Latency-aware Service Placement and Live Migrations in 5G and Beyond Mobile Systems

Badr Eddine Mada[1], Miloud Bagaa[1], Tarik Taleb[1,2] and Hannu Flinck[3]
[1] Dept. of Communications and Networking, School of Electrical Engineering
Aalto University, Espoo, Finland
Emails: badr.mada@aalto.fi ; miloud.bagaa@aalto.fi ; tarik.taleb@aalto.fi
[2] University of Oulu, Oulu, Finland
[3] Nokia Bell Labs, Espoo, Finland
Email: hannu.flinck@nokia-bell-labs.com

***Abstract*—5G system and beyond will build on the network slicing for offering high customizable services with different requirements that run on top of the same shared infrastructure. Each network slice, such as Ultra-Reliable Low latency Communications (URLLC) and Enhanced Mobile Broadband (eMBB), has different requirements that can be even contradicting from a slice to another. A network slice consists of a set of physical or virtual network functions (VNF/PNF) that have various capabilities and run across multiple administrative and cloud domains of different technology. A user can simultaneously request multiple services from different network slices. In this paper, we address the problem of initial placement and live migration of multiple mobile services across centralized and edge cloud by taking into account service types, network conditions and users' mobility features. As a solution to this problem, in this paper, we suggest and evaluate a solution that orchestrates the network services in a cost-efficient way, ensuring that each user could be simultaneously served by multiple slices while perceiving a high QoS and ensuring that the service level agreements (SLAs) of the consumed services are not violated.**

I. Introduction

The fifth generation of mobile network (5G) is anticipated to revolutionize the communication landscape by introducing new applications and highly customized network capabilities for vertical use cases. This change is possible due to the new 5G radio technology, as well as the adoption of Network Function Virtualization (NFV) [1] and Software Defined Networking (SDN) [2] that form the basis of the new customized network architecture. The 5G system can offer content-rich multimedia applications, ultra-low latency and high quality of service (QoS) [3] resulting in classes of latency-sensitive applications, such as Tactile Internet, Emergency medical services, and connected vehicles usable over public cellular networks.

These new verticals build their business models on top of latency-sensitive applications that depend on always-on reachability of extreme end-user experience and superior quality of service even for customers on the move. This means that these latency-sensitive services need to follow the mobility patterns of the end-users efficiently to avoid over-provisioning and unnecessary costly redundancy. Services will be deployed at the edge of the network [4] close to end-users to meet the latency and reliability requirements. A service deployed at the edge of a network has limited service area, e.g. the number of cells and tracking areas, but should be able to adapt to the changing in end-user demands following movements of the users by migrating the services to edge nodes that are closer to the actual location of the actual service usage. By leveraging the new enabling technologies, such as NFV, SDN and network slicing [5], the same infrastructure can be shared and customized to feet the needs of the new verticals (services and applications) in order to meet their very specific SLA in a scalable and adaptive way. The use of virtualization and softwarization of the network functions and the management components allows services to move from a service area to another while adapting to the mobility patterns of the end-users and the available limited resources at the edges.

The deployment of services near to their users has a significant impact on the quality of service (QoS) and the quality of experience (QoE). For instance, a thin client [6] that has most of its applications deployed at the cloud may experience service disruptions as it moves. Therefore, it is important to deploy the services near to the client and migrate the service execution place to the nearest cloud node when the end-user moves. In this paper, we address the problem of initial placement and live migrations of multiple mobile services across a centralized and edge cloud by modeling the placement and deployment of services as integer linear programming (ILP) problem. The aim is to determine where to deploy these services while taking into account the latency required by each service, the resources needed, the network status and the location of the users. We define an objective function that takes into account the allocated resources, and the QoS by finding and using the nearest clouds that satisfy the end-users needs. We have evaluated the execution times and the resources used while changing the number of end-users on each access point and the number of services requested simultaneously by each user.

The remainder of this paper is organized as follows. A problem formulation and a network model is given in Section III, the proposed solution that deploys the services requested in the close proximity of the users is described in Section IV, the evaluation results and conclusion are given in Section V and VI, respectively.

II. Related work

In this section, we briefly summarize the research works relevant to the topic targeted by the present paper. The authors in [7] have proposed an algorithm for placing virtual machines (VMs) in various data centers (DCs). Meanwhile, the solution, proposed in [8], aims at solving VNF placement problem. This solution aims to i) minimize the number of connection points in the network; ii) eliminate the ping-pong traffic iii) and reduce the flow rules on SDN switches. G. Moualla et al. [9] have proposed an iterative linear program that allows the placement of SFCs in data centers while respecting a given service level agreement (SLA) and ensuring the availability of services. To tackle failures and the unavailability of the service function chains (SFC), the authors apply SFC replication and distribute the load equally between each replica.

Besides the solutions that leverage optimization techniques, Artificial Intelligence (AI) has been also widely applied in networking in general and for resource provisioning in particular. For instance, D. Clark et al. [10] have proposed a knowledge-based approach to deal with in various networking situations. The authors have argued that network traffic should follow specified patterns, and to accomplish this, they suggest a solution that creates autonomous self-configurable networks. Deep learning use-cases for networks have been surveyed in [11]. The conclusion is that deep learning techniques are useful for understanding network behavior, and then suggesting recommendations for efficient policies and configurations to enhance the network performances. An Artificial Neural Network (ANN) has been applied to learn and build models for the virtual network (VN) based on collected network data. H. Jmila et al. [12] have proposed a solution that leverages Support Vector Regression (SVR) to predict the amount of CPU needed to handle the incoming traffic flows.

III. Problem formulation and network model

We denote by $\mathcal{G}(\mathcal{V} \cup \mathcal{U}, \mathcal{E}, \mathcal{W})$ the network infrastructure that consists of cloud (C), edge cloud (EC) ($\mathcal{V}$), and (radio) access nodes ((R)AN) $\mathcal{U}$. While $\mathcal{E}$ and $\mathcal{W}$ represent the links between $\mathcal{V} \cup \mathcal{U}$, and their characteristics including the delay and the bandwidth capacity between distinct nodes, respectively. We consider $\mathcal{W} = (\mathcal{W}^{\mathcal{B}}, \mathcal{W}^{\mathcal{L}})$, such that $\mathcal{W}^{\rho}$ and $\mathcal{W}^{\mathcal{L}}$ are the bandwidth capacity and propagation delay of the links in $\mathcal{E}$. We assume that the system supports a set of vertical networks (e.g. a network of an industrial facility), each of which has different requirements, such as Ultra-Reliable Low latency Communications (URLLC) and Enhanced Mobile Broadband (eMBB). Let Γ denote the set of the vertical network in the system, whereby each vertical network $\gamma \in \Gamma$ offers similar services. Each vertical network $\gamma \in \Gamma$ has a specific characteristic vector, denoted by Σ_γ, that represents the required end-to-end delay, reliability and bandwidth. For the sake of simplicity and without losing generality, we consider in the proposed model the end-to-end delay $\Sigma_\gamma^{\mathcal{L}}$ and the network bandwidth $\Sigma_\gamma^{\mathcal{B}}$. Moreover, we consider that we have a set of UEs that consume different services provided by vertical networks. Note that a single UE can use services from multiple verticals at the same time. Let Φ_γ denote the set of users and devices of the vertical network γ. Each user or device $\phi \in \Phi_\gamma$ is expected to generate an amount of traffic λ_ϕ^γ for each service/vertical network $\gamma \in \Gamma$.

Each cloud/edge $u \in \mathcal{V}$ is characterized by a limited storage and computation resources including CPU, RAM and DISK. Let Δ_u be a vector that shows the resources of the cloud $u \in \mathcal{V}$. δ_v denotes a vector of resources used by a VNF $v \in \Upsilon_\gamma$. $\mathcal{F}(.)$ represents a function of resource-service dependency. Formally, there is a correlation between the resources used by a VNF and the expected services offered by that VNF in terms of computation and QoS. According to the resources used by the VNF $v \in \Upsilon_\gamma$, the behavior of the VNF will be affected [13], [14]. Formally, $\mathcal{F}(v, \delta_v)$ denotes a vector of expected services, in terms of delay and QoS that can be offered by the VNF v using the resource δ_v. $\mathcal{F}(v, \delta_v)$ can be defined as follow: $\mathcal{F}(v, \delta_v) = \alpha_{v,\ell} \times \delta_v + \beta_{v,\ell}$, such that $\Psi_{\ell-1} \leq \delta_v \leq \Psi_\ell$.

IV. Proposed solution

In this subsection, we present our proposed solution that aims to cope with the service placement and user mobility. Mainly, the suggested solution is executed in epochs, such that at each epoch the users Φ would be distributed among various eNBs/gNBs. Let $\eta_\gamma(\rho)$ denote the set of UEs of an eNB/gNB $\rho \in \mathcal{U}$ that request the vertical γ. While $\eta(\rho)$ denotes the set of UEs that uses the access point ρ. Formally, $\eta(\rho) = \bigcup_{\gamma \in \Gamma} \eta_\gamma(\rho)$. In our model, a UE can be attached to only one access node at a given time, and hence $\forall \rho_1, \rho_2 \in \mathcal{U}, \rho_1 \neq \rho_2 : \eta(\rho_1) \cap \eta(\rho_2) = \emptyset$. Moreover, we denote by Φ_γ the set of UEs requesting the vertical γ. In the system, one UE $\phi \in \Phi$ can requests multiple services simultaneously. For this reason, we can have the following situation: $\exists \rho \in \mathcal{U}, \exists \gamma_1, \gamma_2 \in \Gamma, \gamma_1 \neq \gamma_2 : \eta_{\gamma_1}(\rho) \cap \eta_{\gamma_2}(\rho) \neq \emptyset$.

A. Variables definition

In what follows, we define the various decision variables. For each vertical $\gamma \in \Gamma$, we define a decision variable $\mathcal{X}_{v,j}$ that denotes if a VNF $v \in \Upsilon_\gamma$ is hosted at the edge cloud $j \in \mathcal{V}$.

$\forall \gamma \in \Gamma, \forall v \in \Upsilon_\gamma, \forall j \in \mathcal{V}$:

$$\mathcal{X}_{v,j} = \begin{cases} 1 & \text{If } v \text{ is hosted at the edge cloud } j \\ 0 & \text{Otherwise} \end{cases}$$

Also, we denote by $\mathcal{A}_{\phi,i}^\gamma$ a decision Boolean variable that shows if a user or a device $\phi \in \Phi_\gamma$ of a vertical $\gamma \in \Gamma$ uses the VNF $i \in \Upsilon_\gamma$,

$\forall \gamma \in \Gamma, \forall i \in \Upsilon_\gamma, \forall \phi \in \Phi_\gamma$:

$$\mathcal{A}_{\phi,i}^\gamma = \begin{cases} 1 & \text{If } \phi \text{ uses the VNF } i \text{ of the vertical } \gamma \\ 0 & \text{Otherwise} \end{cases}$$

Based on the observation that the users are dynamic due to their mobility and connectivity in time, some services of a vertical $\gamma \in \Gamma$ could be not needed, and hence there is no need to deploy them. This will help to reduce the overall cost of the system. In the system any VNF $v \in \Upsilon_\gamma$ of a vertical $\gamma \in \Gamma$ would not be used by a UE or device in the upcoming

epoch, it should be either not deployed or removed if it is already deployed. Formally, a VNF $\forall \upsilon \in \Upsilon_\gamma$ should not be deployed if and only if $\sum_{j \in \mathcal{V}} \mathcal{X}_{\upsilon,j} = 0$. Let $\mathcal{Z}_\upsilon$ be a Boolean variable that shows if a VNF $\upsilon \in \Upsilon_\gamma$ should be deployed or not.

$\forall \gamma \in \Gamma, \upsilon \in \Upsilon_\gamma$:

$$\mathcal{Z}_\upsilon = \begin{cases} 1 & \text{If the VNF } \upsilon \text{ is deployed} \\ 0 & \text{Otherwise} \end{cases}$$

For each vertical $\gamma \in \Gamma$ and VNF $\upsilon \in \Upsilon_\gamma$, we define a real number δ_υ that shows the vector of resources that should be used by the VNF υ. Moreover, we define two variables that show the characteristics of communication between each access point $\rho \in \mathcal{U}$ and a VNF $\upsilon \in \Upsilon_\gamma$ of the vertical $\gamma \in \Gamma$ in terms of delay and bandwidth. For this reason, we define the variable $\mathcal{Y}^{\mathcal{B}}_{\rho,\upsilon}$ that represents the communication bandwidth between the access point ρ and the VNF υ, also the variable $\mathcal{Y}^{\mathcal{L}}_{\rho,\upsilon}$ that shows the propagation delay.

B. Constraints definition

In this subsection, we define the constraints that ensure the functionality of the system.

1) Constraints related with VNFs deployment: The constraints defined in this sub-section related to the deployment of VNFs belonging to different verticals $\gamma \in \Gamma$. A VNF of a vertical should be deployed only at one location. Moreover, the VNF should be deployed if and only if it is requested by at least one user:

$$\forall \gamma \in \Gamma, \upsilon \in \Upsilon_\gamma : \sum_{j \in \mathcal{V}} \mathcal{X}_{\upsilon,j} = \mathcal{Z}_\upsilon \tag{1}$$

The following constraint ensures that each user $\phi \in \Phi_\gamma$ of a vertical $\gamma \in \Gamma$ uses a VNF $\upsilon \in \Upsilon_\gamma$:

$\forall \gamma \in \Gamma, \phi \in \Phi_\gamma$:

$$\sum_{\upsilon \in \Upsilon_\gamma} \mathcal{A}^{\gamma}_{\phi,\upsilon} = 1 \tag{2}$$

The following constraint ensures the deployment of each requested VNF $\upsilon \in \Upsilon_\gamma$:

$\forall \gamma \in \Gamma, \upsilon \in \Upsilon_\gamma, \phi \in \Phi_\gamma$:

$$\mathcal{Z}_\upsilon \geq \mathcal{A}^{\gamma}_{\phi,\upsilon} \tag{3}$$

While constraint (4) ensures that a VNF $\upsilon \in \Upsilon_\gamma$ would be deployed if and only if it is requested by a user.

$\forall \gamma \in \Gamma, \forall \upsilon \in \Upsilon_\gamma$:

$$\mathcal{Z}_\upsilon \leq \sum_{\rho \in \mathcal{U}, \phi \in \eta_\gamma(\rho)} \mathcal{A}^{\gamma}_{\phi,\upsilon} \tag{4}$$

2) Constraints related to VNFs resources: The following constraint ensures that resources are allocated to the VNF if and only if it is deployed.

$\forall \gamma \in \Gamma, \forall \upsilon \in \Upsilon_\gamma$:

$$\delta_\upsilon \leq \mathcal{Z}_\upsilon \times \mathcal{M}, \tag{5}$$

, such that $\mathcal{M}$ is a big number ($\mathcal{M} \approx \infty$).

Moreover, the following constraints ensure that the resources allocated to a VNF $\upsilon \in \Upsilon_\gamma$ meet its users' requests.

$\forall \gamma \in \Gamma, \forall \upsilon \in \Upsilon_\gamma$:

$$\delta_\upsilon \geq \sum_{\phi \in \Phi_\gamma} \mu^{\gamma}_{\phi} \times \mathcal{A}^{\gamma}_{\phi,i} \tag{6}$$

The resources allocated to the VNFs should not exceed the capacity of the host.

$$\forall j \in \mathcal{V} : \sum_{\gamma \in \Gamma, \upsilon \in \Upsilon_\gamma} \delta_\upsilon \times \mathcal{X}_{\upsilon,j} \leq \Delta_j \tag{7}$$

However, this constraint is not linear, then we need to transform it to linear constraint by adding the following constraints and variables.

$\forall j \in \mathcal{V}, \forall \gamma \in \Gamma, \upsilon \in \Upsilon_\gamma$:

$$\hat{\delta}_{\upsilon,j} = \begin{cases} \delta_\upsilon & \text{If the VNF } \upsilon \text{ is deployed at } j \ (\mathcal{X}_{\upsilon,j} = 1) \\ 0 & \text{Otherwise} \end{cases}$$

$$\forall j \in \mathcal{V} : \sum_{\gamma \in \Gamma, \upsilon \in \Upsilon_\gamma} \hat{\delta}_{\upsilon,j} \leq \Delta_j \tag{8}$$

$$\forall j \in \mathcal{V}, \forall \gamma \in \Gamma, \forall \upsilon \in \Upsilon_\gamma : \hat{\delta}_{\upsilon,j} \leq \delta_\upsilon + (1 - \mathcal{X}_{\upsilon,j}) \times \mathcal{M} \tag{9}$$

$$\forall j \in \mathcal{V}, \forall \gamma \in \Gamma, \forall \upsilon \in \Upsilon_\gamma : \delta_\upsilon \leq \hat{\delta}_{\upsilon,j} + (1 - \mathcal{X}_{\upsilon,j}) \times \mathcal{M} \tag{10}$$

$$\forall j \in \mathcal{V}, \forall \gamma \in \Gamma, \forall \upsilon \in \Upsilon_\gamma : \hat{\delta}_{\upsilon,j} \leq \mathcal{X}_{\upsilon,j} \times \mathcal{M} \tag{11}$$

3) Constraints related to the required services: In this subsection, we present the set of constraints that ensure the required services. The following constraint computes the propagation delay between the edge $\rho \in \mathcal{U}$ and the VNF $\upsilon \in \Upsilon_\gamma$.

$\forall \rho \in \mathcal{U}, \forall \gamma \in \Gamma, \forall \upsilon \in \Upsilon_\gamma$:

$$\mathcal{Y}^{\mathcal{L}}_{\rho,\upsilon} = \sum_{j \in \mathcal{V}} \mathcal{X}_{\upsilon,j} \times \mathcal{W}^{\mathcal{L}}_{\rho,j} \tag{12}$$

The following constraint ensures that the capacity of each link is not exceeded. By other words, the traffic generated and forwarded between an access node ρ and a cloud/edge j should not exceed the capacity of that link. Let $\mathcal{Y}^{\mathcal{B}}_{\rho,\upsilon}$ denote the amount of traffic generated between ρ and υ. This constraint is defined as follow:

$\forall \rho \in \mathcal{U}, \forall j \in \mathcal{V}$:

$$\mathcal{Y}^{\mathcal{B}}_{\rho,j} = \sum_{\gamma \in \Gamma, \phi \in \eta_\gamma(\rho), \upsilon \in \Upsilon_\gamma} \lambda^{\gamma}_{\phi} \times \mathcal{A}^{\gamma}_{\phi,\upsilon} \times \mathcal{X}_{\upsilon,j} \tag{13}$$

However, the constraint (13) is not linear due to the multiplication between $\mathcal{A}^{\gamma}_{\phi,\upsilon}$ and $\mathcal{X}_{\upsilon,j}$. In order to transform the constraint to be linear, we add the following variables and constraints. First, we add the following binary variable $\hat{\mathcal{A}}^{\gamma}_{\phi,\upsilon,j}$ that equals to $\mathcal{A}^{\gamma}_{\phi,\upsilon} \times \mathcal{X}_{\upsilon,j}$. We replace the constraint 13 by the following constraints:

$\forall \rho \in \mathcal{U}, \forall j \in \mathcal{V}$:

$$\mathcal{Y}^{\mathcal{B}}_{\rho,j} = \sum_{\gamma \in \Gamma, \phi \in \eta_\gamma(\rho), \upsilon \in \Upsilon_\gamma} \lambda^{\gamma}_{\phi} \times \hat{\mathcal{A}}^{\gamma}_{\phi,\upsilon,j} \tag{14}$$

$\forall \rho \in \mathcal{U}, \forall j \in \mathcal{V}$:

$$\hat{\mathcal{A}}^{\gamma}_{\phi,\upsilon,j} \geq \mathcal{A}^{\gamma}_{\phi,\upsilon} + \mathcal{X}_{\upsilon,j} - 2 \tag{15}$$

$\forall \rho \in \mathcal{U}, \forall j \in \mathcal{V}$:

$$\hat{\mathcal{A}}^{\gamma}_{\phi,\upsilon,j} \leq \mathcal{A}^{\gamma}_{\phi,\upsilon} \tag{16}$$

$\forall \rho \in \mathcal{U}, \forall j \in \mathcal{V}$:

$$\hat{\mathcal{A}}^{\gamma}_{\phi,\upsilon,j} \leq \mathcal{X}_{\upsilon,j} \tag{17}$$

The following constraint ensures that the communication between an access node ρ and a cloud/edge j is not overloaded.
$\forall \rho \in \mathcal{U}, \forall j \in \mathcal{V}$:

$$\mathcal{Y}^{\mathcal{B}}_{\rho,j} \leq \mathcal{W}^{\mathcal{B}}_{\rho,j} \tag{18}$$

Let $\Psi_{\rho,\upsilon}$ denote the transmission delay between the access node ρ and a VNF $\upsilon \in \Upsilon_{\gamma}$. Formally, $\Psi_{\rho,\upsilon}$ is defined as follow:
$\forall \rho \in \mathcal{U}, \forall \gamma \in \Gamma, \forall \upsilon \in \Upsilon_{\gamma}$:

$$\Psi_{\rho,\upsilon} = \sum_{j \in \mathcal{V}} \frac{1}{\mathcal{W}_{\rho,j}} \times \mathcal{Y}^{\mathcal{B}}_{\rho,j} \times \mathcal{X}_{\upsilon,j} \tag{19}$$

However, the constraint (19) is not linear. To transfer the constraint (19) to be linear, we need to add the following constraints and variables. First of all, we add the variable $\hat{\mathcal{Y}}^{\mathcal{B}}_{\rho,j,\upsilon}$ that should equal to $\mathcal{Y}^{\mathcal{B}}_{\rho,j} \times \mathcal{X}_{\upsilon,j}$. Then, we replace the constraint (19) by the following constraints:
$\forall \rho \in \mathcal{U}, \forall \gamma \in \Gamma, \forall \upsilon \in \Upsilon_{\gamma}$:

$$\Psi_{\rho,\upsilon} \geq \sum_{j \in \mathcal{V}} \frac{1}{\mathcal{W}_{\rho,j}} \times \hat{\mathcal{Y}}^{\mathcal{B}}_{\rho,j,\upsilon} \tag{20}$$

$\forall \rho \in \mathcal{U}, \forall \gamma \in \Gamma, \forall \upsilon \in \Upsilon_{\gamma}, \forall j \in \mathcal{V}$:

$$\hat{\mathcal{Y}}^{\mathcal{B}}_{\rho,j,\upsilon} \leq \mathcal{Y}^{\mathcal{B}}_{\rho,j} + (1 - \mathcal{X}_{\upsilon,j}) \times \mathcal{M} \tag{21}$$

$\forall \rho \in \mathcal{U}, \forall \gamma \in \Gamma, \forall \upsilon \in \Upsilon_{\gamma}, \forall j \in \mathcal{V}$:

$$\mathcal{Y}^{\mathcal{B}}_{\rho,j} \leq \hat{\mathcal{Y}}^{\mathcal{B}}_{\rho,j,\upsilon} + (1 - \mathcal{X}_{\upsilon,j}) \times \mathcal{M} \tag{22}$$

$\forall \rho \in \mathcal{U}, \forall \gamma \in \Gamma, \forall \upsilon \in \Upsilon_{\gamma}, \forall j \in \mathcal{V}$:

$$\hat{\mathcal{Y}}^{\mathcal{B}}_{\rho,j,\upsilon} \leq \mathcal{X}_{\upsilon,j} \times \mathcal{M} \tag{23}$$

The following constraint ensures that the end-to-end delay between an access node ρ and a VNF $\upsilon \in \Upsilon_{\gamma}$ does not exceed the delay required by the vertical γ.
$\forall \rho \in \mathcal{U}, \forall \gamma \in \Gamma, \forall \upsilon \in \Upsilon_{\gamma}$:

$$\underbrace{\mathcal{Y}^{\mathcal{L}}_{\rho,\upsilon}}_{(24.a)} + \underbrace{\mathcal{F}(\upsilon, \delta_{\upsilon})}_{(24.b)} + \underbrace{\Psi_{\rho,\upsilon}}_{(24.c)} \leq \Sigma^{\mathcal{L}}_{\gamma} \times \mathcal{Z}_{\upsilon} \tag{24}$$

While the equation (24.a) counts the propagation delay between the access node ρ and the VNF υ, the equation (24.b) refers to the processing time using the VNF υ if it is deployed. Meanwhile, the constraint (24.c) counts the transmission delay between ρ and υ. The constraint 24 ensures that the end-to-end delay of a deployed VNF υ is not exceeding the delay required by its vertical $\Sigma^{\mathcal{L}}_{\gamma}$.

C. Final optimization model

The objective function of the optimization aims to minimize the allocated resources to the deployed services. Thus, the optimization problem is defined as follow:

$$\textbf{min} \sum_{\gamma \in \Gamma, \upsilon \in \Upsilon_{\gamma}} \delta_{\upsilon} \tag{25}$$

s. t.

- deployment constraints : (1), (2), (3), (4)
- resources constraints : (5), (6), (8), (9) , (10) , (11)
- connectivity constraints : (12), (14), (15), (16), (17), (18), (20), (21), (22), (23), (24)

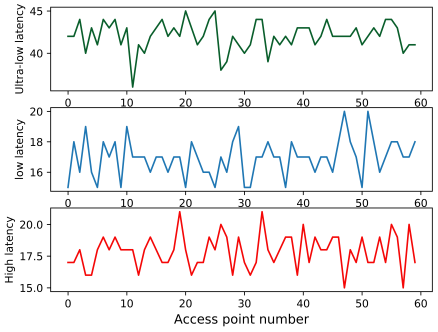
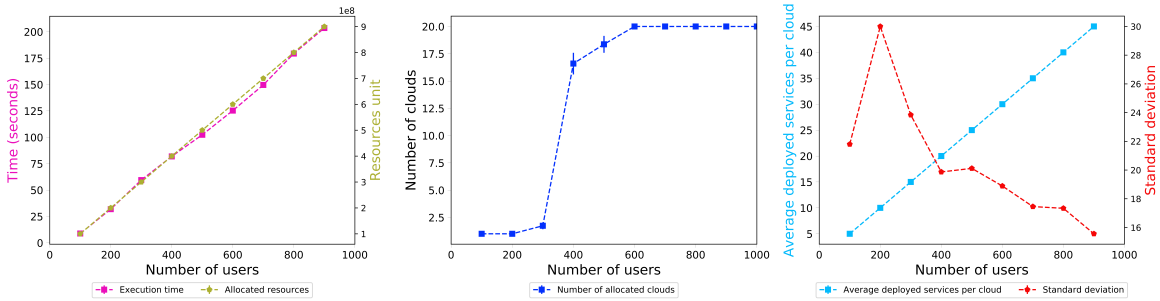


Fig. 1. Average requested service type by access point

V. Performances evaluation

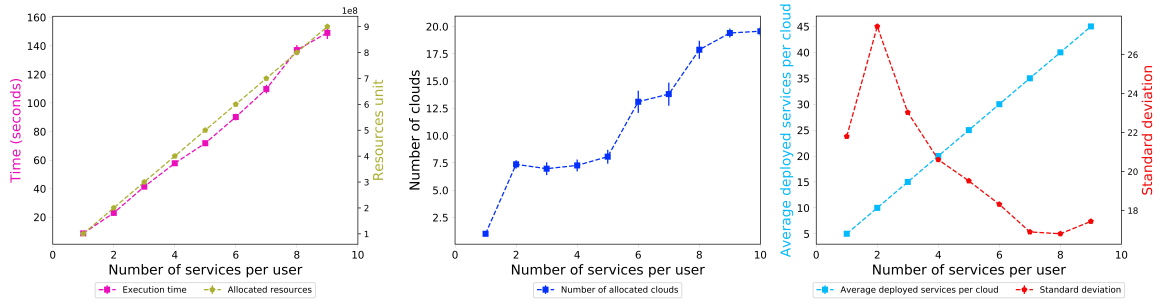
In this section, we evaluate our suggested solution in terms of: i) Time complexity defined as the needed time to execute the suggested algorithm that finds the optimal configurations; ii) Resource cost defined as the allocated resources to host different services; iii) Average deployed services per cloud that shows the average overhead created at each cloud or edge; iv) Standard deviation of services among the cloud that represents the overhead distribution of services on multiple cloud domains. In the simulations we have fixed the number of verticals, clouds and edge clouds and eNodeBs by 20, 20 and 60, respectively. Each user can simultaneously requests multiple services. According to rel 16 of 3GPP standard [15], a UE can be connected to 8 slices simultaneously. In the simulation, we have varied the number of requested services by each user from 1 to 10.

The proposed solution has been implemented and evaluated using python and Gurobi optimizer. In the simulation, each cloud and edge cloud has predefined resources in terms of CPU, RAM, and storage. The edge clouds are connected to all eNodeBs, whereby the virtual links between edge clouds and eNodeBs are characterized by the minimum latency and bandwidth. Meanwhile, the users' requests are randomly generated, whereby each of which has a service ID and requires a specified amount of resources. We have carried out two sets of experiments. In the first experiment, we have varied the number of services per user (SPU) while fixing the number of users in the system by 100. In the second experiment, we have



(a) Resources Cost and execution time. (b) Number of used clouds. (c) The Average service per cloud and its mean square error.

Fig. 2. Varying the number of users.



(a) Resources Cost and execution time (b) [Number of used clouds. (c) The Average service per cloud and its mean square error.

Fig. 3. Varying the number of services per user.

varied the number of users in the system while fixing SPU to 1. For each experiment we run 35 repetitions while changing the requested services types, the characteristics of the virtual links, and the amount of resources required by each service. All the plotted data are presented with 95% confidence interval. We conducted our experiments on a server characterized by a Dual Intel Xeon E5-2680 v3 2.5GHz, 256GB of memory and Linux Ubuntu 16.04 as operation system.

In the simulation, we have considered 20 verticals that are classified into three sub-categories based on the service characteristics they offer as depicted in Fig.1: i) High latency that represents the verticals that do not have any latency requirements. Formally, the network latency on those verticals varies from $70ms$ to $80ms$; ii) Low latency verticals that require low latency for ensuring a specified QoS. These verticals require network latency that varies between $60ms$ and $70ms$; iii) Ultra-low latency verticals that require ultra low latency communication to ensure their functionality. Formally, these network slices have network latency varying between $40ms$ and $60ms$. Additionally, we have randomly assigned the 20 verticals among the users. The latter keep moving across different access-points. Fig. 1 shows the average requested verticals per access point. While the green plot represents the ultra-low latency services, the blue plot represents the low latency vertical and the red plot represents the high latency vertical. The distribution of services over the access points is used as input for evaluating the proposed solution.

Fig. 2 and Fig. 3 depict the evaluation of the proposed solution while varying the number of users and SPU, respectively. In Fig.2, we have varied the number of users from 100 to 900 while fixing SPU by 1. Meanwhile, in Fig. 3, we have varied SPU from 1 to 10 while fixing the number of users by 100.

Fig. 2(a) and Fig. 3(a) show the performances of the proposed solution in terms of time complexity and resource cost while varying the number of users and SPU, respectively. In the figures, the left Y-axis serves for the execution time while the right Y-axis represents the resource cost devoted for serving different running verticals. The first observation that we can draw from Fig. 2(a) is the number of users has a negative impact on the complexity of the suggested solution. In fact, increasing the number of users raises the number of variables and constraints used in the optimization problem defined by the equations (1 - 25). We also observe that the execution time increases linearly with the number of users. From the figure, the execution time varies from $7sec$ to $200sec$ when varying the number of users from 100 to 900 users. Meanwhile, in 3(a), we observe that the SPU has also a negative impact on the execution time. For the same reason, increasing the SPU increases the number of variables and constraints used in the optimization problem defined by the equations (1 - 25). We also note that the execution time increases linearly with SPU. In the figure, we observe that the execution time varies from $6sec$ to $150sec$ when varying the SPU from 2 to 9, respectively. From Fig. 2(a) and Fig.

3(a), we also observe that the resource cost increases linearly with SPU and with the number of users. In fact, increasing the SPU or the number of users raises the number of generated requests, and hence more resources are needed to handle these new requests.

Fig 2(b) and Fig.3(b) show the number of used clouds hosting the running services that should handle the traffic generated by users. From Fig. 2(b), we observe that when the number of users exceeds 300, the proposed solution uses all the available clouds and edges. In that figure, we also observe that the number of used clouds dramatically increases from 2 to 17 when the number of users raises from 100 to 200. This can be explained as follows: Increasing the number of users in the network leads to increase the number of requests, hence the proposed solution uses more clouds and edges to ensure the required QoS. Meanwhile, in Fig.3(b), we observe that all the clouds are used when SPU exceeds 5. We also observe that the number of used clouds dramatically increases from 6 to 18 when SPU varies from 2 to 3. In fact, increasing the number of SPU leads to increase the probability of selecting verticals with various delay requirements, and hence more clouds or edge clouds should be used to fulfill the new requirements.

Fig. 2(c) and Fig. 3(c) show the performances of the proposed solution in terms of the average of deployed services and overhead distribution of services while varying the number of users and SPU, respectively. From Fig. 2(c), we observe that the average of deployed services increases linearly with the number of users in the system. This is since the more users there are, the more requests need to be handled, and hence more services should be deployed in the clouds and edges clouds. We also observe that the number of users has a positive impact on the service distributions among the clouds and edges clouds. From Fig. 2(b), we observe that increasing the number of users automatically increases the clouds utilization. Therefore, more services would be distributed among the clouds and edges, which leads to enhance service distribution. On another hand, Fig. 3(c) shows that the impact of SPU on the average of deployed services and the overhead distribution of services, respectively. The first observation that we can draw from this figure is that the average of deployed services increases linearly with the SPU value. From Fig. 3(c), we observe that increasing the number of SPU results in increasing the number of used clouds, and hence the average services per cloud/edge. Also, we observe that amplifying the SPU has a positive impact on the distribution of services. Raising the number of SPU leads to increase the number of used clouds or edges, and hence increases the likelihood for disturbing the services among the clouds/edges.

VI. Conclusion

We have presented a mathematical model that ensures the life cycle management of mobile services across multiple cloud domains. The proposed model ensures the initial placement, as well as the service migration across multiple clouds. The suggested solution aims to reduce the cost while ensuring the KPIs of deployed slices. We have evaluated the proposed model using simulation, and the obtained results demonstrate the efficiency of the proposed solution.

Acknowledgment

This work is partially supported by 5G!Force project, the European Union's Horizon 2020 research and innovation program under the H2020 MonB5G project with grant agreement No. 871780, and the Academy of Finland's Flagshipprogramme 6Genesis under grant agreement no. 318927.

References

[1] "Network functions virtualisation (nfv); architectural framework, gs nfv 002, etsi nfv."

[2] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, Jan 2015.

[3] J. Rodriguez, *The 5G Internet*. Wiley, 2014. [Online]. Available: https://ieeexplore.ieee.org/document/8043686

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

[5] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing and softwarization: A survey on principles, enabling technologies, and solutions," *IEEE Communications Surveys Tutorials*, vol. 20, no. 3, pp. 2429–2453, thirdquarter 2018.

[6] M. Arumaithurai, J. Seedorf, M. Dusi, E. Monticelli, and R. Lo Cigno, "Quality-of-experience driven acceleration of thin client connections," in *2013 IEEE 12th International Symposium on Network Computing and Applications*, Aug 2013, pp. 203–210.

[7] M. Alicherry and T. V. Lakshman, "Network aware resource allocation in distributed clouds," in *2012 Proceedings IEEE INFOCOM*, March 2012, pp. 963–971.

[8] A. Hirwe and K. Kataoka, "Lightchain: A lightweight optimisation of vnf placement for service chaining in nfv," in *2016 IEEE NetSoft Conference and Workshops (NetSoft)*, June 2016, pp. 33–37.

[9] G. Moualla, T. Turletti, and D. Saucez, "An availability-aware sfc placement algorithm for fat-tree data centers," in *2018 IEEE 7th International Conference on Cloud Networking (CloudNet)*, Oct 2018, pp. 1–4.

[10] D. D. Clark, C. Partridge, J. C. Ramming, and J. T. Wroclawski, "A knowledge plane for the internet," in *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, ser. SIGCOMM '03. New York, NY, USA: ACM, 2003, pp. 3–10. [Online]. Available: http://doi.acm.org/10.1145/863955.863957

[11] A. Mestres, A. Rodriguez-Natal, J. Carner, P. Barlet-Ros, E. Alarcón, M. Solé, V. Muntés-Mulero, D. Meyer, S. Barkai, M. J. Hibbett, G. Estrada, K. Ma'ruf, F. Coras, V. Ermagan, H. Latapie, C. Cassar, J. Evans, F. Maino, J. Walrand, and A. Cabellos, "Knowledge-defined networking," *SIGCOMM Comput. Commun. Rev.*, vol. 47, no. 3, pp. 2–10, Sep. 2017. [Online]. Available: http://doi.acm.org/10.1145/3138808.3138810

[12] H. Jmila, M. Ibn Khedher, and M. El Yacoubi, "Estimating VNF resource requirements using machine learning techniques," in *ICONIP 2017 : 24th International Conference on Neural Information Processing*. Guangzhou, China: Springer, Nov. 2017, pp. 883 – 892. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01682996

[13] G. M. Amdahl, "Validity of the single processor approach to achieving large scale computing capabilities," in *Proceedings of the April 18-20, 1967, Spring Joint Computer Conference*, ser. AFIPS '67 (Spring). New York, NY, USA: ACM, 1967, pp. 483–485. [Online]. Available: http://doi.acm.org/10.1145/1465482.1465560

[14] A. D. R. J. R. McCool, Michael D., *Structured parallel programming: patterns for efficient computation*, 1st ed. Elsevier, 2012.

[15] "https://www.3gpp.org/release-16."