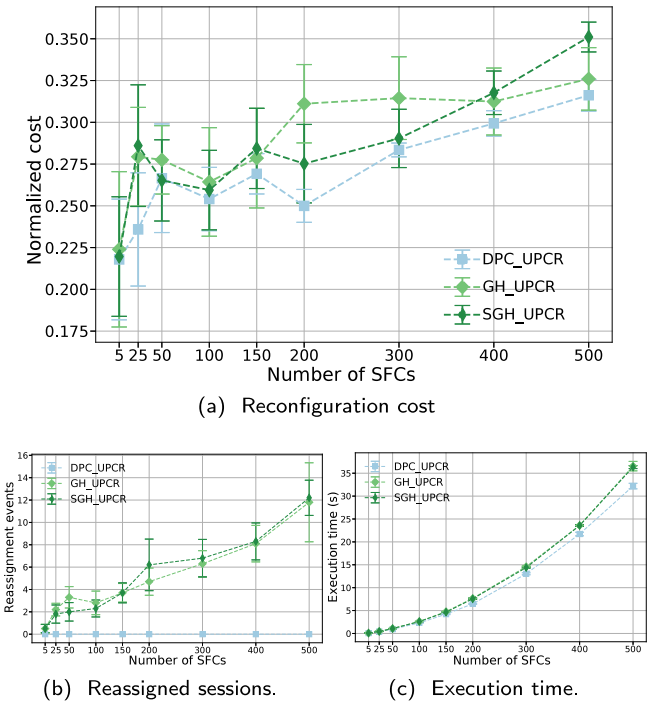


Fig. 6. Performance of the proposed heuristic vs. greedy approaches.

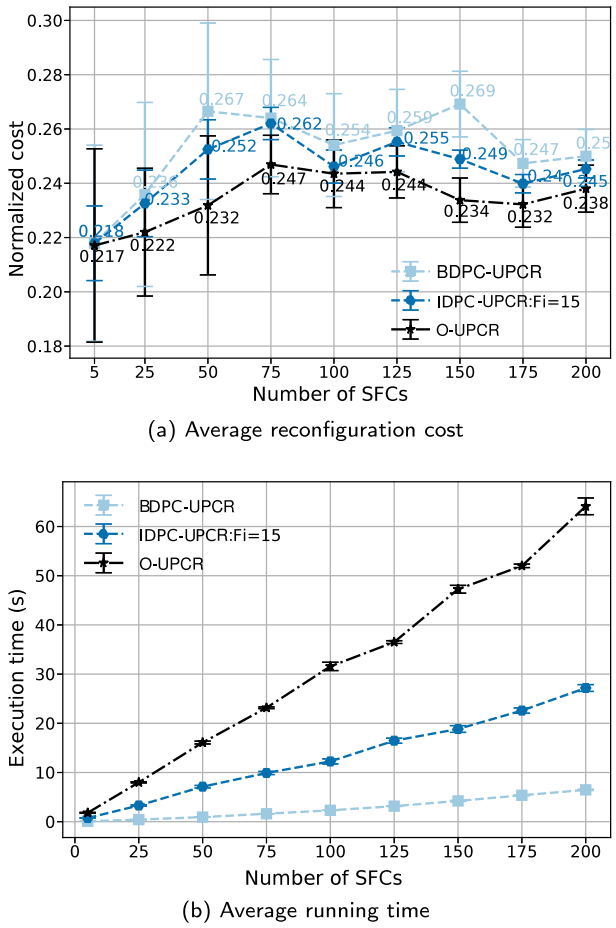


Fig. 7. Performance of the proposed solutions vs. different number of SFCRs.

since there were a greater number of deployed VNFs and thus of possibilities for mapping the selected SFCRs. The results show that the proposed improvement approach allows for significant enhancements in the quality of the reconfiguration solutions with near-optimal results.

Fig. 7(b) summarizes the average runtime of the proposed solutions in terms of CPU. This figure illustrates how the computational time for all solutions increased with the number of SFCRs. As was expected, the ILP model required the highest execution time, whereas the BDPC-UPC was the fastest approach. In general, the basic heuristic approach was between 4–9 and 10–22 times faster than its improved version and the mathematical model, respectively. Moreover, both heuristics scaled well with increasing numbers of SFCRs, while the O-UPCR's running time raised considerably until the point of stopping finding solutions to the problem, in a reasonable time, for SFCRs greater than 200. Regarding the wall-clock time parameter, it should be noted that the ILP model took several hours to perform a reconfiguration event for instances of the problem with more than 150 PDU sessions, while the heuristic approaches only required a few seconds.

5.3. Dynamic scheduling mechanisms

To verify the effectiveness of the proposed scheduling mechanism, we established the following benchmarks:

- Periodic Scheduling Reconfiguration (PSR): The UPC readjustment is executed periodically at fixed time intervals (i.e., every 30 and 60 min).
- Skeptical Scheduling Reconfiguration (SSR) [13]: The UPC is reevaluated in relation to a maximum threshold of allowed latency violations (θ). Similar to the proposed mechanism (OSR), its objective is to tolerate as many latency violations as possible, at each sampling time, without exceeding the established QoS threshold. However, the SSR approach expresses the reconfiguration cost regarding the expected number of relocated sessions.

For these simulations, we ran the system for ten hours and measured the established QoS metric with a sampling time of one minute for 600 samples. We analyzed the performance of the aforementioned scheduling mechanisms based on the following metrics: the number of reconfiguration events, the number of latency violations at the reconfiguration moment, and QoS status at each sampling time. To estimate the number of relocated sessions and reconfiguration cost for the OSR model, we used the results obtained from the periodic reconfiguration with a period of 30 min between each reconfiguration event (PSR_T30). Moreover, we selected an upper bound (θ) on the QoS metric of 3% of sessions with latency violations for the SSR and OSR mechanisms. The number of sessions with latency violations was modeled as a Poisson distribution with a mean of $\lambda = 27$. This distribution was fitted on observed latency violations for a UPC configuration without readjustment for two hours.

To meet the established sampling time, the used UPC reconfiguration solutions must be able to determine the new configurations in less than a minute. Following this criterion, we adjusted the proposed heuristic for a partial unmapping with 30% of additional sessions and one improvement attempt (i.e., $P_r = 30$ and $F_i = 1$). Furthermore, these experiments were conducted by considering both sets of weights (weight_set_1 and weight_set_2) in the objective function.

5.3.1. Reconfiguration events

Fig. 8 depicts the *number of reconfiguration events* and the number of sessions with latency violations at the time of the reconfiguration procedures. To provide a more intuitive and legible representation, we classified the number of latency violations into three groups in relation to the θ value (low, medium, and high). These groups were represented

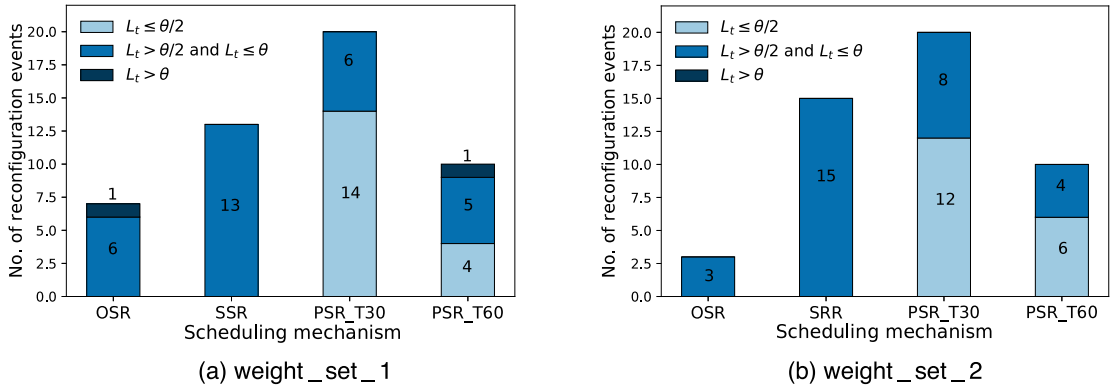


Fig. 8. Number of reconfiguration events w.r.t. QoS values at the reconfiguration moment for different scheduling mechanisms and sets of weight factors. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

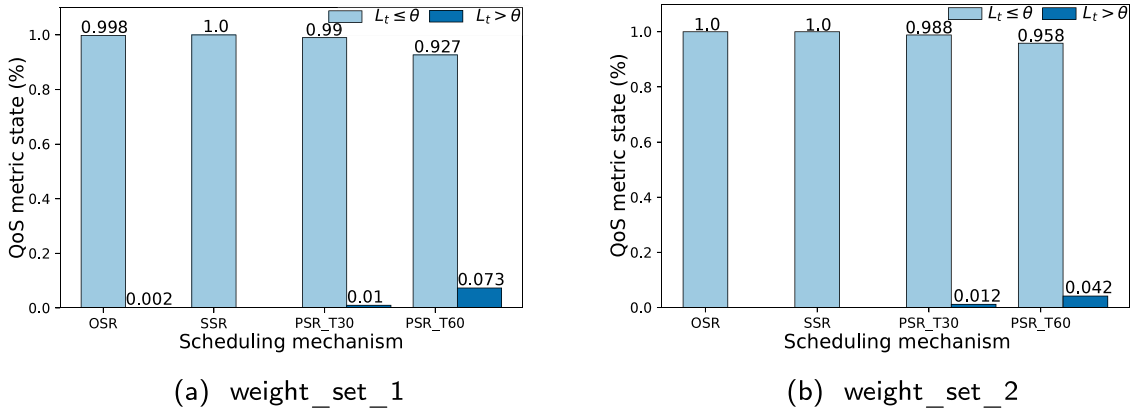


Fig. 9. QoS status for different scheduling mechanisms and sets of weight factors.

with different shades of blue color (from light to dark). This figure highlights how our proposed mechanism not only performed the fewest number of reevaluations for both pairs of weight sets but also triggered these events when the QoS threshold was close to being exceeded. Similarly, the SRR benchmark never readjusted the placement and chaining configuration when the number of sessions with latency violations was low. However, it triggered twice and five times more reconfiguration events than the OSR solution for the first and second pair of weights, respectively.

In contrast, most of the reconfigurations performed by the periodic approaches were executed when the number of sessions with a poor QoS was low. Specifically, more than 60% and 40% of the reconfigurations for the PSR_T30 and PSR_T60 benchmarks, respectively, were triggered when $L_t \leq \theta/2$. In general, the OSR mechanism reduced the number of reconfiguration events between 65%–85% and 30%–70% compared to PSR_T30 and PST_T60, respectively.

5.3.2. QoS metric

Fig. 9 summarizes the *status of the QoS* in relation to the established upper bound. This figure demonstrates that the OST-based solutions (i.e., OSR and SSR) had the best performances. For these two mechanisms, the QoS metric was above the established threshold only one time for the first set of weights and never for the second one. Thus, they were able to keep the QoS under the desired values for almost the entire simulation time in both cases. In contrast, the periodic schedulers had the worst results since they produced a higher number of violations of the established QoS threshold. For both sets, the PSR_T30 resulted in a poor QoS around 1% of the simulation time despite requiring a higher number of reconfiguration events. Additionally, the PSR baseline with UPC readjustment every hour had a greater number of events with a poor QoS. More specifically, the PSR_T60 mechanism resulted in a poor QoS around 7.3% of the simulation time for the first set and 4.2% for the second one.

Fig. 10 illustrates the instantaneous and cumulative values of the *QoS metric over time* for both sets of weight factors. It also represents the reconfiguration moments through dashed gray lines. This figure shows how the OSR mechanism always triggered the reconfiguration when the number of sessions with a poor QoS was close to the established upper bound. With our proposed solution, the QoS threshold was exceeded only one time, and a reconfiguration process was immediately executed. In contrast, when the number of sessions with latency violations was small, no reconfiguration was required by the OSR. The SSR solution showed similar behavior. However, this method produced more readjustment events than our proposed scheduler. The cause for this lies in the method's stopping rule, which decides to reconfigure at lower values of the QoS parameter. Moreover, by more closely examining the representation of the reconfiguration events, we can observe that the SSR and OSR schedulers did not have a fixed frequency. Both approaches depend on the values of the selected QoS metric, which is a random variable.

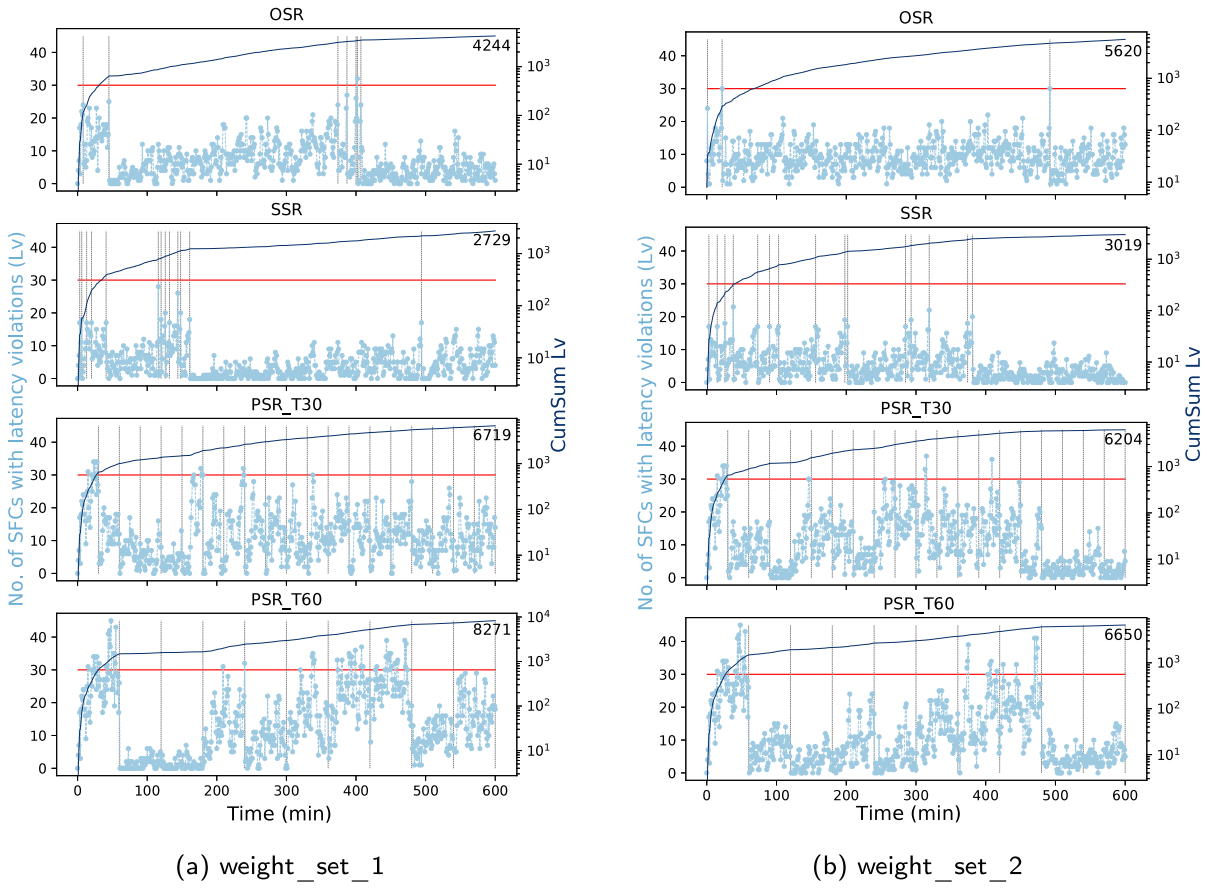


Fig. 10. Number of sessions with latency violation and their cumulative sum per time instance for partial heuristic with P = 40 and not improvement phase.

Regarding the *cumulative sum of sessions with latency violations*, our proposed mechanism had lower values than either variant of the PSR despite performing fewer reconfiguration events. This is because it was able to keep the QoS parameter under acceptable values the whole simulation time, not being the case for the periodic approaches since they are unaware of the status of the QoS metric. Additionally, the PSR method with a reconfiguration period of 60 min (PSR_T60) had the greatest number of sessions with latency violations. These results were expected given that this approach is not only unaware of the QoS status but also takes a greater amount of time between reconfigurations.

6. Conclusion

This paper addresses the problem of dynamic UPF placement and chaining reconfiguration in a MEC ecosystem. We have formulated the problem as an ILP model targeted at reducing the total reconfiguration cost. This cost is expressed as the combination of multiple cost components such as server activation, SFC relocation, and VNF migration. The objective of this model is to determine the optimal SFCR mapping and UPF placement configuration to minimize the overall cost while ensuring 5G stringent latency requirements along with UPF and SFC specificities.

We present a heuristic algorithm (DPC-UPC) to confront the model's computational complexity and provide online solutions to the problem. We evaluated the performance of the proposed heuristic by comparing it with two greedy approaches and the optimal solution. The results demonstrate that DPC-UPC outperformed both heuristics in terms of reconfiguration cost, execution time, and the number of reassignment events. Furthermore, DPC-UPC was able to provide near-optimal solutions in considerably less time than the benchmarks. More concretely, we determined solutions to the problem with average optimality gaps of 7.27% and 4.25% when the improvement phase was omitted and considered, respectively.

Additionally, we envisioned a scheduler mechanism, OSR, based on OST to determine the optimal UPCR time according to the established QoS threshold and instantaneous values of the selected metric (i.e., the number of sessions with latency violations). The conducted experiments revealed significant improvements in the offered QoS and the number of reconfiguration events as compared with the baselines. More specifically, OSR not only offered the best QoS by keeping the QoS metric under the desired values for almost the entire simulation time but also required the minimum number of reconfiguration events (i.e., with reductions between 30% and 85%).

Directions for future research include designing a reconfiguration mechanism to pre-determine placement and chaining reconfigurations capable of adapting to future user locations (i.e., anticipatory UPC reconfiguration based on mobility predictions) to reduce the impact of the reconfiguration events on the QoS and the QoE of the users.

CRediT authorship contribution statement

Irian Leyva-Pupo: Conceptualization, Formal analysis, Methodology, Data curation, Validation, Visualization, Software, Writing – original draft. **Cristina Cervelló-Pastor:** Conceptualization, Formal analysis, Validation, Methodology, Supervision, Writing – review & editing, Visualization, Funding acquisition, Project administration. **Christos Anagnostopoulos:** Conceptualization, Formal analysis, Validation, Methodology, Supervision, Writing – review & editing. **Dimitrios P. Pezaros:** Conceptualization, Formal analysis, Validation, Methodology, Supervision, Writing – review & editing, Funding acquisition, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work has been supported by the Agencia Estatal de Investigación of Ministerio de Ciencia e Innovación of Spain under project PID2019-108713RB-C51/ MCIN/ AEI/ 10.13039/501100011033 and through a predoctoral FPI scholarship. Additionally, this work was supported in part by the PETRAS National Centre of Excellence for IoT Systems Cybersecurity, which has been funded by the UK EPSRC under grant number EP/S035362/1

References

[1] 5G Americas, 5G Services and Use Cases, Technical report, 5G Americas, 2017, URL https://www.5gamericas.org/wp-content/uploads/2019/07/5G_Service_and_Use_Cases_FINAL.pdf.

[2] Rashid Mijumbi, Joan Serrat, Juan-Luis Gorricho, Niels Bouten, Filip De Turck, Raouf Boutaba, Network function virtualization: State-of-the-art and research challenges, IEEE Commun. Surv. Tutor. 18 (1) (2016) 236–262, http://dx.doi.org/10.1109/COMST.2015.2477041.

[3] Yun Chao Hu, Milan Patel, Dario Sabella, Nurit Sprecher, Valerie Young, Mobile edge computing—A key technology towards 5G, ETSI White Paper 11 (11) (2015) 1–16.

[4] Bego Blanco, Jose Oscar Fajardo, Ioannis Giannoulakis, Emmanouil Kafetzakis, Shuping Peng, Jordi Pérez-Romero, Irena Trajkovska, Pouria S. Khodashenas, Leonardo Goratti, Michele Paolino, et al., Technology pillars in the architecture of future 5G mobile networks: NFV, MEC and SDN, Comput. Stand. Interfaces 54 (2017) 216–228.

[5] 3GPP, System Architecture for the 5G System (5GS); Stage 2, Technical Specification (TS) 23.501, 3rd Generation Partnership Project (3GPP), 2020, URL https://www.3gpp.org/ftp/Specs/archive/23_series/23.501/23501-g40.zip. Version 16.4.0.

[6] Muhammad Ali Imran, Yusuf Abdulrahman Sambo, Qammer H. Abbasi, Enabling 5G Communication Systems to Support Vertical Industries, John Wiley & Sons, 2019.

[7] Irian Leyva-Pupo, Cristina Cervelló-Pastor, Efficient solutions to the placement and chaining problem of user plane functions in 5G networks, J. Netw. Comput. Appl. (2021) 103269.

[8] Sebastian Peters, Manzoor A. Khan, Anticipatory user plane management for 5G, in: 2018 IEEE 8th International Symposium on Cloud and Service Computing (SC2), IEEE, 2018, pp. 9–15.

[9] Sebastian Peters, Manzoor Ahmed Khan, Anticipatory session management and user plane function placement for AI-driven beyond 5G networks, Procedia Comput. Sci. 160 (2019) 214–223.

[10] Irian Leyva-Pupo, Alejandro Santoyo-González, Cristina Cervelló-Pastor, A framework for the joint placement of edge service infrastructure and user plane functions for 5G, Sensors 19 (18) (2019) 3975.

[11] Yuanzhe Li, Xiao Ma, Mengwei Xu, Ao Zhou, Qibo Sun, Ning Zhang, Shangguang Wang, Joint placement of UPF and edge server for 6G network, IEEE Internet Things J. (2021).

[12] Tejas Subramanya, Davit Harutyunyan, Roberto Riggio, Machine learning-driven service function chain placement and scaling in MEC-enabled 5G networks, Comput. Netw. 166 (2020) 106980.

[13] Irian Leyva-Pupo, Cristina Cervelló-Pastor, Christos Anagnostopoulos, Dimitrios P. Pezaros, Dynamic scheduling and optimal reconfiguration of UPF placement in 5G networks, in: Proceedings of the 23rd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, 2020, pp. 103–111.

[14] Junjie Liu, Wei Lu, Fen Zhou, Ping Lu, Zuqing Zhu, On dynamic service function chain deployment and readjustment, IEEE Trans. Netw. Serv. Manag. 14 (3) (2017) 543–553.

[15] Narjes Tahghigh Jahromi, Somayeh Kianpisheh, Roch H. Glitho, Online VNF placement and chaining for value-added services in content delivery networks, in: 2018 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN), IEEE, 2018, pp. 19–24.

[16] Gustavo Miotto, Marcelo Caggiani Luizelli, Weverton Luis da Costa Cordeiro, Luciano Paschoal Gaspary, Adaptive placement & chaining of virtual network functions with NFV-PEAR, J. Internet Serv. Appl. 10 (1) (2019) 1–19.

[17] Yicen Liu, Hao Lu, Xi Li, Donghao Zhao, An approach for service function chain reconfiguration in network function virtualization architectures, IEEE Access 7 (2019) 147224–147237.

[18] Biyi Li, Bo Cheng, Junliang Chen, A multi-stage approach for virtual network function migration and service function chain reconfiguration in NFV-enabled networks, in: 2020 IEEE International Conference on Web Services (ICWS), IEEE, 2020, pp. 207–215.

[19] Yan-Ting Chen, Wanjiun Liao, Mobility-aware service function chaining in 5G wireless networks with mobile edge computing, in: ICC 2019-2019 IEEE International Conference on Communications (ICC), IEEE, 2019, pp. 1–6.

[20] Yunjie Gu, Yuxiang Hu, Yuehang Ding, Jie Lu, Jichao Xie, Elastic virtual network function orchestration policy based on workload prediction, IEEE Access 7 (2019) 96868–96878.

[21] Lei Wang, Mahdi Dolati, Majid Ghaderi, CHANGE: Delay-aware service function chain orchestration at the edge, in: 2021 IEEE 5th International Conference on Fog and Edge Computing (ICFEC), 2021, pp. 19–28, http://dx.doi.org/10.1109/ICFEC51620.2021.00011.

[22] Richard Cziva, Christos Anagnostopoulos, Dimitrios P Pezaros, Dynamic, latency-optimal vNF placement at the network edge, in: IEEE INFOCOM 2018- IEEE Conference on Computer Communications, IEEE, 2018, pp. 693–701.

[23] Jintian Hu, Gaocai Wang, Xiaotong Xu, Yuting Lu, Study on dynamic service migration strategy with energy optimization in mobile edge computing, Mob. Inf. Syst. 2019 (2019) 1–12.

[24] Christos Anagnostopoulos, Kostas Kolomvatsos, An intelligent, time-optimized monitoring scheme for edge nodes, J. Netw. Comput. Appl. 148 (2019) 102458.

[25] Jia Yan, Suzhi Bi, Ying-Jun Angela Zhang, Optimal model placement and online model splitting for device-edge co-inference, 2021, arXiv preprint arXiv:2105.13618.

[26] S. Homma, T. Miyasaka, S. Matsushima, D. Voyer, User plane protocol and architectural analysis on 3GPP 5G system, in: IETF, Internet-Draft Draft-Ietf-Dmm-5g-Uplane-Analysis-04 (Work in Progress), 2020, URL https://tools.ietf.org/pdf/draft-ietf-dmm-5g-uplane-analysis-04.pdf.

[27] Marcelo Caggiani Luizelli, Weverton Luis da Costa Cordeiro, Luciana S. Buriol, Luciano Paschoal Gaspary, A fix-and-optimize approach for efficient and large scale virtual network function placement and chaining, Comput. Commun. 102 (2017) 67–77.

[28] Xincai Fei, Fangming Liu, Hong Xu, Hai Jin, Adaptive VNF scaling and flow routing with proactive demand prediction, in: IEEE INFOCOM 2018-IEEE Conference on Computer Communications, IEEE, 2018, pp. 486–494.

[29] Thomas S. Ferguson, Optimal stopping and applications, 2006, URL https://www.math.ucla.edu/~tom/Stopping/Contents.html.

[30] William E. Hart, Carl D. Laird, Jean-Paul Watson, David L. Woodruff, Gabriel A. Hackebeil, Bethany L. Nicholson, John D. Siirola, Pyomo–Optimization Modeling in Python, Vol. 67, second ed., Springer Science & Business Media, 2017.

[31] Gurobi Optimization, LLC, Gurobi optimizer, 2021, URL http://www.gurobi.com.

[32] Francisco J. Martinez, J.-C. Cano, Carlos T. Calafate, Pietro Manzoni, Citymob: a mobility model pattern generator for VANETs, in: ICC Workshops-2008 IEEE International Conference on Communications Workshops, IEEE, 2008, pp. 370–374.

Irian Leyva-Pupo received her BSc. degree in Telecommunication and Electronic Engineering from the Havana University of Technology Jose Antonio Echeverria (Havana, Cuba). She is currently a Ph.D. student at the Universitat Politècnica de Catalunya (UPC), Barcelona, Spain. Her main research interests include optimization problems, NFV, SDN, and 5G networks.

Cristina Cervelló-Pastor received her MSc. and PhD. degrees in Telecommunication Engineering, both from the Barcelona School of Telecommunications Engineering (ETSETB), Universitat Politècnica de Catalunya (UPC), Barcelona, Spain. She is an Associate Professor and she has been the Head of the Network Engineering Department at UPC from 2012 to 2020. She is currently the director of the Castelldefels School of Telecommunications and Aerospace Engineering (EETAC) and the leader of the research group on Broadband Network Modeling and Evaluation (BAMPLA). She has been responsible and actively participated in diverse national and European competitive projects (IA@EDGE, 5GCity, NOVI, FEDERICA, ATDMA, A@DAN, Euro-NGI, Euro-FGI, EURO-NF, etc.) and private funding R&D projects. In parallel, she has published diverse papers in national and international journals and conferences and she has been supervising several theses in the field of optimization, management, optimal resource allocation, topology discovery, and routing in SDN/NFV and 5G.

Dr **Christos Anagnostopoulos** is an Associate Professor in Distributed and Pervasive Computing, School of Computing Science, University of Glasgow. His expertise is in the areas of large-scale distributed data systems and in-network data management. He has received funding for his research by the EC/H2020, UK EPSRC and the industry. He is an author of over 160 refereed scientific journals/conferences [scholar]. He is leading the Essence: Pervasive & Distributed Computing Lab within the Knowledge and Data Engineering Systems Group (IDA Section). Before joining Glasgow, he was an Assistant Professor at Ionian University (2013) and Adjunct Assistant Professor at the University of Athens (2009). He has held Postdoctoral position at University of Athens and Research Fellow at University of Glasgow in the area of contextaware & large-scale distributed computing. He holds a B.Sc., M.Sc., and Ph.D. in Computing Science, University of Athens (2008), he is an associate fellow of the HEA and member of ACM, IEEE and IEEE STC.

Dimitrios Pezaros received the B.Sc. (2000) and Ph.D. (2005) degrees in Computer Science from the University of Lancaster, UK. He is currently (full) Professor of Computer Networks, founding director of the Networked Systems Research Laboratory (netlab) and interim director of the CyberDefence lab in the School of Computing Science at the University of Glasgow. He was a visiting professor at the University of Athens, Department of Informatics and Telecommunications during 2018–19. Prof Pezaros has published widely and is leading research in computer communications, network and service management, and network resilience, currently exploring technologies such as Software-Defined Networking (SDN) and Network Function Virtualization (NFV). He has received significant funding for his research from public funding agencies (e.g., EPSRC, EC, FAA, LMS) and industry (e.g., BT, EDF, NXP). Prof Pezaros is a chartered engineer, a fellow of the BCS and the IET, and a senior member of the IEEE and the ACM.