Design and Evaluation of Reconfigurable SDN LEO Constellations

Arled Papa*, Tomaso de Cola†, Petra Vizarreta*, Mu He*, Carmen Mas-Machuca*, Wolfgang Kellerer*
*Chair of Communication Networks, Technical University of Munich
†DLR- German Aerospace Center, Institute for Communications and Navigation
*arled.papa@tum.de, †tomaso.decola@dlr.de, *petra.vizarreta@lkn.ei.tum.de, *mu.he@tum.de, *cmas@tum.de, *wolfgang.kellerer@tum.de

***Abstract*—In the context of the 5G ecosystem, the integration between the terrestrial and satellite networks is envisioned as a potential approach to further enhance the network capabilities. In light of this integration, the satellite community is revisiting its role in the next generation 5G networks. Emerging technologies such as Software-Defined Networking (SDN) which rely on programmable and reconfigurable concepts, are foreseen to play a major role in this regard. Therefore, an interesting research topic is the introduction of management architecture solutions for future satellite networks driven by means of SDN. This anticipates the separation of the data layer from the control layer of the traditional satellite networks, where the control logic is placed on programmable SDN controllers within traditional satellite devices. While a centralized control layer promises delay reductions, it introduces additional overheads due to reconfiguration and migration costs. In this paper, we propose a method to quantify the overhead imposed on the network by the aforementioned parameters while investigating the use-case scenario of an SDN-enabled satellite space segment. We make use of an optimal controller placement and satellite-to-controller assignment which minimizes the average flow setup time with respect to varying traffic demands. Furthermore, we provide insights on the network performance with respect to the migration and reconfiguration cost for our proposed SDN-enabled architecture. Finally, we compare our proposed space segment SDN-enabled architecture with alternative solutions in the state-of-the-art given the aforementioned performance metrics.**

***Keywords*—SDN, LEO constellation, Reconfigurable networks, Programmability, Flexibility.**

I. Introduction

The next generation 5G networks are envisioned to accommodate the increased traffic requirements and heterogeneity of future applications. In light of providing this level of flexibility, programmable and reconfigurable approaches such as Software-Defined Networking (SDN) have been identified as promising solutions.

The introduction of SDN triggers novel concepts such as Network Slicing (NS) and Mobile Edge Computing (MEC), which in turn can further enhance the network efficiency. The former is anticipated to play a major role in the 5G ecosystem by allowing the deployment of multiple virtual networks with distinct Quality of Service (QoS) requirements, in an isolated fashion on top of a common physical infrastructure. Thus, promising cost reduction and moreover supporting the required level of flexibility in the network [1]. The latter, benefits from the centralization logic of SDN and suggests the enhancement of the mobile edge i.e., Radio Access Network (RAN) with computing capabilities. Hence, potential delays can be further reduced since the connection to the core network can be omitted or diminished [2].

In the context of the 5G ecosystem, Satellite Communication (SatCom) can be exploited to complement 5G terrestrial networks. This integration has been observed as a promising approach to further improve the delivery of communication services [3]. For instance, given the high coverage and relatively low delays that Low Earth Orbit (LEO) mega-constellation can provide, a potential backhauling solution can be initiated [4]. Thus, satellite networks can support terrestrial networks to fulfill the requirements of high data rates and ubiquity, including offloading or balancing the traffic in dense populated areas [5].

Although there is vast of ongoing research on the introduction of SDN-aware solutions on the satellite domain [6], [7], there still remain interesting open research topics which lie in the tasks of system management, efficient resource allocation and network reconfiguration by means of SDN [8], [9]. Initial works mostly focus on the satellite ground segment [10], [11], whereas the satellite space segment has not received yet so much attention.

According to the SDN paradigm, the data layer of the network is separated from the control layer. The control logic is centralized in an entity referred to as SDN controller. It is the controller's task to update the forwarding rules of the data layer devices (i.e., SDN switches) when appropriate. The time required for the installation of a forwarding rule is known in the literature as the **flow setup time**. To perform efficiently and with respect to delay-critical operations this parameter should be retained as small as possible. Given the fact that the flow setup time is positively correlated to the number of data layer devices that a control has to manage, in large-scale networks such as LEO constellations, where the distance between satellites is relatively large, the resulting latency is high if the control layer contains only a single controller [12].

Consequently, a distributed control layer becomes a must. The introduction of multiple controllers in the control layer is not an unknown notion in the terrestrial networks [13], [14]. However, the problem then arises where and how many controllers should be placed on the control layer, which is referred to as the Controller Placement Problem (CPP). Given, traffic fluctuations, the position of controllers and switch-to-

controller assignment may change over time. This change is known as the controller migration and switch-to-controller reassignment, respectively. The total overhead imposed on the network due to controller and switch-to-controller reassignment is referred to in this paper as the **migration cost**.

Considering the dynamic nature of satellite networks such as LEO constellations, topology changes may occur. This results in altering communication paths previously selected for various source destination pairs. Hence, in an SDN-enabled architecture, the SDN controller is responsible for tracing these changes and hereafter modify the flow tables of the respective satellite SDN switches to adapt to the new flow paths. To achieve this, several reconfiguration messages need to be exchanged between the SDN controller and satellite SDN switches, which in turn imposes additional costs on the network. We refer to this cost type in this paper as **reconfiguration cost**.

In this paper we evaluate the migration and reconfiguration cost in a LEO constellation. To achieve this, we extend our previous work [15] and provide a mathematical framework to quantify such costs. In more details, the contributions of this paper are highlighted as follows:

- Given the SDN-enabled LEO constellation architecture and dynamic SDN controller placement problem proposed in our previous work [15], which focuses on the investigation of the impact of the traffic variations on the average flow setup time, we extend our framework by introducing the notion of migration and reconfiguration costs.
- We provide a mathematical framework to quantify the additional costs in the network and make use of our model to evaluate and provide insights of our system with respect to migration and reconfiguration costs.
- We compare the proposed SDN-enabled architecture with existing SDN-enabled satellite architectures in the literature with respect to the average flow setup time, average flow reconfiguration cost and migration cost.

The rest of the paper is structured as follows. We briefly outline the current SDN-enabled solutions both for terrestrial and satellite networks in section II. Section III, introduces our envisioned architecture for the SDN-enabled LEO constellation and elaborates the method for the traffic modeling. Moreover, it emphasizes the problem formulation which minimizes the average flow setup time and introduces the metrics of migration and reconfiguration costs. In Section IV, the results and comparisons with alternative architectures with respect to the aforementioned performance metrics are provided. The paper is finally concluded with Section V, where the main findings of our work are highlighted.

II. Background

In the perspective of next generation 5G networks, emerging technologies envisioned for terrestrial networks such as Network Slicing (NS), Mobile Edge Computing (MEC) and Software-Defined Networking (SDN) are foreseen as potential solutions to be adopted from the satellite community and bridge the connection to the terrestrial networks [10], [16]. An overview of architectures and challenges regarding the satellite integration in 5G are reflected in [17] and [18], where SDN is paving the way towards programmable and reconfigurable next generation 5G networks. Therefore, both academia and industry are investigating on solutions which mainly rely on the SDN paradigm. From the architectural viewpoint, the introduction of SDN in the network implies the separation of the control layer from the data layer. This results in a functional split, where the control logic of the network is centralized to entities referred to as SDN controllers. From the mathematical perspective, the main contributions rely on investigating the SDN controller placement with respect to performance metrics such as reliability, latency, load balancing. In this section, we briefly review the existing architectures, tools and applications of the SDN paradigm both in the terrestrial and satellite networks.

A. SDN-enabled Terrestrial Networks

In the terrestrial networks the main research question falls into the category of the SDN controller placement problem. This results in determining the position of the SDN controllers in the network, as well as defining the switch-to-controller assignment. There exist multiple studies in the literature for the SDN controller placement evaluating various performance metrics. For instance, the minimization of controller-to-switch latency or load balancing [19], [20], control layer reliability [21] or dynamic controller placement based on flow dynamics [22], [23].

B. SDN-enabled Satellite Networks

While the potential of SDN is well investigated in terrestrial networks, in satellite networks the notion of SDN is relatively new. Recently, constellations such as SpaceX (Starlink), Telesat, and OneWeb have been introduced and a first analysis of these constellations is available at [24]. Specifically for SpaceX, routing solutions have been introduced in [25], [26]. In the aforementioned approaches the routers are in charge of calculating the routing paths. However, in SDN-like solutions, only the SDN controller is responsible for the routing decisions. Two main proposals exist in the literature with respect to the location of the SDN control layer in a satellite network. The first approach considers the SDN control layer on the ground segment of the satellite network, whereas the second approach envisions the control layer on the satellite space segment.

Static SDN-enabled satellite networks: OpenSAN [27] introduces one of the first SDN-enabled satellite architectures, which proposes the incorporation of SDN controllers on Geosynchronous Equatorial Orbit (GEO) satellites. Following the same logic, the authors in [28] are the first to provide insights on the site-diversity use-case in a prototype satellite environment. Alternatively, in [29], the multi-path Transmission Control Protocol (TCP) approach is proposed by leveraging from an SDN-enabled LEO constellation. The authors show the improvements on the throughput by utilizing SDN. In this approach, the SDN controllers are placed on the satellite ground segment. However, all the aforementioned

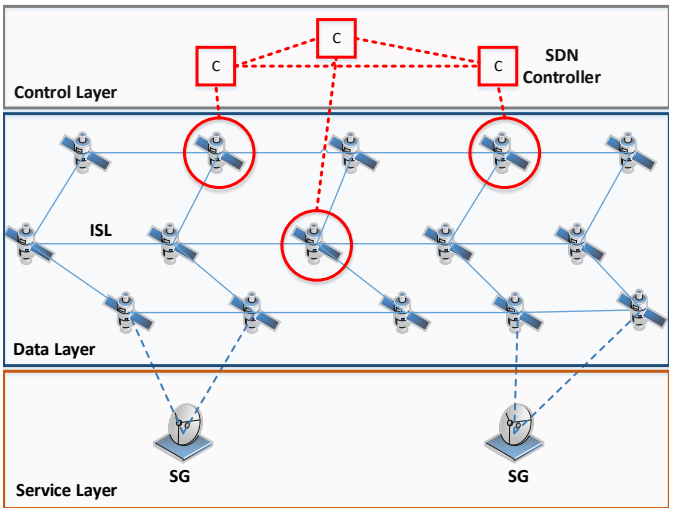


Figure 1: Proposed SDN-enabled LEO constellation architecture.

works either consider an architectural concept or a prototype network and do not take into account network dynamics such as traffic fluctuations or topology changes. To fill this void, in our previous work [15] we leveraged the SDN paradigm and introduced an SDN-enabled LEO constellation, where we investigated the dynamic SDN controller placement based on varying traffic demands.

Dynamic SDN-enabled satellite networks: While considering a dynamic network such a LEO constellation, topology changes influence drastically the network performance. Consequently, path reconfigurations become crucial such that the connections are sustained. In [30], an SDN-aware routing solution was proposed, to overcome the issues of topology changes and it is shown to outperform alternative algorithms in the state-of-art. Following that logic, it has been shown that SDN can alleviate the handover process for LEO constellations and preserve the required Quality of Experience (QoE) [31]. However, this level of flexibility comes with the cost of multiple reconfiguration messages that have to traverse the control layer, in order to reach the designated satellite SDN switches, whose flow tables need to be adapted. In order to support this operation, appropriate interfaces and protocols have to be developed to allow for such a communication between the control and data layer. Although OpenFlow [32], which is one of the mainly exploited Southbound protocols in the terrestrial networks already is a potential candidate, specific adaptations including messages which account for topology changes in dynamic environments such as LEO constellations have to be incorporated [33]. The overhead introduced by the aforementioned messages can play a major role in the network performance, yet it has not been evaluated. Therefore, in this paper we propose a method to quantify this overhead and be able to provide insights on the network performance.

III. System Model

In this section we present our SDN-enabled LEO constellation architecture. We further describe the system model and

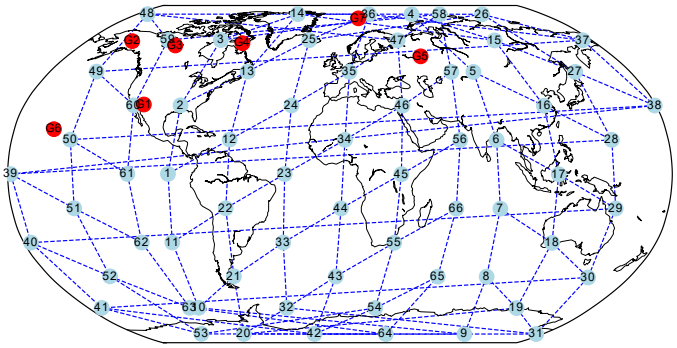


Figure 2: Iridium constellation illustrated for one network snapshot r. The satellite nodes are shown with blue circles, whereas the satellite gateways are denoted by red circles.

elaborate on the traffic modeling. Sec. III-B introduces the data layer modeling, whereas Sec. III-C describes the control layer modeling, respectively. Within the control layer, the notion of flow setup time, reconfiguration and migration cost are defined. Finally, our optimization problem is presented in details. The overall system variables and parameters are detailed in Table II.

A. SDN-enabled LEO constellation architecture

The SDN-enabled LEO constellation architecture is shown in Fig. 1. In our proposed architecture, we envision a system of three layers, namely data layer, control layer and service layer. In this work we focus our analysis on the Iridium constellation, which provides an initial network model for our simulation. We construct the LEO constellation network according to data provided in [34], where 66 satellites are distributed into 6 orbital planes and we use Systems Tool Kit (STK) [35], a software that emulates the satellite movements to monitor the connections among the satellites. The service layer consists of 7 Satellite Gateways (SG) which enable the connection to the backbone network. The position of these gateways is predefined from the existing Iridium constellation and are marked with red dots in Fig. 2. In our system, the satellites serve normally as SDN switches, therefore the data layer consists of all the satellites of the constellation. Similar to the SDN concept, the control layer is centralized, where SDN controllers are placed on the satellites depending on the optimization problem explained in Sec. III-D. This implies that several satellites can have both SDN switch and SDN controller functionalities at the same time.

The network topology is defined as a graph $G = (V, E)$. The vertices V represent the network nodes (i.e., satellites and SG), whereas the edges E denote the communication links. According to our Iridium constellation created with STK, we notice that each satellite establishes 2 stable inter-satellite links (ISLs) with satellites on the same orbit and 2 stable ISLs with the satellites on the neighbor orbits. However, the satellites of the first and sixth orbit (i.e., counter rotating orbital planes) while they can establish 2 stable ISLs with satellites on the same orbit, they contain only 1 ISL with satellites on the neighbor orbit as shown in Fig. 1. Unlike

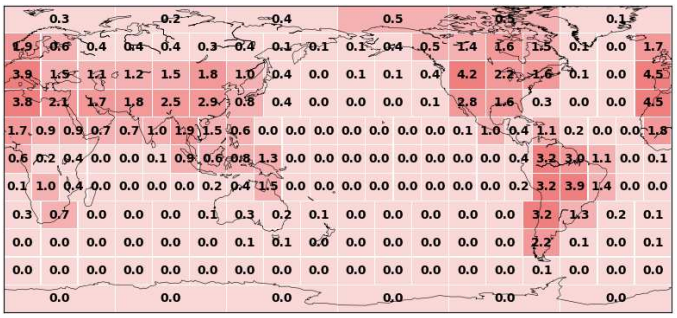


Figure 3: Traffic distribution of regions in terms of new flows per second for τ_i 10 *AM* GMT.

the ISLs between the satellites of the constellation which are stable during the whole time, the communication links between the satellites and SG are instable and hereby change over time. Therefore, mobility is introduced in the system. In order to cater for this dynamic topology feature, in our model we introduce the notion of snapshots. A snapshot r captures the network topology at a time instance. We compute for every hour all the network topologies which result due to satellite movements. This allows us to obtain the set of snapshots R for the examined hour. We split each hour into 60 snapshots, which means we check every single minute to cover all the network changes irrespective of the satellite speed. A snapshot of the network topology is illustrated in Fig. 2.

As aforementioned, we consider each satellite $s \in S \subseteq V$ capable of acting as an SDN switch and SDN controller depending on the controller placement algorithm output. We envision a distributed control layer, where K satellites of the constellation carry SDN controller functionalities. For the communication between the satellites selected as SDN controllers and the normal satellite SDN switches the existing network links (i.e., in-band control) are assumed. Further, the shortest path algorithm is used to realize the routing in the network. The forwarding latency between two network nodes $s, k \in V$ at a given snapshot $r \in R$ is denoted by $d_{r,s,k}$.

B. Data plane traffic modeling: Spatial & temporal variations

The data layer traffic modeling is based on [36]. This implies the division of the earth into $15^\circ \times 15^\circ$ square regions. As a result, a set I of 288 square regions are created. For each square region $i \in I$, the corresponding traffic requirement density w_i is calculated based on the forecast of voice traffic for 2005. Additionally, to account for temporal traffic variations, a user hourly activity denoted by τ_i is introduced for each square region i according to [37]. While identifying the satellites flying over each square region, a mapping between satellites and square regions $i \in I$ is realized. We use STK [35], a software that emulates the satellite movements, in order to divide the earth into square regions. In that regard, we make use of STK's special feature **grid coverage**, which as a result provides a set I of 192 square regions. The rational behind this difference is the fact that STK separates the earth into equal square regions in terms of surface and not coordinates. In order to account for this difference with the theoretical model, we adjust our model to map 288 regions of the theoretical model to 192 regions, based on the regions' coordinates. Each square region $i \in I$ communicates with the satellite gateways using the LEO satellites of the Iridium constellation. Therefore, the traffic profile contains satellites as sources and satellite gateways as destinations. To adapt our model to today's internet traffic requirement, we use the notion of region square requirement density with the assumption that the IP traffic density requirement of each region i is proportional to the active users within a region square.

Global Consumer Internet Traffic		
Service type	Traffic 2016 (PB per Month)	Service percentage
File sharing	6013	29.5%
Internet Video	10423	50.9%
Web,email,data	3863	18.8%
VoIP	147	0.8%
T_A	20446	100%

Table I: Consumer Internet Traffic, 2016 by sub-segment [38]

Input Parameters	
S	Set of satellites
V	Set of nodes (satellites, gateways)
E	Set of links where $E \subseteq V \times V$
C	Set of controllers where $C \subseteq S$
F	Set of the flows in the network, where f[src] and f[dst] are the source and destination for $f \in F$
R	Set of the snapshots of the network
$k_{s,f}$	Constant variable indicating whether satellite $s \in S$ is the initiator of the flow $f \in F$
$p_{f,r}$	Set of nodes V which are part of the flow path from source to destination of $f \in F$, $r \in R$
K	Number of controllers to be placed
$d_{r,s,k}$	Forwarding latency from satellite s to satellite k at snapshot r, where $s,k \in S$
List of variables	
y_c	Binary variable notating if a controller is placed on $c \in C$
$x_{s,c}$	Binary variable notating if a node $s \in S$ is assigned to $c \in C$
$z_{c,s,k}$	Binary variable notating if two consecutive nodes $s, k \in S$ are assigned to the same controller $c \in C$

Table II: System parameters and variables

The IP traffic rate generated from each region $i \in I$ is calculated through Equation (1), where T_A denotes the overall IP traffic requirement based on CISCO traffic demands for 2016 [38]. For each region i, the spatial traffic requirement is determined by the ratio between the IP traffic rate of the square region itself and the overall regions' IP traffic rate, multiplied with T_A. Table I portrays the composition of the overall Internet traffic T_A provided by [38] for different service types in PB per month. Each service type is accompanied with the user preference in %.

Further, a multiplication with the temporal traffic requirement of square region τ_i, defines the total IP traffic rate of square region λ_i at a given time instance. For our evaluation, we assume a uniform distribution of the traffic from each square region $i \in I$ to any of the satellite gateways. Hence,

the IP traffic rate of region i denoted by λ_i, to any destination is $\lambda_i/7$ and it is expressed in *Mbps*.

$$\lambda_i = \frac{w_i}{\sum_{k=1}^{k=192} w_k} \cdot T_A \cdot \tau_i. \quad (1)$$

To stay compliant with the SDN paradigm, the IP traffic rate is adjusted to flows per second. Consequently, the data rate is divided by the average flow size, where the average flow size is based on the flow characteristics of the most relevant services offered by the system as shown in Table I. However, only a portion of this traffic invokes the controller. Hence, a variable η is set to 10% to account for this ratio. We base the selection of η on [39], where an evaluation of SDN controller to switch communication control overhead is presented for various number of SDN controllers in a distributed control plane. Eventually, the traffic model for each square region $i \in I$ is expressed in number of new flows per second and is used to compose the flow profile F. The corresponding model is illustrated in Fig. 3, where for each square region $i \in I$ the flow profile F contains the satellite flying over the region at the time as a source and a satellite gateway as a destination.

C. Control layer traffic modeling

Due to the separation of the data and control layers implied in the SDN architecture, the implementation of a model to distinguish the traffic among the two layers becomes mandatory. Moreover, a high emphasis should be put on providing solutions to facilitate the communication among the two layers. To this end, the necessity of Southbound Application Programmable Interfaces (API), similar to OpenFlow [32] are a must. In this work, we focus mainly on three required control traffic messages which are exchanged in the control layer, namely flow configuration messages, flow reconfiguration messages and migration messages. To obtain a better understanding of the aforementioned messages, in the following we elaborate more on the occurrences of such messages and their impact on the network performance.

1) Flow configuration messages: Let's consider a distributed SDN control layer as shown in Fig. 4. Upon a new flow arrival at satellite s_1 of the data layer, which does not contain forwarding information about the specific flow, a request to the respective controller c_1 is initiated. This request is referred to as the initial flow setup request. Consequently a flow configuration message invokes the controller c_1. It is then the controller's task to calculate the flow path and hereafter establish the appropriate flow rules on the satellite SDN switches controlled by c_1, in this case satellite s_1 and s_3.

In a distributed control layer, it is common that a flow path contains data layer switches (i.e., satellite SDN switches), which belong to different controller domains (i.e., are controlled by different controllers). In such a scenario, an intermediate flow setup request is initiated. Similar to Fig. 4, satellite s_5 triggers such a request since it is controlled by controller c_2. Consequently, another flow configuration message invokes the control layer. Similarly to the previous case, controller c_2 establishes the rules for the flow tables of satellite s_5 and s_6.

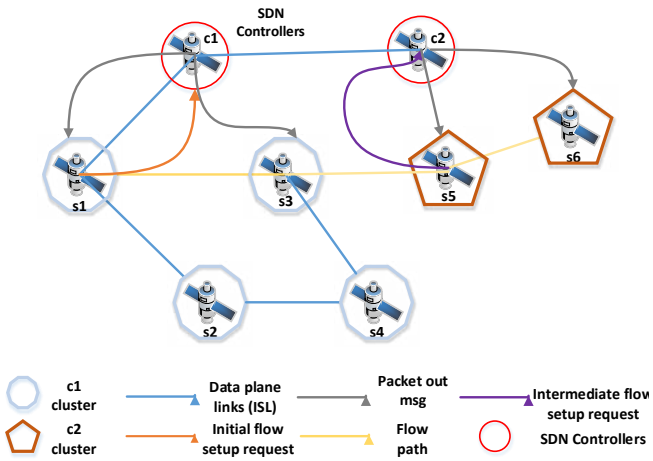


Figure 4: Illustration of the flow setup process upon a new flow arrival for a multi-domain system.

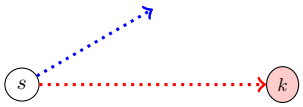
Once the flow reaches the destination, the flow setup process terminates. The difference between the time satellite s_1 sent the first flow configuration message to controller c_1 and the time satellite s_6 delivers the flow packet to its destination, determines the flow setup time.

We calculate the flow setup time in our system based on [23]. More specifically, the flow setup time includes the round-trip forwarding latency of the initial flow configuration message request and intermediate flow configuration message request if any, plus the forwarding latency between the source and the destination. The total flow setup time in the system for one hour is denoted by A_f. The calculation is expressed in (2), where the first term of the equation provides the forwarding latency between the source and destination, whereas the second term accounts for the forwarding latency acquired due to flow setup and intermediate flow setup requests for each flow f of the flow profile F at each snapshot r from the snapshot set R.

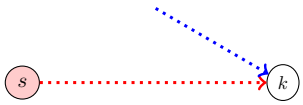
$$A_f = \sum_{f \in F} \sum_{r \in R} d_{r,f[src],f[dst]} + 2 \cdot d_{r,s,c} \cdot \sum_{f \in F} \sum_{r \in R} \sum_{c \in C} \sum_{s \in p_{f,r}} [k_{s,f} \cdot x_{s,c} + x_{s,c} - z_{c,s,k}] \quad (2)$$

2) Flow reconfiguration messages: As mentioned, given the dynamic nature of satellite networks (i.e., LEO constellations), keeping track of topology changes becomes vital since previous calculated flow paths might not be optimal due to the movement of the satellites. In more details, for a specific flow in our system the shortest path algorithm is used to calculate the flow path. Upon a topology change, the shortest path algorithm is then applied to identify a new shortest flow path. Consequently, the controllers of the control layer are responsible to re-configure the satellite SDN switches of the data layer to adapt to such changes. In order to achieve this, several reconfiguration messages need to be exchanged in the control layer. In the following, we introduce the notion of reconfiguration messages and provide

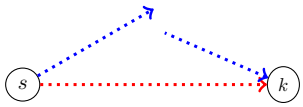
(a) satellites s and k have been removed from path j



(b) satellite k has been removed from path j



(c) satellite s has been removed from path j



(d) link (s,k) not part of path j

Figure 5: Flow_mod message calculation case a) - Link (s,k) part of path i.

(a) satellites s and k have been added

(b) satellite k has been added

(c) satellite s has been added

(d) satellites s and k have been part of the previous path

Figure 6: Flow_mod message calculation case b) - Link (s,k) not part of path i but part of path j.

a mathematical framework to quantify the amount and the satellites SDN switches, whose flow tables need to be adapted by the respective controllers.

As elaborated in Sec. III-B, a flow profile F is created for each region square $i \in I$. To account for the mobility we introduce the notion of snapshots R, which accounts for all the topology changes in the network. Further, we calculate for each flow $f \in F$ the flow paths of each snapshot $r \in R$. We then group all the paths for each flow $f \in F$ and construct the matrix PF, whose rows are binary entries denoting 1 if the satellite is part of the flow path and 0 otherwise, whereas the columns are the satellites of the constellation $s := (1, 2, ...m)$. The resulting matrix PF is illustrated in (3).

$$\mathbf{PF} = \begin{array}{c} \\ p_1 \\ p_2 \\ p_3 \\ \vdots \\ p_{|R|} \end{array} \begin{array}{c} \begin{array}{ccccc} s_1 & s_2 & s_3 & \cdots & s_m \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & \cdots & 0 \\ 1 & 1 & 1 & \cdots & 1 \\ 0 & 0 & 1 & \cdots & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & \cdots & 0 \end{bmatrix} \end{array} \tag{3}$$

In order to calculate the number of reconfiguration messages needed to be exchanged in the control layer, we apply the XOR operation among two consecutive rows i, j of the matrix PF. This results in a vector t whose entries are again binary, 1 indicating that a satellite has been added or removed in the new flow path or 0 otherwise. This operation is then performed for all the consecutive pairs of the matrix PF for all the flows $f \in F$.

In the following to ease our analysis, we will refer to two consecutive rows $i, j \in PF$ as previous flow path and new flow path, respectively. We introduce all the possible scenarios of the occurrence of reconfiguration messages by considering two main categories. The first considers the case when a previous link between two satellites (s,k) has been removed from the new flow path calculation, whereas the second case involves a new link between satellite (s,k) to be added in the new flow path calculation. Within the main categories there exist different sub-cases which are illustrated in Fig. 5 and Fig. 6 respectively. The respective sub-cases are described as follows:

5.a. Link (s,k) has been removed, both satellites s, k are not anymore part of new flow path j.

5.b. Link (s,k) has been removed, satellite k is not anymore part of new flow path j.

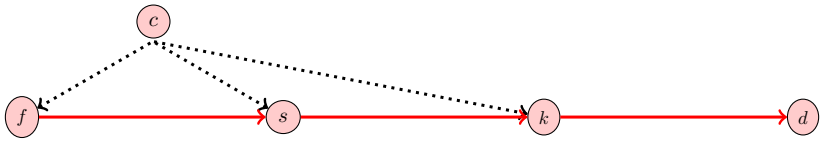
5.c. Link (s,k) has been removed, satellite s is not anymore part of new flow path j.

5.d. Link (s,k) has been removed, satellites s, k are both part of new flow path j, however, they do not share anymore a link.

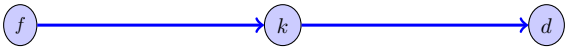
Algorithm. 1 describes the aforementioned scenarios. To ease the understanding of Algorithm 1 let us take an example. For lines 3-13, a link between satellites (s,k) exists in the previous flow path i. Given that the XOR operation between two rows $i, j \in PF$ records if satellites have changed their state (i.e., added or removed in the new flow path j), if t_s=1, means that satellite s has been removed from the path. Differently, in the second part (i.e., lines 14-24), no link between satellites (s,k) exists in the previous flow path i. Therefore, the XOR operation between two rows $i, j \in PF$, means that if t_s=1, satellite s has been added to the new flow path j. Considering this logic, we detail the description of the sub-cases as follows:

If link (s,k) was part of the previous path i, 4 sub-cases are identified. If both satellites are removed from the new path j (i.e., sub-case 5a), then both of them require a reconfiguration message from the controller, unless satellite s is not the source satellite (i.e., the first satellite of the flow path).

Furthermore, if satellite k has been removed from the



(a) Flow path from satellite source f to destination d before the re-configuration.



(b) Flow path from satellite source f to destination d after the re-configuration.

Figure 7: Re-configuration message exchange due to flow path changes.

Algorithm 1 Satellites to receive reconfiguration messages

for $s := 1, 2, 3, ..., N$ **do**
  for $k := 1, 2, 3, ..., N$ **do**
    if link (s, k) in path i **then**
      if $t_s = 1 \wedge t_k = 1$ **then**
        $\lambda_{r,f}.insert(s, k)$ if s is the src satellite
      else if $t_s = 0 \wedge t_k = 1$ **then**
        $\lambda_{r,f}.insert(s, k)$
      else if $t_s = 1 \wedge t_k = 0$ **then**
        $\lambda_{r,f}.insert(s)$ if s is the src satellite
      else if $t_s = 0 \wedge t_k = 0 \wedge (s, k) \notin$ path j **then**
        $\lambda_{r,f}.insert(s)$
      end if
    end if
    if link (s, k) in path j **then**
      if $t_s = 1 \wedge t_k = 1$ **then**
        $\lambda_{r,f}.insert(s, k)$ if s is the src satellite
      else if $t_s = 0 \wedge t_k = 1$ **then**
        $\lambda_{r,f}.insert(k)$
      else if $t_s = 1 \wedge t_k = 0$ **then**
        $\lambda_{r,f}.insert(s)$ if s is src satellite
      else if $t_s = 0 \wedge t_k = 0 \wedge (s, k) \notin$ path i **then**
        $\lambda_{r,f}.insert(k)$
      end if
    end if
  end for
end for

Algorithm 2 Reconfiguration message calculation

Initialization:
$\lambda_{r,f} = \emptyset, \forall f \in F, \forall r \in R$
while $i \leq N$ **do**
  j = i + 1
  Calculate $t :=$ XOR between path i and path j
  Identify the satellites who need reconfiguration messages according to Alg. 1
  Store $\lambda_{r,f}$
end while

flow path j (i.e., sub-case 5b), then both satellites require a reconfiguration message. Sub-case 5c occurs when satellite s is removed for the new flow path j, and as a result this satellite has to receive a reconfiguration message, only if it is the source satellite. Finally, sub-case 5d considers the possibility that the link between satellites s, k does not exist in the new flow path j, however, both satellites are part of the new flow path. In that sub-case, satellite s needs a reconfiguration message.

Moreover, Alg. 1 considers the second scenario of a link between satellites (s, k) established in the new flow path j. Following the same logic, 4 sub-cases are identified. Sub-case 6a considers both satellites s, k being added to the new flow path j. Consequently, satellite k receives a reconfiguration message, whereas satellite s receives a reconfiguration message if it is the source satellite. Sub-case 6b considers satellite k being added to the new flow path j. Thus, satellite k requires a reconfiguration message. Further, sub-case 6c considers the satellite s being added to the new flow path j. Therefore, satellite s receives a reconfiguration messages if it is the source satellite. Finally, sub-case 6d, accounts for a new link being established by satellites (s, k), however, both of them were part of the previous flow path i. In that regard, satellite k requires a reconfiguration message from the controller.

In order to ease the explanation and provide a better understanding for the re-configuration message algorithm, let us use an example. Fig. 7 portrays the changes of a flow path when a topology change occurs. Fig. 7a represents the initial flow path i before the change, whereas Fig. 7b shows the new flow path j. In this example, satellite s has been removed from the new flow path j. Consequently, according to Openflow [32], a new path insertion is needed for the nodes affected by this change (i.e., satellites f, and k, respectively), whereas a flow deletion is required for the removed satellite s. Satellite f receives a message from the controller according to the sub-case 5b, whereas satellite k receives a message from the controller according to sub-case 6d. In general, satellite s would have retrieved a deletion message from sub-case 5b and sub-case 5c, respectively. More specifically, if a satellite s between two satellites f, k of a flow path i is deleted then Algorithm 1 may result in a duplicated message. Thus, the notion of source satellite is introduced to cater for such an occurrence.

6.a. Link (s, k) is now part of flow path j, satellite k has been added to the flow path.
6.b. Link (s, k) is now part of flow path j, satellite s has been

added to the flow path.

6.c. Link (s, k) is now part of flow path j, both satellites s, k have been added to the flow path.

6.d. Link (s, k) is now part of flow path j, both satellites s, k have been part of the previous flow path.

The detailed operation of calculating the reconfiguration messages in the network is described in Alg. 2. For each flow $f \in F$ and for each snapshot $r \in R$, a new list referred to as $\lambda_{r,f}$ is created. After the initialization phase, we iterate over all the snapshots for a specific flow and apply the XOR operation between two consecutive rows $i, j \in PF$ to construct the vector t. Further, we use Alg. 1 to identify the satellites which require reconfiguration messages due to topology changes. According to Alg. 1, we account for all the possible scenarios that might trigger a reconfiguration message and insert the respective satellites in the corresponding list. Finally, we store the list values and apply the same method until we have calculated all the possible messages for all the flows $f \in F$ and snapshots $r \in R$.

Once we have collected all the possible satellites that will require reconfiguration messages from the controllers, we calculate the average flow reconfiguration time that occurs in the network as follows:

$$A_r = \frac{1}{|F|} \sum_{f \in F} \sum_{r \in R} \max_{\forall c \in C, \forall s \in \lambda_{r,f}} \{x_{s,c} \cdot d_{r,s,c}\} \tag{4}$$

According to (4), for each flow $f \in F$ and for each snapshot $r \in R$, the maximum delay to the controller is considered for the satellites which need a reconfiguration message. Further, the average flow reconfiguration time is computed by taking the average over all snapshots and flows.

3) Migration messages: While considering a logically centralized, but physically distributed SDN architecture, more than one controller might be available in the control layer. In order to maintain a global view of the network, these controllers need to constantly exchange messages amongst each other, which in turn may cause overhead on the system. However, this overhead has been covered in other works [22], hereafter is out of the scope of this paper. We focus on the overhead introduced to due migrations that might occur in the system namely **controller migrations** and **satellite-to-controller reassignment**, respectively. In general, if a dynamic controller placement is considered, there is a possibility to move the control logic over time (i.e., the satellites which act as controllers might differ over time). This introduces potential delays due to information exchange between the previously selected and newly selected controllers. The aforementioned delay is referred to as **control migration cost**. Furthermore, since the output of the dynamic controller placement defines also a satellite-to-controller assignment, a new controller selection might trigger also a new satellite-to-controller assignment. Consequently, an additional cost is introduced namely, **satellite-to-controller reassignment cost**. In the rest of the paper we refer to **migration cost** as the combination of the two. In order to ease the understanding of the migration process, the overall migration cost is summarized in Algorithm. 3. A detailed explanation of each of the aforementioned components is provided as follows:

Algorithm 3 Migration cost calculation

1: **Initialization**:
2: $\forall s \in S$ identify the controller $c \in C$ from the previous controller selection with the shortest distance according to (5).
3: Determine $\forall c \in C$ whether a migration is needed according to (7).
4: For all the controllers that have to be migrated calculate the maximum latency as in (8).
5: Determine $\forall s \in S$ whether a satellite has been reassigned to a new controller according to (10).
6: Calculate the switch reassignment cost as the maximum latency among all the reassigned satellite switches as presented in (11).
7: Store the total migration cost as $C_r + S_r$.

a. **Controller migration:** We assume a logically centralized, but physically distributed SDN control layer, where each SDN controller contains a local copy of the network state known as *data store* D of 100 MB as in [40]. For the migration link speed we assume a bandwidth L_s =1 Gbps. While a controller migration occurs, only the data store needs to be transferred to the newly selected controllers. Moreover, we assume that all the controllers are identical and the control migration happens simultaneously. As a result a new selected controller can get this information from any of the previous selected ones. Hence, for each of the network nodes, the minimum distance to the previously selected controllers is calculated as follows:

$$md_s = \operatorname{argmin} d(s, prev) \quad \forall s \in S \tag{5}$$

For each satellite $s \in S$, the controller with the minimum distance is recorded according to Equation (5). In the new controller selection procedure, if a satellite s is selected as the controller, the result of Equation (5) is utilized to select te controller from which it will retrieve the *data store D*.

In order to quantify the controller migration occurrences, in this paper we provide a mathematical framework as follows: We denote y_c as a binary variable which holds the value 1 if the controller $c \in C$ is selected in the new controller placement selection. Similarly, y'_c is defined as a binary variable which holds the value 1 if the controller $c \in C$ was selected in the previous controller placement selection. Further, we apply the XOR operation between variables y_c and y'_c, whose result is denoted by the variable θ_c as presented in inequality (6). This results again in a binary variable whose value is 1 if the controller $c \in C$ has changed his state in the new controller placement selection (i.e., either active from inactive or vice versa).

$$\forall c \in C: \quad \begin{aligned} &\theta_c \leq y_c + y'_c \\ &\theta_c \geq y_c - y'_c \\ &\theta_c \geq -y_c + y'_c \\ &\theta_c \leq 2 - y_c - y'_c \end{aligned} \tag{6}$$

Moreover, to determine the controllers which need to retrieve the data store from the previously selected controllers, we define the variable mc_c, which takes the value 1 if controller $c \in C$ is selected in the new controller placement selection but was not previously selected. The operation is a multiplication between the variable θ_c and y_c as expressed in following equation:

$$mc_c = \theta_c \cdot y_c, \forall c \in C \tag{7}$$

Since the migration of all controllers occurs simultaneously, the migration cost consists of the maximum propagation delay between the previous and newly selected controller plus the transfer duration of the data store [40]. The controller migration cost then it is calculated according to the formula below:

$$C_r = \max_{\forall c \in C} \left[mc_c \cdot d(md_c, c) + \frac{D}{L_s} \right] \tag{8}$$

b. **Switch reassignment:** Dixit et al. [41] provided a mechanism to quantify the cost of the switch reassignment process. Moreover, Bari et al. [22] provided a mathematical description for considering the controller migration. However, the switch-to-controller reassignment was not taken into account. In order to fill this void and encounter for the overhead, in our model we provide a method for quantifying the number of migration messages needed both for controller migration and satellite-to-controller reassignment.

We define the binary variable $x_{s,c}$ for the satellite SDN switch-to controller assignment. The value of $x_{s,c}$ results in 1 if the satellite SDN switch $s \in S$ is assigned to controller c in the new controller assignment, whereas $x'_{s,c}$ the value 1 if satellite SDN switch $s \in S$ was assigned to controller c in the previous controller placement selection. We further denote variable $\zeta_{s,c}$ as a binary result of the XOR operation between $x_{s,c}$ and $x'_{s,c}$ as follows:

$$\forall s \in S, \forall c \in C: \quad \begin{aligned} &\zeta_{s,c} \leq x_{s,c} + x'_{s,c} \\ &\zeta_{s,c} \geq x_{s,c} - x'_{s,c} \\ &\zeta_{s,c} \geq -x_{s,c} + x'_{s,c} \\ &\zeta_{s,c} \leq 2 - x_{s,c} - x'_{s,c} \end{aligned} \tag{9}$$

The value of $\zeta_{s,c}$ is 1 if satellite $s \in S$ is controlled by a new controller c in the new controller placement selection or 0 otherwise. As introduced in [41], 6 messages need to be exchanged between the new controller and the switches, which includes hello and handshake messages. We denote the variable $ms_{s,c}$ to identify the controller that the satellite SDN switch has to establish the new connection with as follows:

$$ms_{s,c} = \zeta_{s,c} \cdot x_{s,c}, \forall s \in S, \forall c \in C \tag{10}$$

Given the fact that the switch reassignment occurs simultaneously, the switch reassignment cost is the maximum propagation delay of all the satellite SDN switches to the new controllers. This is expressed by the equation as follows:

$$S_r = \max_{\forall s \in S, \forall c \in C} [ms_{s,c} \cdot 6 \cdot d(s,c)] \tag{11}$$

D. Optimization Problem Formulation

The dynamic controller placement is formulated as an ILP and is solved in a Gurobi framework implemented in Python. The input parameters for this optimization problem along with the optimization variables are denoted by Table II. The optimization goal is to minimize the average flow setup time for a given flow profile F.

$$\min \frac{1}{|F|} A_f \tag{12}$$

The constraints related to the controller placement problem are as follows: Constraint (13), which assures that the total number of controllers to be placed in the network is K.

$$\sum_{c \in C} y_c = K, \tag{13}$$

Constraint (14), guarantees that a satellite s is controlled by a controller c only if the controller is active.

$$x_{s,c} \leq y_c, \forall s \in S, \forall c \in C \tag{14}$$

Constraint (15), ensures that each satellite s is controlled by exactly one controller.

$$\sum_{c \in C} x_{s,c} = 1, \forall s \in S \tag{15}$$

If two satellites belong to different controller domains they need to be assigned to different controllers. A helper binary variable $z_{c,s,k}$ is identified to quantify such an occurrence. Equation (16) separates satellites in distinct controller domain clusters.

$$z_{c,s,k} = x_{s,c} \cdot x_{k,c}, \forall s \in S, \forall k \in S, \forall c \in C \tag{16}$$

In order to be able to solve the problem in a linear optimizer, Equation (16) is replaced by the following three equations.

$$z_{c,s,k} \leq x_{s,c}, \forall s \in S, \forall k \in S, \forall c \in C \tag{17a}$$

$$z_{c,s,k} \leq x_{k,c}, \forall s \in S, \forall k \in S, \forall c \in C \tag{17b}$$

$$z_{c,s,k} = x_{s,c} + x_{k,c} - 1, \forall s \in S, \forall k \in S, \forall c \in C \tag{17c}$$

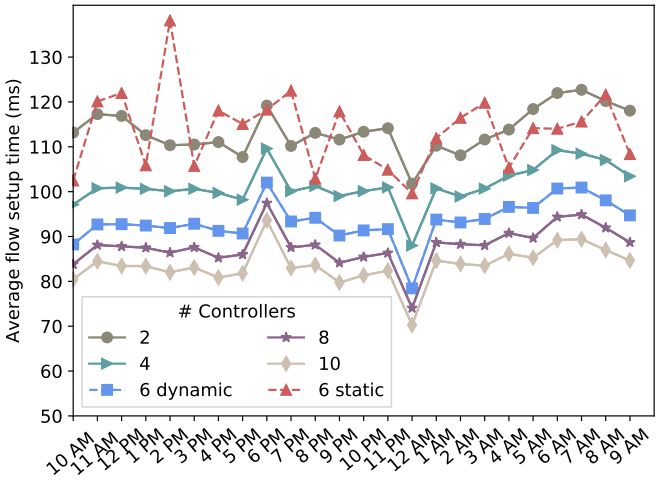


Figure 8: Optimal average flow setup time for various simulation GMT hours as a function of the number of controllers K. Furthermore the average flow setup time is illustrated for the static approach with red dotted line for $K = 6$.

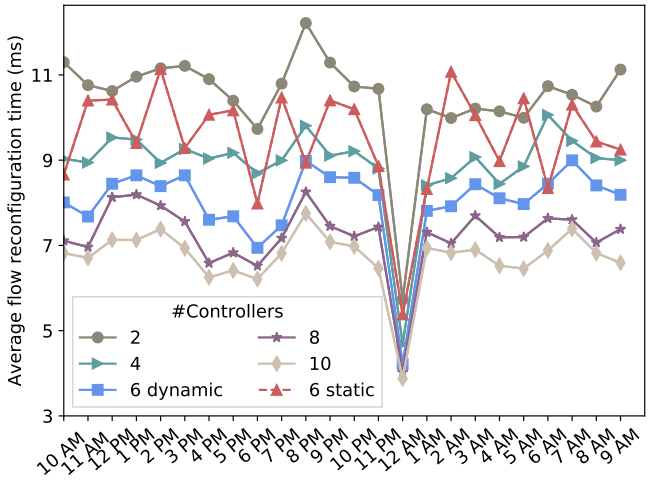


Figure 9: Optimal average flow reconfiguration time for various simulation GMT hours as a function of the number of controllers K. Furthermore the average flow reconfiguration time is illustrated for the static approach with red dotted line for $K = 6$.

IV. Results and Discussions

In this section we discuss the main findings of our paper. We evaluate our SDN-enabled LEO constellation and give insights on the network performance with respect to the average flow setup time, average flow reconfiguration time cost and migration time cost. Moreover, we compare our architecture with existing ones in the literature.

A. Dynamic SDN Controller Placement

We begin our evaluation with the dynamic controller placement problem in a LEO constellation network. We focus our study on two main metrics namely, average flow setup time and average flow reconfiguration time respectively. The former is the time it takes for a flow path to be established, whereas the latter is the required time for a flow path to be re-established due to topology changes. For the evaluation we consider a daily simulation ranging from 7/25/2017 *10* AM GMT up to 7/26/2017 *9* AM GMT. Further, we apply our optimization algorithm each hour. The outcome of the optimization is the controller selection and the satellite-to-controller assignment. In our approach we vary the number of controllers K that can be placed in the network to examine the effect on the considered performance metrics. As aforementioned, in a distributed control layer, the synchronization cost is correlated to the number of controllers present in the control layer. Hence, considering the synchronization overhead the adequate number of controllers on the control layer must be identified [42]. Therefore, not all the satellites of the constellation can be assigned as controllers. For our analysis we limit the number of deployed controllers in the network to 10. Fig. 8 illustrates the impact of the number of controllers on the average flow setup time. The average flow setup time is decreasing with the number of controllers deployed in the network. This effect is noticed for all the considered hours in our simulation. Moreover, it can be observed that the minimum average flow setup time is recorded for the simulation hour which corresponds to *12* AM GMT. On the other hand, the highest average flow setup time is recorded at *6* PM GMT. If we recall Fig. 3, the highest traffic density corresponds to Europe and therefore the highest and lowest flow setup times are recorded for the most or least occupied hours in this region. These results again emphasize the impact of the traffic demands on the performance of the network and highlight the importance of accounting for such variations in our model.

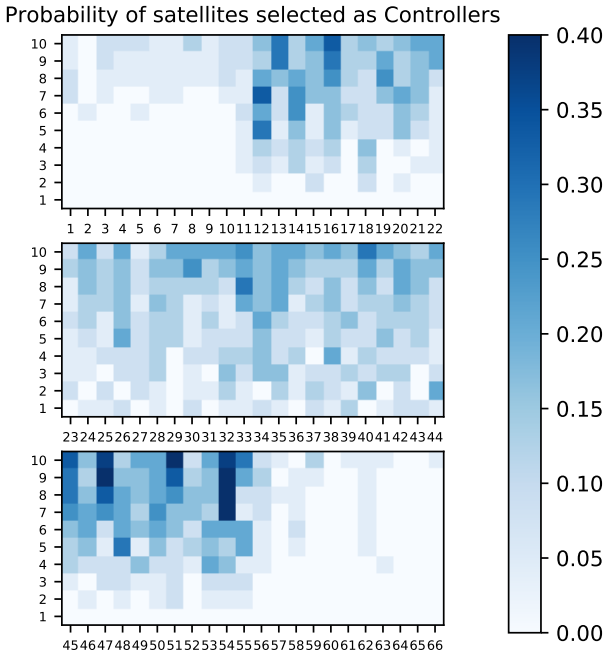


Figure 10: Heat map presenting the frequency of each satellite being selected as a controller for various number of deployed controllers in the network.

We further proceed with our analysis considering the aver-

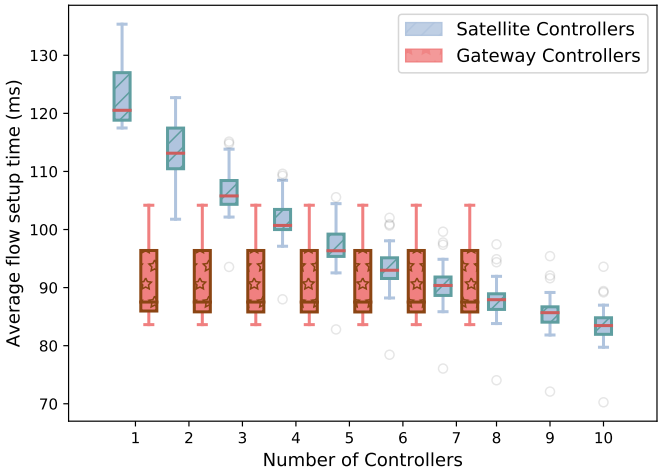


Figure 11: Optimal average flow setup time *ms* for different architecture as a function of the number of controller K.

age flow reconfiguration time. Alternatively from the average flow setup time, the average flow reconfiguration time is not the objective of our optimization model. However, similarly to the previous scenario we use the outcome of our optimization algorithm (i.e., controller selection and satellite-to-controller assignment) for our evaluation. Analogous to the previous analysis, we vary the number of controllers deployed in the network to examine the effect on the average flow reconfiguration time. The number of controllers is kept at 10 and we calculate the average flow reconfiguration time for all the simulation hours. In that regard, Fig. 9 presents the results for this performance metric. A similar trend to the average flow setup time can be observed. Even in this case, the average flow reconfiguration time is decreasing with the number of controllers deployed in the network. Moreover, again the minimum average flow reconfiguration time is recorded for the simulation hour *12* AM GMT, whereas the highest average flow reconfiguration time is noticed at simulation hours *8* PM GMT, which corresponds to most and least occupied hours in the region of Europe.

B. Dynamic vs. Static Controller Placement

Following the logic of dynamic and reconfigurable LEO constellations, our next evaluation consists on the comparison between the dynamic and static controller placement. In our model, we apply our optimization algorithm each simulation hour and as a result a different controller placement or satellite-to-controller assignment might occur, thus rendering the system more dynamic. Alternatively, the static controller placement does not account for such changes but instead deploys a static controller selection and satellite-to-controller assignment for all the simulation hours. However, considering the dynamic nature of LEO constellations and variations in the traffic patterns over time and space, such an approach might result inefficient. In order to answer this question, in this subsection we compare our proposed dynamic approach with the static one. To account for a fair comparison, we determine the static controller selection and satellite-to-controller assignment by considering the selection which achieves the best overall performance for all the simulation hours. In more details, we construct a probability heat map as illustrated in Fig. 10. The heat map presents the frequency of a satellite being selected as a controller with respect to various number of deployed controllers in the network. It can be observed that the satellites positioned at the counter-rotating planes of the constellation (i.e., $1^{st}, 6^{th}$ orbital planes) have the lowest probability of selection for all the considered controllers. This result is related to the fact that these satellites only contain 3 ISLs compared to the remaining satellites of the constellation which in turn contain 4 ISLs. Hence, longer paths are needed to reach the remaining satellites in case that they operate as controllers. This results in higher flow path setup times and therefore are not preferred by the optimization algorithm.

In our evaluation, we fix the number of controllers to K=6 for this comparison. The results are shown both for the average flow setup and flow reconfiguration time in Fig. 8 and Fig. 9 respectively. It can be noticed that for both performance metrics the trend is similar, the dynamic approach performs better than the static one. Regarding the average flow setup time, which is the metric we optimize for, as shown in Fig. 8 the difference in performance varies between 8.5% and 40%, whereas on average the dynamic approach results 20% more efficient. Similarly, the difference in performance between the dynamic and static approach for the average flow reconfiguration time varies between 7% and 33%, whereas again on average the dynamic approach results approximately 20% more efficient as illustrated in Fig. 9. Considering the above shown results, we can argue that the static controller placement indeed results inefficient especially in a dynamic environment with high traffic fluctuations. Nonetheless, regarding our third performance metric, migration cost, the dynamic approach introduces more overhead. This overhead is relatively high and as such can result fatal for the user experience if not tackled, since it can result in packet losses and thus disturb applications such as video streaming. Therefore, when planning the network, a trade-off must be obtained.

C. Architecture Comparison

Compared to our architecture, two alternative SDN-enabled satellite architectures exist in the literature. [29] considers the SDN control layer on the Satellite Gateways (SG), whereas [27] proposes the SDN control layer on GEO satellites. Differently from our work, none of the above mentioned studies targets the controller placement and therefore there exist no mathematical framework to realize a comparison. In order to fairly compare the systems, we leverage the traffic model and dynamic controller placement used in our paper.

For each of the simulation hours we apply our optimization problem and obtain the controller selection and satellite-to-controller assignment for each architecture. Further, we group the output of the 24 considered simulation hours and we construct boxplots to have a clear view on the variation for the considered performance metrics. We repeat our evaluation for the average flow setup time, average flow reconfiguration time and migration time respectively. We then vary the number of

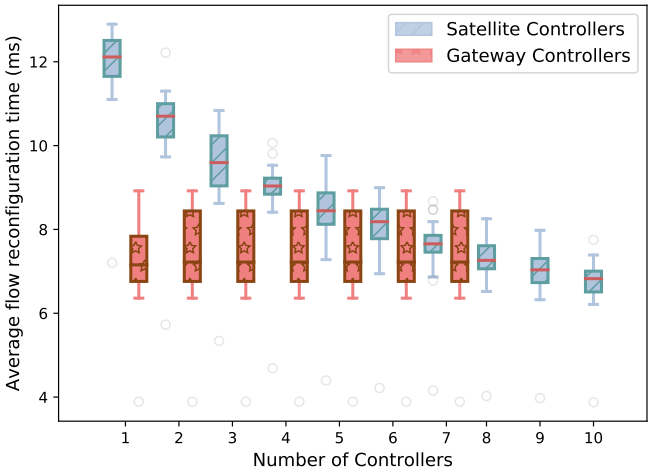


Figure 12: Average flow reconfiguration time *ms* for different architecture as a function of the number of controller K.

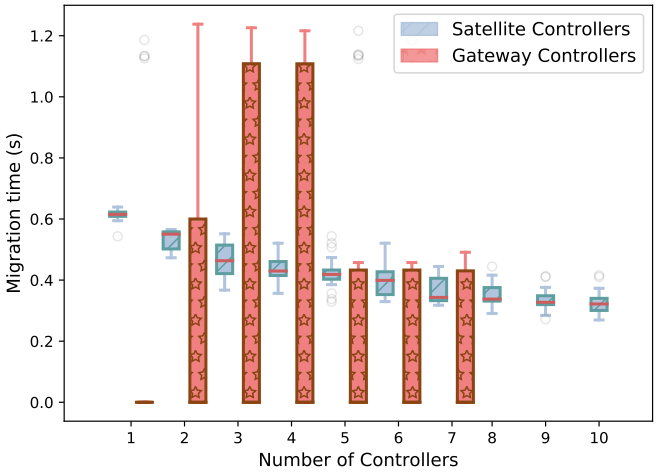


Figure 13: Average flow migration *ms* for different architecture as a function of the number of controller K.

deployed controllers in the network K in order to quantify the impact on the network performance. The number of controllers K is varied and the results are illustrated in Fig. 11, Fig. 12 and Fig. 13 respectively. Regarding our proposed SDN-enabled LEO constellation architecture, a common trend can be observed for all the considered metrics. It is clear that the network performance is increased with the number of controllers deployed in the network. However, this performance increase is not linear. It can be noticed that the law of diminishing returns applies, since after a specific amount of controllers the benefits are minor. Alternatively, for the SG-controllers architecture, regardless of the number of controllers K deployed in the network the average flow setup and average flow reconfiguration time does not decrease. The rational behind this is that as shown in Fig. 2, the satellite gateways are geographically close to each other and therefore deploying more controllers does not bring any benefit.

Regarding the architecture comparison, it can be noticed both for Fig. 11 and Fig. 12 that for a low number of controllers K deployed in the network, the SG-controller architectures performs better than our SDN-enabled LEO architecture, however this performance difference decreases with increasing number of controllers. If more than 7 controllers are considered, the LEO-controllers achieves better results. Since the number of SG is fixed to 7, there is no further improvement in the SG-gateways solution. Instead in LEO-controllers, further increasing the number of controllers can certainly improve the performance, however at higher operational expenditures, hence requiring for a trade-off from the architectural viewpoint.

Due to the high altitude of GEO satellites, the forwarding latency towards the satellites of the LEO constellation which is more than 100 *ms* effects drastically the average flow setup time. These values vary between 320 *ms* up to 335 *ms*, more than twice of the values recorded for the LEO-based SDN architecture. Hence, full analysis of the results for the GEO-based architecture is omitted.

Finally, we compare our LEO-constellation architecture with the SG-controller one regarding the migration time cost for each of the simulated hours. The results for this performance metric are presented in Fig. 13. Unlike the two first performance metrics used to compare the two architectures, the migration time cost is in terms of seconds. According to Fig. 13, an interesting result is observed. The median of the migration time for various number of controllers for the SG-controller architecture is 0. Due to the fact that during the optimal controller selection, the same controller was used for several GMT hours, in more than half of the hours the migration time is 0. However, at the point where the migration was required, really high values were noticed. The 75-percentiles for different controllers deployed in the network vary between 0.43 and 1.1 seconds. Instead, the 75-percentiles achieved for the LEO-controllers vary between 0.34 to 0.62 seconds, resulting in much lower migration times. While comparing the 75-percentiles, the proposed architecture outperforms the second architectures for all the number of used controllers apart from the case where one controller is deployed in the network. This difference varies between 82% for 4 controllers and 3% for 6 used controllers respectively. Moreover, from Fig. 13 one can identify the decrease in the difference between the two architecture when more controllers are deployed. Given the fact that the total number of controllers that can be placed for the Gateway-Controller architecture is 7, the probability of migrating the controllers becomes lower.

V. Conclusion

Given the high interest towards the integration of satellite and terrestrial networks, solutions which rely on programmable and adaptable concepts are proposed to achieve this unification. From the architectural perspective Software-Defined Networking (SDN) solutions are envisioned to provide the satellite networks with the flexibility required to achieve this unification. To this aim, the separation of the control layer from the data layer of satellite networks is proposed. However, the introduction of the SDN paradigm is relatively new to the satellite domain especially in the satellite

space segment. In order to fill this void, in this paper we proposed an SDN-enabled LEO constellation satellite network driven by means of SDN. In that regard, we formulated an optimization problem to tackle the SDN controller placement and switch-to-controller assignment by taking into account topology changes and traffic variations in space and time. Moreover, we introduced the notion of control migration and path reconfiguration and developed a mathematical approach to quantify and provide insights on the impact of the number of controllers to this overhead. Finally, we measured the amount of overhead messages for various approaches in the literature and provided a comparison with our proposed method. The results show that our approach minimizes the overhead both for the migration cost and reconfiguration cost when the optimal controller placement is considered with respect to the average flow setup time given a set of user demands.

REFERENCES

[1] N. Alliance, "Description of network slicing concept," *NGMN 5G P*, vol. 1, 2016.

[2] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computingA key technology towards 5G," *ETSI white paper*, vol. 11, no. 11, pp. 1–16, 2015.

[3] ETSI, "Satellite Earth Stations and Systems (SES); Combined Satellite and Terrestrial Networks scenarios." Tech. Rep. 103 124, 2013.

[4] A. Guidotti, A. Vanelli-Coralli, M. Caus, J. Bas, G. Colavolpe, T. Foggi, S. Cioni, A. Modenini, and D. Tarchi, "Satellite-enabled LTE systems in LEO constellations," in *Communications Workshops (ICC Workshops), 2017 IEEE International Conference on*. IEEE, 2017, pp. 876–881.

[5] A. Varasteh, S. Hofmann, N. Deric, M. He, D. Schupke, W. Kellerer, and C. Mas Machuca, "Mobility-aware joint service placement and routing in space-air-ground integrated networks," *arXiv preprint arXiv:1902.01682*, 2019.

[6] R. Ferrús, H. Koumaras, O. Sallent, G. Agapiou, T. R. M., A. Kourtis, C. Boustie, P. Gelard, and T. Ahmed, "SDN/NFV-enabled satellite communications networks: opportunities, scenarios and challenges," *Phys. Commun.*, vol. 18(2), pp. 95–112, 2016.

[7] L. Bertaux, S. Medjiah, P. Berthou, S. Abdellatif, A. Hakiri, P. Gelard, F. Planchou, and M. Bruyere, "Software defined networking and virtualization for broadband satellite networks," *IEEE Communications Magazine*, vol. 53, no. 3, pp. 54–60, Mar. 2015.

[8] L. Boero, R. Bruschi, F. Davoli, M. Marchese, and F. Patrone, "Satellite Networking Integration in the 5G Ecosystem: Research Trends and Open Challenges," *IEEE Network*, vol. 32, no. 5, pp. 9–15, 2018.

[9] M. Sheng, Y. Wang, J. Li, R. Liu, D. Zhou, and L. He, "Toward a flexible and reconfigurable broadband satellite network: Resource management architecture and strategies," *IEEE Wireless Communications*, vol. 24, no. 4, pp. 127–133, 2017.

[10] T. Ahmed, A. Alleg, R. Ferrus, and R. Riggio, "On-demand network slicing using SDN/NFV-enabled satellite ground segment systems," in *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)*. IEEE, 2018, pp. 242–246.

[11] T. Ahmed, E. Dubois, J.-B. Dupé, R. Ferrús, P. Gélard, and N. Kuhn, "Software-defined satellite cloud RAN," *International Journal of Satellite Communications and Networking*, vol. 36, no. 1, pp. 108–133, 2018.

[12] A. Tootoonchian, S. Gorbunov, Y. Ganjali, M. Casado, and R. Sherwood, "On Controller Performance in Software-Defined Networks," in *Presented as part of the 2nd Workshop on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services*, 2012.

[13] Y. Ganjali and A. Tootoonchian, "HyperFlow: A Distributed Control Plane for OpenFlow." in *INM/WREN*, 2010.

[14] S. Hassas Yeganeh and Y. Ganjali, "Kandoo: a framework for efficient and scalable offloading of control applications," in *Proceedings of the first workshop on Hot topics in software defined networks*. ACM, 2012, pp. 19–24.

[15] A. Papa, T. de Cola, P. Vizarreta, M. He, C. Mas Machuca, and W. Kellerer, "Dynamic SDN Controller Placement in a LEO Constellation Satellite Network," in *IEEE GLOBECOM*, 2018.

[16] Z. Zhang, W. Zhang, and F.-H. Tseng, "Satellite mobile edge computing: Improving QoS of high-speed satellite-terrestrial networks using edge computing techniques," *IEEE Network*, vol. 33, no. 1, pp. 70–76, 2019.

[17] A. Guidotti, A. Vanelli-Coralli, M. Conti, S. Andrenacci, S. Chatzinotas, N. Maturo, B. Evans, A. Awoseyila, A. Ugolini, T. Foggi *et al.*, "Architectures and key technical challenges for 5G systems incorporating satellites," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 3, pp. 2624–2639, 2019.

[18] T. de Cola, A. Perez-Neira, R. Channasandra, and S. Covaci, "Guest editorial," *IEEE Network*, vol. 32, no. 5, pp. 6–8, Sep. 2018.

[19] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," in *Proceedings of the first workshop on Hot topics in software defined networks*. ACM, 2012, pp. 7–12.

[20] S. Lange, S. Gebert, T. Zinner, P. Tran-Gia, D. Hock, M. Jarschel, and M. Hoffmann, "Heuristic approaches to the controller placement problem in large scale SDN networks," *IEEE Transactions on Network and Service Management*, vol. 12, no. 1, pp. 4–17, 2015.

[21] P. Vizarreta, C. Mas Machuca, and W. Kellerer, "Controller placement strategies for a resilient SDN control plane," in *Resilient Networks Design and Modeling (RNDM), 2016 8th International Workshop on*. IEEE, 2016, pp. 253–259.

[22] M. F. Bari, A. R. Roy, S. R. Chowdhury, Q. Zhang, M. F. Zhani, R. Ahmed, and R. Boutaba, "Dynamic Controller Provisioning in Software Defined Networks," in *Proc. 9th Int. Conf. Network and Service Management (CNSM 2013)*, Oct. 2013, pp. 18–25.

[23] M. He, A. Basta, A. Blenk, and W. Kellerer, "Modeling flow setup time for controller placement in SDN: Evaluation for dynamic flows," in *Proc. IEEE Int. Conf. Communications (ICC)*, May 2017, pp. 1–7.

[24] I. del Portillo, B. G. Cameron, and E. F. Crawley, "A technical comparison of three low earth orbit satellite constellation systems to provide global broadband," *Acta Astronautica*, vol. 159, pp. 123–135, 2019.

[25] M. Handley, "Delay is not an option: Low latency routing in space," in *Proceedings of the 17th ACM Workshop on Hot Topics in Networks*, 2018, pp. 85–91.

[26] D. Bhattacherjee and A. Singla, "Network topology design at 27,000 km/hour," in *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies*, 2019, pp. 341–354.

[27] J. Bao, B. Zhao, W. Yu, Z. Feng, C. Wu, and Z. Gong, "OpenSAN: a software-defined satellite network architecture," in *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 4. ACM, 2014, pp. 347–348.

[28] T. Li, H. Zhou, H. Luo, Q. Xu, and Y. Ye, "Using SDN and NFV to Implement Satellite Communication Networks," in *Proc. Int. Conf. Networking and Network Applications (NaNA)*, Jul. 2016, pp. 131–134.

[29] P. Du, S. Nazari, J. Mena, R. Fan, M. Gerla, and R. Gupta, "Multipath TCP in SDN-enabled LEO satellite networks," in *Proc. MILCOM 2016 - 2016 IEEE Military Communications Conf*, Nov. 2016, pp. 354–359.

[30] Y. Zhu, L. Qian, L. Ding, F. Yang, C. Zhi, and T. Song, "Software defined routing algorithm in LEO satellite networks," in *2017 International Conference on Electrical Engineering and Informatics (ICELTICs)*. IEEE, 2017, pp. 257–262.

[31] B. Yang, Y. Wu, X. Chu, and G. Song, "Seamless handover in software-defined satellite networking," *IEEE Communications Letters*, vol. 20, no. 9, pp. 1768–1771, 2016.

[32] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[33] S. Xu, X.-W. Wang, and M. Huang, "Software-defined next-generation satellite networks: architecture, challenges, and solutions," *IEEE Access*, vol. 6, pp. 4027–4041, 2018.

[34] E. Lutz, M. Werner, and A. Jahn, *Satellite Systems for Personal and Broadband Communications.* Springer, 2000.

[35] AGI, "Systems Tool Kit (STK)," https://www.agi.com/products/engineering-tools.

[36] C. Chen and E. Ekici, "A routing protocol for hierarchical LEO/MEO satellite IP networks," *Wireless Networks*, vol. 11, no. 4, 2005.

[37] X. Alberti, J. M. Cebrian, A. D. Bianco, Z. Katona, J. Lei, M. A. Vazquez-Castro, A. Zanus, L. Gilbert, and N. Alagha, "System capacity optimization in time and frequency for multibeam multi-media satellite systems," in *2010 5th Advanced Satellite Multimedia Systems Conference and the 11th Signal Processing for Space Communications Workshop*, Sept 2010, pp. 226–233.

[38] CISCO, "The zettabyte era: Trends and analysis," CISCO, Tech. Rep., 2017.

[39] E. Sakic and W. Kellerer, "BFT protocols for heterogeneous resource allocations in distributed SDN control plane," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–7.
[40] M. He, A. Basta, A. Blenk, and W. Kellerer, "How flexible is dynamic SDN control plane?" in *2017 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. IEEE, 2017, pp. 689–694.
[41] A. Dixit, F. Hao, S. Mukherjee, T. Lakshman, and R. Kompella, "Towards an elastic distributed SDN controller," in *ACM SIGCOMM Computer Communication Review*, vol. 43, no. 4. ACM, 2013, pp. 7–12.
[42] E. Sakic and W. Kellerer, "Response Time and Availability Study of RAFT Consensus in Distributed SDN Control Plane," *IEEE Transactions on Network and Service Management*, vol. 15, no. 1, pp. 304–318, 2018.

Arled Papa completed his Bachelor of Science in Electronics Engineering at the Polytechnic University of Tirana, Albania in 2015. He received his Master of Science in Communications Engineering at the Technical University of Munich (TUM) in November 2017 with high distinction. In February 2018 he joined the Chair of Communication Networks at TUM as a research and teaching associate. His research focuses on the design and analysis of QoS, Network Slicing and Network Virtualization concepts in Aeronautical applications, in-flight entertainment and connectivity services.

Tomaso de Cola Cola was born in Manosque, France, on April 28, 1977. He received the "Laurea" degree (with honors) in telecommunication engineering, in 2001, the Qualification degree as Professional Engineer in 2002 and the Ph. D. degree in Electronic and Computer Engineering, Robotics and Telecommunications in 2010 from the University of Genoa, Italy. From 2002 until 2007, he worked with the Italian Consortium of Telecommunications (CNIT), University of Genoa Research Unit, as scientist researcher. Since 2008, he has been with the German Aerospace Centre (DLR), where he is involved in different European Projects focusing on different aspects of DVB standards, CCSDS protocols and testbed design. He is co-author of more than 80 papers, including international conferences and journals. His main research activity concerns: TCP/IP protocols, satellite networks, transport protocols for wireless links, interplanetary networks as well as delay tolerant networks. Dr. de Cola has served on the technical program committee at several IEEE International Conferences and served as Guest Editor of IEEE Network, IEEE JSAC, and IEEE Wireless Communication Magazine. He currently serves as editor of IEEE Communication Letters, IEEE Wireless Communication Letters, IEEE Journal, and Elsevier Computer Networks. He is acting as chair of the Satellite and Space Communications (SSC) technical committee within ComSoc, deputy-area director of the Space-Internetworking Services (SIS) within the CCSDS standardization body as well as chair of the SatCom working group within the ETP Networld2020 platform.

Petra Vizarreta received the Dr.-Ing. (Ph.D.) degree from the Technical University of Munich in 2019, the Master degree from Karlsruhe Institute of Technology and Polytechnic University of Catalonia in 2011 and the Bachelor degree from University of Belgrade in 2009. She joined the Chair of Communication Networks at Technical University of Munich as a researcher in September 2015. Her research interest include modelling and design of dependable softwarized networks.

Mu He received his Masters Degree in Communication Engineering from the Technical University of Munich (TUM) in 2015. Currently he is pursuing his Ph.D. degree at the chair of communication networks of Prof. Wolfgang Kellerer. His research interests cover network planning in the context of software defined networking, network optimization and runtime reconfiguration with programmable data planes.

Carmen Mas Machuca (M'96-SM'12) is a Privat Dozent/Adjunct Teaching Professor at the Chair of Communication Networks, Technical University of Munich (TUM), Germany. She received her Dipl.-Ing. degree (Master) from Universitat Politècnica de Catalunya (UPC, Spain) in 1995 and her Dr.-Ing. degree (Ph.D.) from École Polytechnique Fédérale de Lausanne (EPFL, Switzerland) in 2000. Dr. Mas-Machuca has published more than 150 peer-reviewed papers. Her main research interests are in the area of techno-economic studies, network planning and resilience, SDN/NFV optimization problems and next generation converged access networks.

Wolfgang Kellerer (M'96-SM'11) is a Full Professor with the Technical University of Munich (TUM), heading the Chair of Communication Networks at the Department of Electrical and Computer Engineering. Before, he was for over ten years with NTT DOCOMO's European Research Laboratories. He received his Dr.-Ing. degree (Ph.D.) and his Dipl.-Ing. degree (Master) from TUM, in 1995 and 2002, respectively. His research resulted in over 200 publications and 35 granted patents. He currently serves as an associate editor for IEEE Transactions on Network and Service Management, IEEE Networking Letters and on the Editorial Board of the IEEE Communications Surveys and Tutorials. He is a member of ACM and the VDE ITG.