H2020 5G-Crosshaul project
Grant No. 671598

D4.2 - Final design of 5G-Crosshaul Applications and Algorithms

Abstract

This Deliverable describes the final design of all methods and algorithms in WP4. It includes refinements of the design described in D4.1, and it outlines the implementation procedure, the addressed KPIs, the evaluation and validation tests that have been performed. Finally, it lists potential requirements on interfaces to neighbouring network domain controllers.

Document Properties

Document Number:	D4.2
Document Title:	**Final design of 5G-Crosshaul Applications and Algorithms**
Document Responsible:	POLITO
Document Editor:	Claudio Casetti (POLITO)
Target Dissemination Level:	Public
Status of the Document:	Final
Version:	1
Reviewers:	

Disclaimer: *This document has been produced in the context of the 5G-Crosshaul Project. The research leading to these results has received funding from the*

European Community's H2020 Programme under grant agreement N° H2020-671598.

Table of Content

List of Contributors

Partnet No.	Partner Short Name	Contributor's name
P01	UC3M	Sergio González, Antonio de la Oliva
P02	NEC	Xi Li, Andrés García Saavedra
P05	ATOS	José Enrique González Blázquez, Beatriz Lopez
P11	VISIONA	David Jiménez, Danny R. Fonseca
P13	NXW	Giada Landi, Francesca Moscatelli, Marco Capitani, Elian Kraja
P16	FhG-HHI	Kei Sakaguchi
P17	CTTC	Ramón Casellas Ricard Vilalta, José Núñez, Josep Mangues, Marco Miozzo
P18	CREATE-NET	Leonardo Goratti, Domenico Siracusa, Raihana Ferdous
P19	POLITO	Carla Fabiana Chiasserini, Claudio Ettore Casetti
P21	ITRI	Samer Talat, Sean Chang, Shahzoob Bilal Chundrigar, Chia-Lin Lai

List of Figures

List of Tables

List of Acronyms

Acronym	Description
5G	5th Generation
AP	Access Point
API	Application Programming Interface
BER	Bit Error Rate
BS	Base Station
BSS	Business Support Systems
CAPEX	Capital Expenditure
CDN	Content Delivery Network
CDNMA	Content Delivery Network Management Application
CIR	Committed Information Rate
CP	Control Plane
CPU	Central Processing Unit
DB	Data Base
DHCP	Dynamic Host Configuration Protocol
EIR	Extended Information Rate
EMMA	Energy-Management and Monitoring Application
ETSI	European Telecommunications Standards Institute
GUI	Graphical User Interface
GW	Gateway
HD	High Definition
HO	Handover

HST	High Speed Train
ICMP	Internet Control Message Protocol
ID	Identifier
ILP	Integer Linear Programing
INLP	Integer Non-Linear Programming
IR	Internal Report
KPI	Key Performance Indicator
MAC	Media Access Control
MANO	Management and Orchestration
MEC	Mobile Edge Computing
MMA	Mobility Management Application
MOS	Mean Opinion Score
MTA	Multi-Tenancy Application
NBI	Northbound Interface
NFV	Network Function Virtualization
NFVO	Network Function Virtualization Operator
NIC	Network Interface Card
OF	Open Flow
ONOS	Open Network Operating System
OPD	OpenDaylight
OPEX	Operational Expenditure
OS	Operating System
OTT	Over-The-Top
PoA	Point of Access
PoC	Point of Connection

PPP	Public Private Partnership
QoE	Quality of Experience
QoS	Quality of Service
RAN	Radio Access Network
REST	Representational State Transfer
RMA	Resource Management Application
RRH	Remote Radio Head
SDN	Software-Defined Networking
SLA	Service Level Agreement
SP	Service Provider
SQL	Structured Query Language
SUT	System Under Test
TCP	Transmission Control Protocol
TE	Traffic Engineering
TTL	Time to Live
TVBA	TV Broadcasting Application
UE	User Equipment
UP	User Plane
vCDN	Virtual Content Delivery Network
vEPC	Virtual Evolved Packet Core
VI	Virtual Infraestructure
VIM	Virtual Infrastructure Manager
VIMaP	Virtual Infrastructure manager and Planner Application
VLAN	Virtual Local Area Network
VM	Virtual Machine

VN	Virtual Network
VNE	Virtual Network Embedding
VNF	Virtual Network Function
VNFP	Virtual Network Function Provider
VNF-FG	Virtual Network Function Forwarding Graph
VNO	Virtual Network Operator
VNP	Virtual Network Provider
VoD	Video on Demand
WP	Work Package
XCF	5G-Crosshaul Common Frame
XCI	5G-Crosshaul Control Infrastructure
XFE	5G-Crosshaul Forwarding Element
XPFE	5G-Crosshaul Packet Forwarding Element
XPU	5G-Crosshaul Processing Unit

Executive Summary

This deliverable presents the work accomplished within the scope of Work Package 4 (WP4) of the 5G-Crosshaul project. Like D4.1 before it, this document is focused on 5G-Crosshaul applications, more specifically, on the final application design, implementation, interaction and the evaluation of each of these applications. Solidly based on the experience gained from the experimentation of Crosshaul applications, this document provides a more refined view of the applications and of how they relate to the rest of the Crosshaul architecture, than what was shown in results D4.1.

Unlike the previous deliverable, we have chosen to take a per-application approach in the description of design and evaluation, devoting one section for each application (Sections 2 through to Section 13). Every section thus starts with a summary of the application, its purpose and goals, the input it requires and the services it provides. This introduction is followed by a subsection on the application consolidated design, describing its place in the 5G-Crosshaul architecture through a schematic visualization. Next, implementation details are provided, for each internal component of the application, highlighting, where possible, the interactions with other components and applications and describing the required interfaces. Each application section then carries a specific subsection that summarizes in tabular form the project KPIs addressed: specifically, it focuses on how to measure such KPIs in terms of what metrics, their description and the measurement procedure. Finally, a validation and evaluation subsection describes the performance of the application, detailing the methodology for the evaluation (simulation, emulation, test-bed implementation) and illustrating performance evaluation results.

Section 14 relates the experiments conducted in the various demos in WP5 with the applications developed within WP4 as described in this deliverable that were tested therein. The mapping to the WP5 demos shows how the developed Crosshaul applications are integrated into the different demonstrators for the system evaluation.

After all applications have been thus described, a final section (Section 15) specifies the requirements that the application layer places on the XCI through its NBI or EBI/WBI, in presence of neighboring mobile core and access architectures. Two interaction models, a hierarchical and a peer-to-peer one, are presented, referring to other two Phase-1 projects (5G-NORMA and 5G-Ex, respectively). Then a per-application table summarizes their requirements from and to RAN and core network domains.

1 Introduction

The objective of this document is to present the final design of the 5G-Crosshaul applications and algorithms defined in the scope of WP4. Similarly to what was done in D4.1, each application is presented detailing the design, the implementation procedure, the KPIs that it addresses, and the evaluation and validation tests that have been performed. Sections 2 through to 13 contain such detailed description of the applications developed within 5G-Crosshaul:

- RMA – Resource Manager Application (Sections 2 and 3)
- EMMA – Energy Management and Monitoring Application (Sections 4-7)
- VIMAP - Virtual Infrastructure Manager and Planner (Section 8)
- MMA – Mobility Management Application (Section 9 and 10)
- MTA – Multi-Tenancy Application (Section 11)
- CDNMA – Content Delivery Network Management Application (Section 12)
- TVBA - TV Broadcasting Application (Section 13)

Section 14 listed the different applications which were provided as components to the experiments conducted in the various demos in WP5. Section 15 outlines the interfaces to interwork with controllers of the neighbouring network domains, i.e., the RAN controller and the Core Network controller, detailing the potential Crosshaul application requirements for such interfaces. Section 16 carries the conclusions. Further details on algorithms for various applications are contained in the Appendix (section 17).

2 Crosshaul Resource Manager Application (RMA) on joint Path Computation and Virtual Network Function Placement

The 5G-Crosshaul unified transport platform introduces the necessity of a fully integrated and unified management of fronthaul and backhaul resources in a sharable, scalable and flexible way to ensure that all users, including high-mobility ones, receive the requested services with adequate level of QoS.
To this end, Resource Management Application is developed with the goal of optimizing the 5G-Crosshaul resources, motivated by the base requirements: (i) to manage Crosshaul resources including networking, computing and storage resources in a flexible and dynamic way, (ii) to cope with the level and variation of demand expected from 5G Points of Attachment (5G PoA), (iii) to optimize the resource utilization and maximize the cost-efficiency while meeting various service requirements.

2.1 Consolidated design

The RMA provides a centralized and automated management of 5G-Crosshaul resources, to promptly provision transport services with an adequate quality while ensuring that resources are effectively utilized. Therefore, RMA provides different specialized services which are necessary to manage the resources available in the 5G-Crosshaul network at any given time and geographical location. RMA also manages the various transmission technologies which are part of the Crosshaul data plane. Indeed, different utilization can be based on network load and type of available technologies (optical fiber, mmWave, copper, etc.).

The RMA provides an efficient allocation and management of network resources in infrastructures composed of XPFEs and XPUs through two types of services: *Path Computation (PC) service*, and *Path Computation-Virtual Network Functions Placement (PC-VNFP) service.* The first refers to the service provided by this application in computing the optimal path for network flows based on specific inputs such as the identifier of a tenant network, the source and destination XPFEs, as well as the bandwidth and latency demands of the flows. In this endeavor, the RMA mainly deals with the network resources and technology types available within the 5G-Croshaul network. On the other hand, the PC-VNFP service stands for a more challenging task performed by this application: optimal path computation together with the placement of the VNFs on XPU nodes considering the network and computing resources, and technology types available within the 5G-Crosshaul network. This indicates that given specific inputs such as the identifier of a tenant network, the source and destination node IDs, as well as the bandwidth and latency demands of the flows, list of VNFs to be placed in XPU nodes, and the Forwarding Graph (FG) information which a flow requires to fulfil the specific service chain, the PC-VNFP service computes the optimal path from source to destination ensuring that the service chain requirement (i.e., FG) is satisfied. RMA provides these services to any requesting entity in the 5G-Crosshaul

network, including any other application or module of the XCI. To provide the PC-VNFP service mentioned above, RMA ought to achieve the following objectives:

1. **VNF placement:** Placement of services on computing nodes (i.e., XPUs) in such a way that service requests of a flow are fulfilled and the resource utilization is optimized. It is assumed that one VNF deployed in a XPU can serve up to a certain flow demand which, if exceeded, would require either a scale-up or creation of a new VNF.

2. **Path Computation/Flow allocation:** Computation of the optimal path for a flow in such a way that the request for a chain of services/VNFs (i.e., FG) that are deployed in the XPUs is met and resource utilization is optimized.

A possible way of approaching this complex task of joint placement of services and routing of the flows while ensuring the optimum resource utilization is through solving an optimization problem which consists of minimizing the overall cost function in terms of (i) **VNF placement**, and (ii) **Path Computation**. Since 5G-Crosshaul includes the possibility of deploying VNFs on XPU nodes and connecting XPFEs through different transmission technologies, the optimization is formulated as a problem that does the minimization shown in Eq. (1), subject to several different constraints that stand for the PC-VNFP service provided by the RMA.

$$U = \min\left(C_{VNF} + C_f + C_d\right) \ , \qquad (1)$$

where C_{VNF} is the cost associated to deploying a VNF over an XPU node; C_f denotes a fixed parametric cost and C_d denotes a dynamic parametric cost associated to a link. Costs are introduced here as a penalty that the system incurs in case of making specific decisions. Overall cost minimization clearly yields the optimal solution. All costs are unit-less and serve the purpose of describing the differences between selected transmission technologies. Details about the different cost components in Eq. (1) find the following physical explanation:

- **VNF deployment cost (C_{VNF}):** Fixed cost paid when deploying a VNF on a XPU node.

- **Fixed technology cost (C_f):** Cost associated to using a transmission technology. Particularly, a wireless transmission technology can reasonably exhibit lower fixed cost, thinking for instance to reflect simpler and quicker installation process. On the other hand, a fixed transmission technology would require larger initial effort for laying down the necessary infrastructure.

- **Dynamic technology cost (C_d):** Cost associated to the use of a specific transmission technology. This is the unit of cost paid by each flow for using

such technology, and a flow with higher bandwidth demand shall incur in larger cost. Moreover, a fixed transmission technology shall exhibit in general a lower dynamic cost, which is an expression of larger available bandwidth and higher reliability than a wireless transmission technology counterpart.

With the goal of optimizing the objective function defined in Eq. (1), (e.g., minimizing the overall cost), the RMA superintends the process of allocating the resources for the packet flows that traverse the 5G transport network by means of two resource management policies. The first policy makes the attempt to use a minimal set of network resources to serve the packet flows injected in the network provided that the required chain for VNF services of each flow is fulfilled. The second one spreads packet flows over the available network resources still fulfilling the service chain but making the attempt to balance traffic load across the links. While the decision to statically select only one policy would compromise either the overall used resources or the capacity, the motivtion to study two polices addressing opposite network requirements is to demonstrate that in a programmable network RMA can dynamically use the resource management policy that can satisfy at best different groups of services.

1) **Policy1 – PC-VNFP**
   The goal of this policy is to ensure the use of resources as efficient as possible. This translates directly into the possibility of reusing links and XPUs which have been used by flows already allocated in the network by RMA. Only when the capacity of used links and the number of VNFs deployed in XPU nodes is exceeded, other links (of different technologies) and XPUs will be used. This allows a less fragmented, and thus more efficient use of resources. Resources which are not used could be conveniently put into a low energy state and reactivated when necessary (e.g. to cope with subsequent overloading conditions).

2) **Policy2 – Load Aware PC-VNFP (LA-PC-VNFP)**
   The general conditions of this policy are similar to that for Policy1 but in addition Policy2 makes the attempt to send the flows over the links which are the least loaded in the network. This is achieved by increasing the cost of a link as its residual capacity reduces. The additonal goal of this policy is to ensure the availability of resources by avoiding the exhaustive use of specific links. To this end, a resonable and simple scheme which is used by operators for traffic engineering has been followed [1]. The idea is to perform the dynamic optimization of the link cost with the goal of reducing the maximum link capacity utilization.

The initial implementation of the RMA algorithms for PC and PC-VNFP services rely on the formulation of an equivalent ILP problem. The advantage of solving the ILP formulation consists in determining the best possible solution, if it exists. The main disadvantage incurred is a potentially long execution time to solve the ILP which makes it almost impractical to deploy in real time scenario. To this end, a heuristic algorithm

for the PC-VNFP service is developed to obtain a sub-optimal allocation of flows within an acceptable execution time ensuring the optimized use of network and compute resources. The ILP formulation of the RMA algorithms and the sub-optimal solution are detailed in the Appendix, in section 17.4.

2.2 Implementation

The RMA functions can be implemented within different modules of the 5G-Crosshaul architecture (e.g. in the XCI at the SDN controller or at the MANO level) or in a dedicated application (i.e., the RMA).

Figure *1* shows the high-level design where RMA is implemented as an application operating on top of the 5G-Crosshaul XCI SDN controller and VIM services. A description of the RMA application components shown in

Figure *1* is found in Table 1.

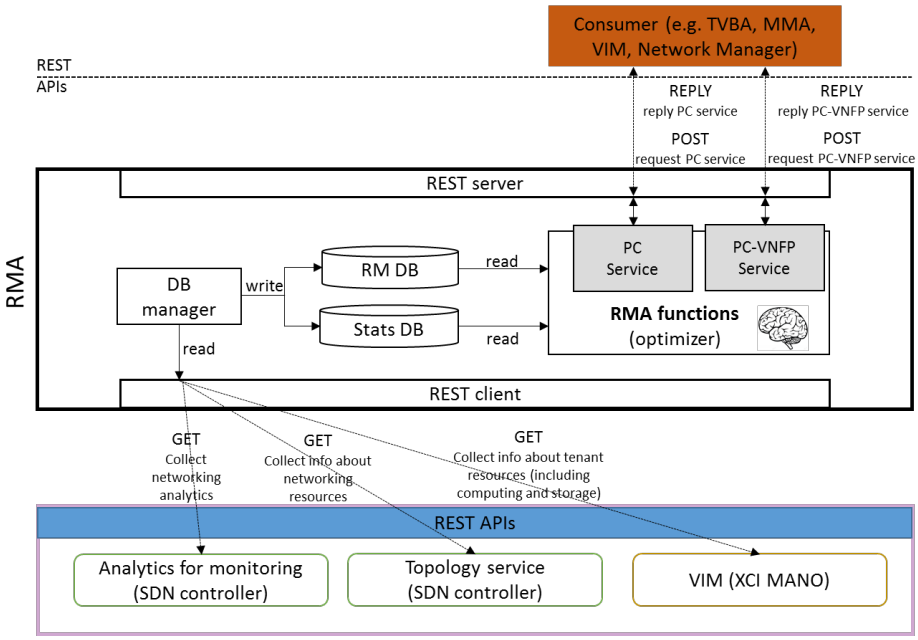


Figure 1: RMA application high-level design

Table 1: RMA application components

Component	Description
REST Client	Entity in charge of interacting with the XCI components via REST APIs.
DB manager	Entity in charge of collecting the relevant information that is needed for the resource management and optimization. Such information is collected from the various XCI modules (i.e., tenants' resources, including computation and storage, from VIM, network resources from SDN controller). The DB manager processes this information and stores it in the databases.
RM (Resource	Database that stores the inventory of 5G-Crosshaul elements (XPFEs,

Manager) DB	XPUs), the location of resources and their status, the list of existing services mapped to the resources they are using.
Stats DB	Optional database that stores the statistics related to the active services and the resources they are using.
RMA functions	The brain of the RMA application, i.e., it contains the algorithms that compute the optimum allocation of 5G-Crosshaul resources, leveraging the information stored in the Resource and Stats DBs. The RMA is developed as a passive stateful element, which answers to query issued by other applications (e.g. MMA, CDNMA, TVBA) or XCI modules (e.g. VIM) via a REST interface. Two main services are available : • Path Computation (PC) service: this service computes the allocation of network resources (given the endpoints of the network connection or the placement of the VNFs to interconnect); • Path Computation and Virtual Network Function Placement (PC-VNFP) service: this service computes the allocation of network, storage and computing resources (given the VNF forwarding graph)
REST server	Entity in charge of implementing the north bound REST API of the RMA application to allow other applications or XCI modules to request the RMA services.

The RMA application operates on physical infrastructures including software-based XPFEs and XPUs and it is developed as a python application. The application exposes a REST API towards other applications (e.g., CDNMA, TVBA) for sending path computation requests following different service constraints. The RMA relies on the XCI controllers for the actual provision and allocation of resources and can operate over physical or virtual network resources, on a per-network or a per-tenant basis. The RMA applications interacts with the XCI components, in particular the SDN controller, the VIM and the NFVO, making use of the following services and interfaces (based on REST APIs):

- SDN Controller services:
    - ability to query the topology and inventory of the network and XPU (e.g. in terms of network links, nodes, capabilities, etc. as well as computing and storage resources)
    - ability to setup and tear-down of network connections with specific paths
    - collection of resource utilization statistics
- VIM and NFVO services:
    - instantiation of virtual resources for networks, subnets and network ports
    - collection of resource utilization monitoring data for XPUs
    - collection of information about resource capabilities and availabilities in XPUs

2.3 KPIs

The following table lists the KPIs addressed by RMA:

Table 2: list of KPIs addressed by RMA

List of related project KPIs	***Obj.6. Design scalable algorithms for efficient 5G-Crosshaul resource orchestration*** *(5GPPP KPI) Increase of total 5G-Crosshaul network throughput by > 20%*		
Measurement of project KPIs as well as application specific ones	**KPI metrics (unit)**	**Description**	**Way of measurement**
	Link utilization (%)	Average utilization of links when flows are allocated. The link utilization ratio is defined as: $\eta = \frac{number\ of\ used\ l\text{inks}}{total\ number\ of\ l\text{inks}}$ The ratio η allows quantifying network resource utilization for all the links and also per technology (e.g., mm-wave, Ethernet).	Experimental assessments.
	Average link capacity utilization (normalized value)	This metric is the measure of the average link capacity used for the flow allocation. Defined as: $\frac{used\ link\ capacity}{\max link\ capacity}$ This is used for evaluating and comparing the link utilization in diverse resource management policies developed in RMA. For example, in case of Policy1 where the objective is to avoid fragmentation in resource utilization, a higher value of the ratio indicates the efficiency in achieving the objectives.	Experimental assessments.

	Average path length	Average length of the path for n allocated flows	Experimental assessments
	Service utilization (%)	This metric is used for the evaluation of efficient placement and utilization of VNFs in the XPU nodes. This metric includes : a) Placement (%) of VNFs in XPUs b) Reuse (%) of VNFs in XPUs Lower number of placment and higher number of reuse of VNFs ensures a saving, thus reducing total cost in resource management.	Experimental assessments.
	Parametric Cost	Cost to perform the optimal path computation together with the placement of the VNFs on XPU nodes considering the network resources and technology types. $cost = path\ computation\ cost + VNF\ placement\ cost$ This metric is used to compare the efficiency of RMA with the state-of-the-art solution.	Experimental assessments
	Computation time (min)	Time required for the resource management. This metric is used to validate the scalability of the RMA algorithm in resource optimization.	Experimental assessments
	Blocking probability of flow request	This KPI is used for the evaluation of the blocking probability of the flows in the context of limited network resources	Experimental assessments.

List of benchmark approaches **(used to compare with proposed solutions)**	1. Benchmark approach: Shortest Path algorithm for traffic routing and manualalgorithm for provisioning of NFVs Evaluate the resource optimization solution with the context where no resource management solution is applied, in particular, we compare our solution in the context where traffic steering is performed based on shortest path algorithms and manual/greedy algorithm for provisioning of network services in the processing nodes. 2. Benchmark approach: k-shortest path algorithm The sub-optimal solution of Resource Management Application is compared with the solution where resource management solution is not applied, in particular, the RMA is compared with the solution where k-shortest path algorithm is used for traffic steering and greedy algorithm is used for NFVs placement in the processing nodes.

2.4 Validation and Evaluation

2.4.1 Evaluation Environment and Scenario

An analytical and simulative approach is chosen as main methodology for the evaluation of the RMA algorithms. Matlab simulation environment is used for this purpose.

The functional validation for RMA in a real test-bed will be performed in WP5 [2] under demo 2 and will be reported in D5.2. The focus of this integrated demonstration is on the distribution of media content over the 5G-Crosshaul infrastructure. Here, the application plane includes four applications (TVBA, MMA, RMA and CDNMA) and the control plane the elements from the Crosshaul Control Infrastructure (XCI), the SDN controller, the VIM, the VNFM and the NFV orchestrator. The evaluation of RMA will be measured by the integration of RMA on the 5TONIC testbed and use cases involving the RMA satisfying the path computation requests of other applications (such as CDNMA and TVBA) by provisioning optimal paths.
This section summarizes the simulative evaluation of the RMA algorithms for PC-VNFP service. The following two scenarios are considered:

- Optimal solution for managing network and computing resources - The joint task of placement of services and routing of the flows while ensuring the optimum resource utilization is defined as an optimization problem and is formulated as an equivalent Integer Linear Programming (ILP) problem. This is performed using Matlab simulation environment, and in particular using the Matlab optimization toolbox for solving Mixed ILP (MILP).
- Sub-optimal solution for managing network and computing resources - The validation of the heuristic solution to find a sub-optimal solution for resource management in the reasonable time frame is performed through the Matlab simulation environment.

2.4.2 Test Cases for functional validation

The objective of the validation of the algorithms is to evaluate the effectiveness of the RMA algorithms in finding the efficient placement of network services and appropriate routing of flows that ensures minimum use of network resources and cost. With above objective, three types of tests are performed including the simultaneous allocation of flows (i.e. All Flows Allocation), the back-to-back/consecutive allocation of flows (Flow-by-Flow Allocation), and the testing of a suitable heuristic for managing resources to run in polynomial time. A short description of these experiments is found in Table 3. For the details on test cards, see [3].

Table 3: RMA algorithm for PC-VNFP service test card

Test Algorithm	CREATE-NET_PCVNFP _01	Execution Status	Passed
Test Name	Flow-by-Flow Allocation		
Objectives	Optimise use of network and compute resources through minimization of the cost function for voice and video traffic		
Test Algorithm	CREATE-NET_PCVNFP _02	Execution Status	Passed
Test Name	ALL Flow Allocation		
Objectives	Optimise use of network and compute resources through minimization of the cost function for voice and video traffic		
Test Algorithm	CREATE-NET_PCVNFP _03	Execution Status	Passed
Test Name	Heuristic for Flow Allocation		

2.4.3 Evaluation Result

To study the RMA, we assume that the heterogeneous transport system is composed of mm-Wave and ethernet links that essentially interconnect computing nodes (e.g., XPUs) hosting different VNFs through intelligent switching elements (e.g., XPFE). Indeed, mm-Wave technology can result extremely useful whenever the deployment of infrastructures is considered too expensive. Thus, in the simulation environment, each XPFE element is assumed to embed a dual type of transmission technology (e.g., mm-wave and Ethernet) to connect to another switching element. In this simulation environment, 3GPP propagation model at 2.5 GHz is used for modeling mm-wave links. Detail about the modeling of mm-wave link is described in [3]. Each transimission technology is associated with a fixed and a dynamic cost. As anticipated, the fixed cost of Ethernet is chosen to be higher than that of mmWave, but this is exactly the opposite for the dynamic cost. For dynamic cost, this is quantified whereby the link success

probability $p_s \in [0,1]$. For Ethernet it is assumed $p_s^{(eth)} = 1$, whereas $p_s^{(mmwave)} \leq 1$. The dynamic cost function is computed as $C_d = 1/p_s$ irrespective of the selected technology.

In the simulation scenario, several random network topologies are generated where the deployment of switching and computing nodes (i.e., XPFEs and XPUs) is as generic as possible. Here, it is assumed that, each XPU node is equipped with a number of cores, with one core fully dedicated to one virtual machine (VM), and a VM hosting one VNF. It is also assumed that one core can serve up to a certain flow demand which, if exceeded, would require either VM scale-up or creation of a new VM for that VNF, within the same or in another XPU. A fixed cost is considered for deployment of a VNF on a XPU node. All costs are unit-less.

For the validaiton, allocation of resources to a set of flows has been considered, with each flow having different throughput and latency demand in the dual technology network (i.e., Ethernet and mm-Wave) and having the request to fulfill a specific forwarding graph (FG) (e.g., FG:$s_1 \rightarrow s_3 \rightarrow s_8$ indicates that the flow needs to traverse the XPUs hosting services s_1, s_3, and s_8 in given order before reaching the destination node). Generation of flow requests is performed through a random process (generating random bandwidth and latency demand). For each flow requiest, the RMA algorithm determines the optimum path from source to destination node, which can fulfill the bandwidth and latency demands, as well as the constraint to traverse services inside the FG. To accomplish this objective, the RMA algorithm can select any of the technological options available for transmission (i.e. either Ethernet or mm-Wave). For any given number of nodes and flows to allocate, flow ingress (source node) and flow egress (destination node) are selected at random, as well as the FG for each flow. Network topology remains unchanged as well as the source-destination pair and FG but simulations are repeated randomizing the order in which flows are allocated in the network. The randomization leads to different selection of computing nodes where to deploy services and switching elements to traverse. In other words, this yields different allocation strategy results.

Here, the effectiveness of RMA PC-VNFP service in finding the appropriate routing for each flow jointly with the efficient placement of VNFs is evaluated. Specifically, we focus on investigating the contribution of RMA in the following KPIs to meet the project objectives:

- Increase of total 5G-Crosshaul network throughput by > 20%
- Scalable management framework: algorithms that can support 10 times increased node densities

To this end we evaluate the performance metrics (listed in section 2.3) obtained as a result of the experimental tests of the RMA algorithms.

Numerical values used for the test cases to validate RMA are shown in Table 4.

Table 4: Numerical parameters used to simulate the RMA algorithm for PC-VNFP

Parameter	Value
Geographical area	[2000 × 2000] m
Coverage radius of a mmWave transmitter	≤ 200 m
Total number of simulations for given number of flows and nodes	50
Total number of nodes (XPU+XPFE)	$10 \leq n \leq 100$
Percentage of XPUs	40%
Percentage of XPFEs	60%
Number of flows	$5 \leq f \leq 1000$
Percentage of Video flows	70%
Percentage of Audio flows	30%
Mean and Standard dev of demand for video traffic	5 Mbit/s, 1
Mean and Standard dev of demand for voice traffic	1 Mbit/s, 0.2
Latency constraint for video traffic	<100 ms
Latency constraint for voice traffic	<10 ms
Fixed cost [Eth, mmwave]	[100, 1]
Dynamic cost	$1/p_s$
Capacity [Eth, mmwave]	[10000, 2000] Mbit/s
Fixed cost for deploying a VNF in a XPU core	1000
Core per XPU	4

The experimental results obtained from the test cases are presented in the following sub-sections.

2.4.3.1 *Optimal solution with consecutive and simulataneous flow allocation*

The experimental evaluations of RMA PC-VNFP service obtained as a result of solving the equivalent ILP formulation when flows are allocated in consecutive fashion (test card - CREATE-NET_PCVNFP _01) and simultaneous fashion (test card – CREATE-NET_PCVNFP_02) are presented here.

In case of simultaneous allocation (e.g., all flow allocation, test card – CREATE-NET_PCVNFP_02), the numerical values used to obtain results are almost identical as those shown in Table 4, except in this case the number of flow, f used for the experiment is $5 \leq f \leq 65$ and with number of nodes 15. Figure 2 shows the utilization of different link technologies over the overall available links in the network, while increasing the total number of flows with a number of nodes in the graph equal to 15. The figure shows the overall utilization of links in the network upon the same conditions. It can be noticed that in correspondence of 65 flows to allocate, the overall utilization of network resources remains below 50%.

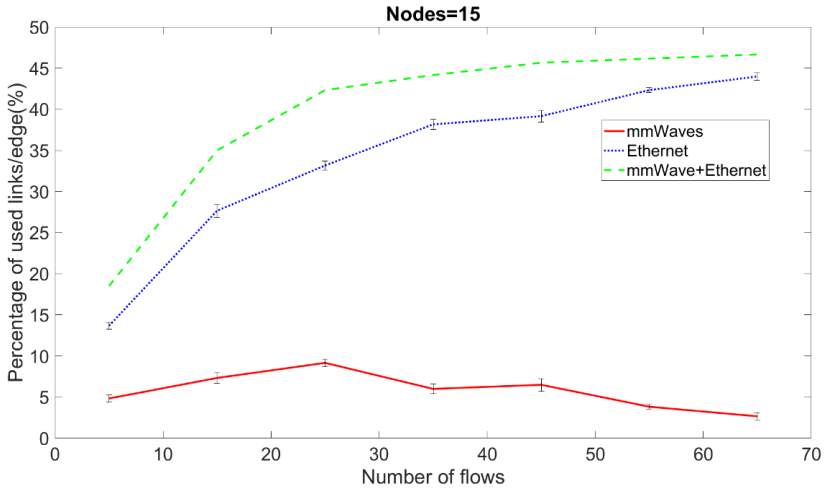


Figure 2: Percentage of the total number of links used in a network with 15 nodes (XPUs+XPFEs) deployed, while varying the number of flows to be allocated in the network on all Flow allocation basis

In the second experiments (test card - CREATE-NET_PCVNFP _01), when the flows are allocated in consequitive fashoin, two different resource management policies that address opposite network requirements have been studied. The first policy policy (named 'PC-VNFP') tries to exploit the links already used as much as possible for the purpose of avoiding any waste. The second one (name 'LA-PC-VNFP') does distribute the load across the whole transport network to improve and maintain resources availability. The two resource management policies are validated through extensive simulations and comparison between them is presented. The first aims to reduce resource utilization upon serving all the packet flows injected, whereas the second spreads the traffic to avoid overloading conditions over the integrated fronthaul-backhaul network.

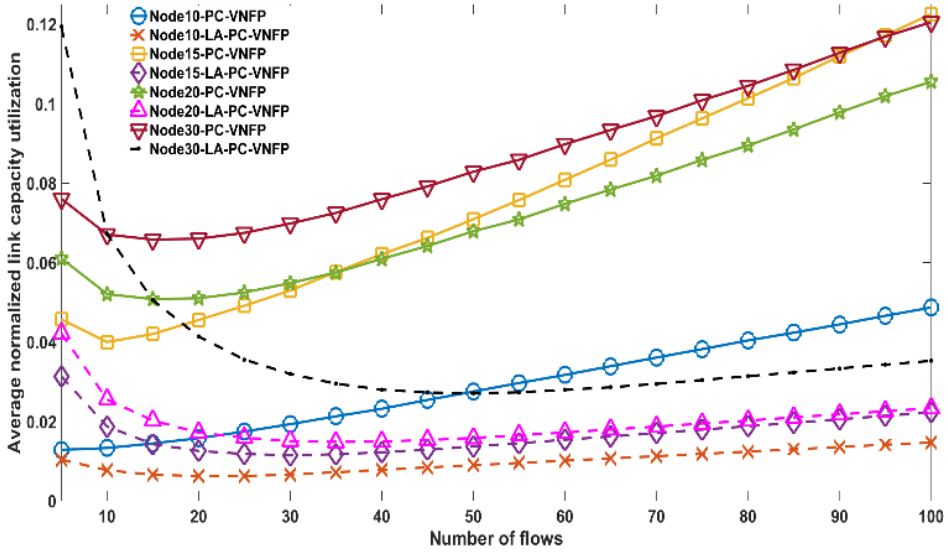


Figure 3:Average normalized link capacity utilization vs. the number of flows to be allocated in the network

Figure 3 shows that the average normalized link capacity used in Policy1 ('PC-VNFP') is higher than in Policy2 ('LA-PC-VNFP'), indicating that Policy1 tries to exploit as much as possible the capacity of fewer links, whereas Policy2 reduces link resource utilization more effectively.

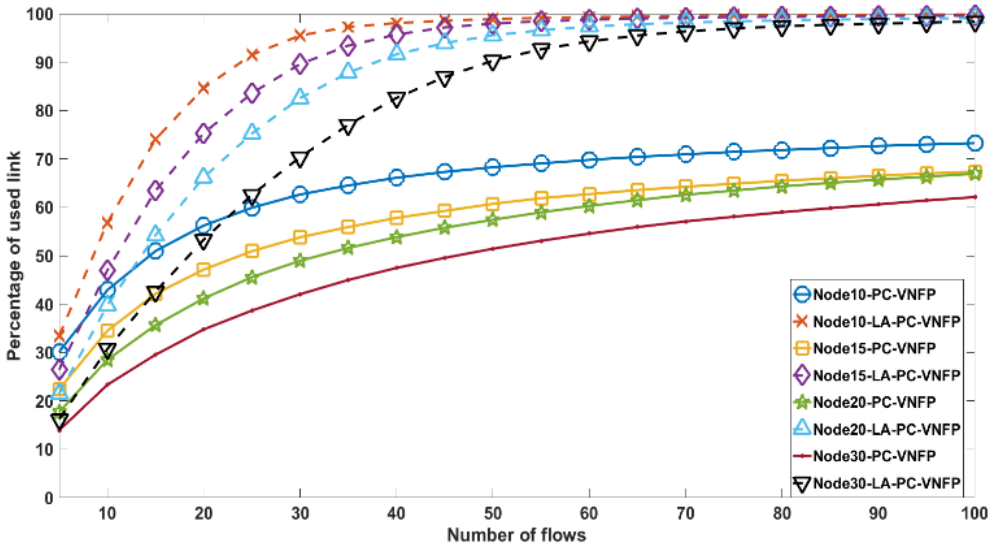


Figure 4: Percentage of the total number of links used in a network vs. the number of flows to be allocated in the network

On other hand, Figure 4 shows that for Policy2 ('LA-PC-VNFP') the percentage of used links is a way higher than for Policy1 ('PC-VNFP'), which proves that being Policy2 load-aware, it makes the attempt to spread traffic over the network. These results allow us to preliminary conclude that RMA can satisfy at best different network requirements still serving all packet flows, such as, 'Policy1–PC-VNFP' to avoid the fragmentation in resources allocation, and 'Policy2-LA-PC-VNFP' for a load-aware resource allocation over the network.

2.4.3.2 *Sub-optimal solution with consecutive flow allocation*

The validation results of the heuristic algorithm (test case - CREATE-NET_PCVNFP _03) providing RMA PC-VNFP services is presented here. To show the efficiency of the heuristic solution in performing resource optimization in reasonable time, a comparison of the execution time between the ILP solution and heuristic solution of RMA PC-VNFP services when the flows (here, the number of flows is 100) are allocated in consecutive fashion is shown in Figure 5(a). In particular, the execution time scales exponentially with the number of nodes due to the fact that the ILP must compute a larger number of possible combinations of possible paths in order find the optimum solution for each flow. Figure 5(b) shows that the sub-optimal heuristic solution allows to run experiments also for large instances of the problem within an acceptable execution time (here, $O(n^2)$). Each simulation runs on a single Intel i7 core operating at 2.6 GHz.

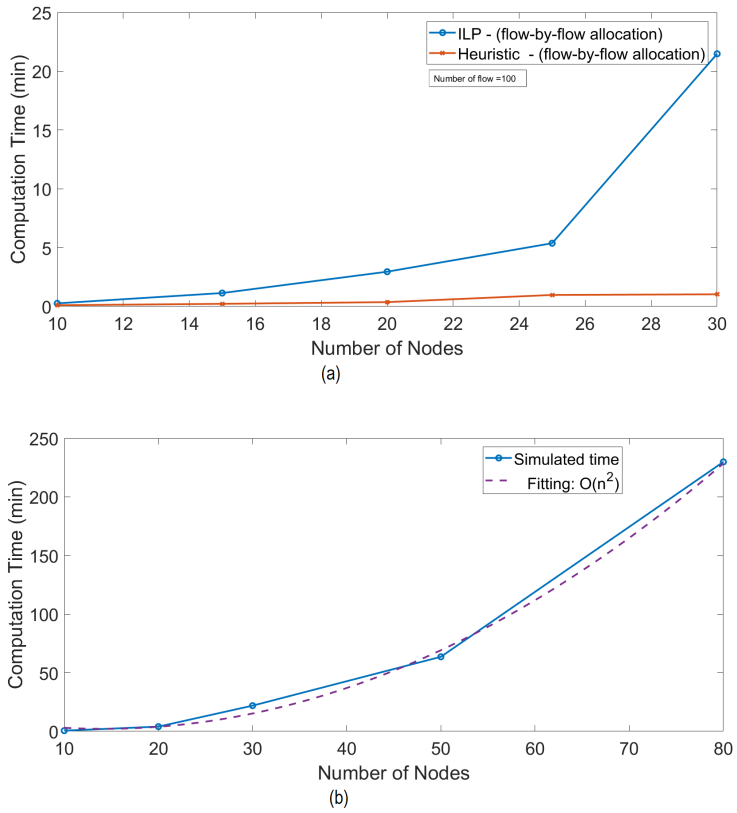


Figure 5:(a) Comparison of time to optimize resources for the ILP and the heuristic solution when flows are allocated consecutive fashion (flow-by-flow) with varying the number of nodes and flows, (b) computation time for the heuristic solution with increasing number of nodes

Figure 6 shows that the average normalized link capacity in the heuristic solution, while increasing the total number of flows and nodes. The average link capacity utilization is below 10% which allows to conclude that the objective of avoiding fragmentation in resource utilization has been achieved, compatibly with the compound demand of flows.

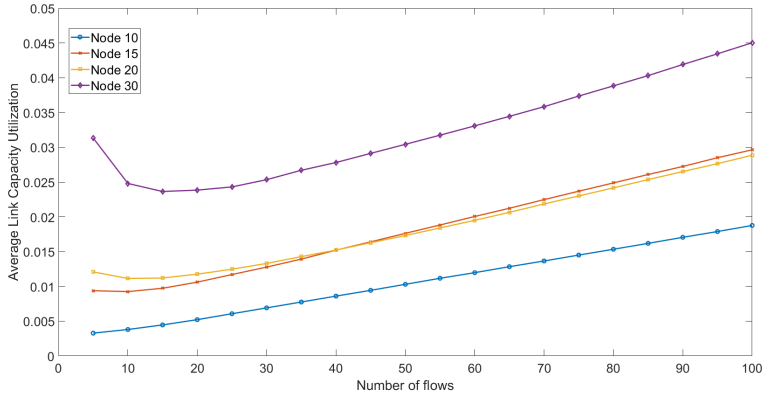


Figure 6: Average normalized link capacity utilization vs. the number of flows to be allocated in the network for sub-optimal solution

We compare the heuristic solution for RMA PC-VNFP service with the state-of-the-art solution that is done in practice where traffic/flow steering among the end points is

performed following k-shortest path solution and a greedy approach is adopted for allocating the VNFs in the XPUs. While, the heuristic solutions for RMA uses a variation of the standard k-shortest path solution where link cost assigned per technology (e.g., mm-wave and Ethernet) is used as the weight function instead of the link/hop-count parameter. The RMA searches for the optimal placement of VNFs with the goal of minimizing the placement cost.

Figure 7 shows that the RMA PC-VNFP solution reduces the resource management cost significantly compared to the state-of-the-art approach. It is noticed that the cost saving increases with the increase of the network size, for example, in case of a network with 30 nodes (40% XPUs and 60% XPFE nodes) RMA reduces the cost up to 90%.

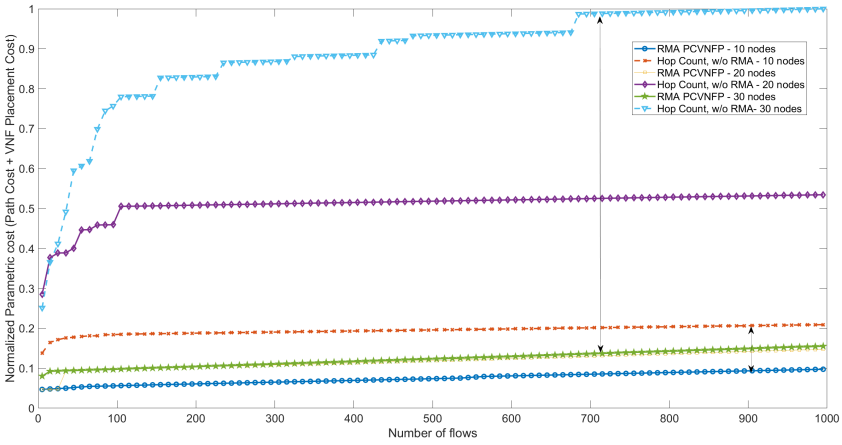


Figure 7: Comparison of cost among the RMA PCVNFP and the SoA solution where path computation with k-shortest path and greedy algorithm for VNF placement

2.4.4 Summary

To summarize, extensive simulations of the RMA PC-VNFP service show evidence that a heterogenous set of network technologies can be managed following different optimization criteria, and a 5G-Crosshaul software-defined network could intelligently make use of the most effective policy depending on the specific traffic conditions and VNFs chain specification.

3 Crosshaul Resource Manager Application (RMA) on joint Routing and C-RAN Functional Splits

The main requirement in the path towards 5G systems is to increase the centralization degree of RAN functionality. However, due to the stringent requirements of current fronthaul solutions, classical C-RAN technology is deemed unfeasible in many realistic scenarios for 5G and flexible RAN functional splits have been defined to enable a smoother migration. The idea of flexible RAN functional split is to divide a classic BS into a set of functions that can either be processed co-located with the radio unit (RU) or offloaded into a central unit (CU), depending on the transport requirements and centralization needs. Different RAN functional splits and their requirements have been studied by the Small Cell Forum and 3GPP, as summarized in Table 5.

To this end, the next generation Cloud-RAN faces a significant problem. That is, how to decide the functional splits and transport resources jointly in order to better balance cost/performance (the more aggressive we offload the BS functions to the XPU, the higher are the gains) and tough requirements (the less BS functions are offloaded to the XPU, the more relaxed the network constraints are) which are subject to more limited - and likely shared -transport resources. *Our aim is retaining as much centralization degree as possible when full offloading of BS functionality is unfeasible due to transport constraints.* Therefore, we define a new metric "**degree of centralization**" as our objective. The higher the centralization degree, the higher are the gains regarding OpEx and CapEx reduction as well as higher spectrum efficiency. To solve this problem, we develop this application to jointly route traffic across fronthaul/backhaul (i.e., Crosshaul) transport networks and select proper functional splits for each BS to maximize the degree of centralization while meeting next generation fronthaul network constraints on parameters such as capacity, latency, and jitter.

Table 5: Functional splits analysis in small cell forum. 1 user/TTI, 20 MHz BW, IP MTU 1500B; DL: MCS 28, 2x2 MIMO, 100RBs, 2 TBs of 75376 bits/subframe, CFI = 1; UP: MCS 23, 1x2 SIMO, 96 RBs, 1 TB of 48936 bits/subframe

Split #	BS Functional decomposition	DL/UL BW req. (Mb/s)	Delay req. (μs)	Gains
A	RRC - PDCP	151/48	30e3	• Enables L3 functionality for multiple small cells to use the same HW; • Enhanced mobility across nodes w/o inter-small cell data forwarding/signaling; • Reduced mobility-related signaling to the mobile core segment; • No X2 endpoints between small cells and macro eNBs; • Control plane and user plane separation.
B	PDCP - RLC	151/48	30e3	• Enables L3 and some L2 functionality to use the same HW.
C	RLC - MAC	151/48	6e3	• Resource sharing benefits for both storage and processor utilization.
D	MAC I - MAC II	151/49	6e3	• Synchronized coordination and control of multiple cells; • Coordination across cells enables CA, CoMP, eICIC or cross carrier scheduling.
E	MAC - PHY	152/49	250	• Enhancements to CoMP with RU frame alignment and centralized HARQ.
F	PHY split I	173/452	250	• More opportunities to disable parts of the CU at quiet times to save power; • Central L1 CU can be scaled based on average utilisation across all cells; • Smaller CU results in less processing resource and power saving; • Enhancements to joint reception CoMP with uplink PHY level combining.
G	PHY split II	933/903	250	
H	PHY split III	1075/922	250	
I	PHY split IIIb	1966/1966	250	
J	PHY split IV	2457.6/2457.6	250	

3.1.1 Problem Statement

Despite the interest of Crosshaul architectures for 5G, to our knowledge, no study has looked into the implications that a flexible RAN centralization poses on path computation with high path diversity. In fact, despite its benefits, a joint implementation of both, flexible RAN centralization and a Crosshaul path computation, is inherently challenging. On the one hand, a proper choice of BS split points depends on the transport capabilities of the Crosshaul network, like available bandwidth, latency or jitter, which in turn is unknown until all paths have been computed (note that in Crosshaul, links can be shared). On the other hand, an adequate routing across the Crosshaul requires knowledge of the BS split points, because such choices set the demands of the flows to route. Therefore, we face a coupled problem where routing and selection of RAN splits must be optimized jointly, a problem that to the best of our knowledge has not been identified before.

To illustrate this, let us set up a simple scenario, depicted in Figure 8, with one XPU and 5 RUs. 4 RUs are located within the same interference domain A, whereas the remaining RU is located in another interference domain B. In this example we consider only capacity constraints for simplicity. We now analyze different strategies to optimize our toy example.

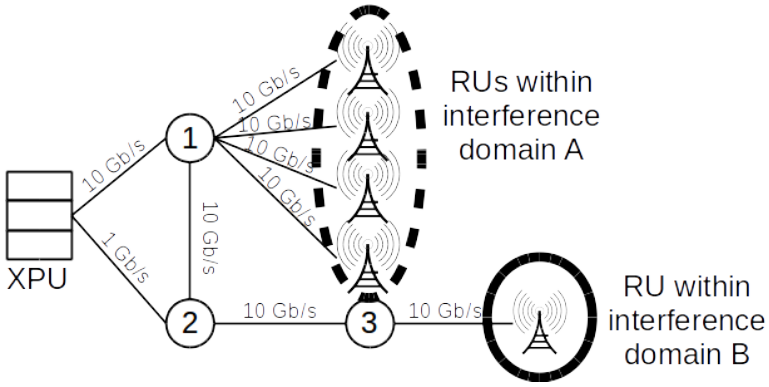


Figure 8: simple scenario

Solution 1: Best split, then best routing

The first strategy is to decide first on the RAN split---without routing knowledge---and then, based on such decision, to find a feasible set of paths. Given a split, the problem can be seen as an Unsplittable Flow Problem (UFP) which is NP-Hard. Regarding Figure 8, we may be tempted to pick split J (Table 5) for all BSs to grasp the advantages of full centralization. This choice, however, implies that five 2.5-Gb/s flows shall be routed from the XPU to each RU, which is unfeasible (this is obvious in such a simple example). It thus becomes evident that a (mildly) better approach is to set the paths first and then, based on the transport capabilities given by these routes, find a good feasible RAN split, which leads to our next strategy.

Solution 2: Best routing, then best split

The second strategy is to decide on routing first, based on some criteria (e.g. shortest-path), with no knowledge of each flow demand (which highly depends on the RAN split). For instance, if we apply a max-min criterion, the outcome would be five 2-Gb/s flows. Now, upon such routing knowledge, we could find the highest feasible RAN split. This is also a NP-Hard problem since it could be seen as a BAG-UFP problem in a tree which, to our knowledge, does not have approximation algorithms. However, given the simplicity of our example, we can find, via exploration, that the solution with highest centralization degree is split I for all BSs (Table 5.). However, as we show next, we can still do better with a joint solution.

Solution 3: Joint decision

We can observe that one RU is in a different (smaller) interference domain B, which implies lower gains in centralization (no coordination issues with other BSs). With this in mind, we could route the traffic of the RU in domain B through path XPU-2-3 using split G. This releases additional capacity in path XPU-1 for the remaining RUs which can now use split J, maximizing in this way the advantages of centralization of the whole system. Reaching this solution implies a joint routing/RAN split selection which has not been studied before. Of course, this problem is also NP-hard and becomes very complex in large-scale scenarios.

3.1.2 Our goal

Despite the enormous interest in the design of NGFI, driven by industry and standardization (NGMN , 3GPP , IEEE , Small-Cell forum), to the best of our knowledge, we are the first to formalize a new problem arising of such flexible fronthaul: the coupled problem of optimally taking routing and BS function placement decisions. The above analysis makes evident that functional split selection complicates an already hard problem (QoS routing) and thus traditional solutions do not apply here. *In light of this, we develop algorithmic solutions to this new problem, conveniently wrapped into a decision-making engine named **WizHaul**, that serve two purposes*:

- **Network planning.** WizHaul solves a problem that is essential for any Crosshaul planning tool: that of deciding the optimal BS functional split and link provisioning between CUs and RUs.

- **Fault-tolerance.** WizHaul is a centralized decision engine that can be integrated in Software-Defined Network (SDN)-based platforms to support fault-tolerance in a more dynamic fashion.[1]

The latter is useful when link changes occur. For instance, an operator may make a configuration choice based on measurements available at a planning phase. However,

[1] The timescale of these configuration changes is days or even weeks, and thus we do not envision functional split changes adapting to fast events such as, e.g., mobility of single users or wireless fading.

interfering links may appear with time, possibly rendering the original choice invalid. RAN cloudification, already demonstrated in the past and realized in some early commercial solutions enables these new use cases.

3.1.3 Related Work

QoS routing is one of the most fundamental problems of networking. Hence, though it has been a matter of research for over two decades, it is still under study. The most similar problem is the multi-commodity unsplittable flow problem (UFP). This problem routes multiple flows using single paths subject to network constraints. There are a few versions depending on the objective [4]. **Max-EDP** (maximum edge-disjoint path problem) aims to route a subset of flows across disjoint paths maximizing a profit function [5]. **Max-UFP** generalizes it allowing links to be shared [6]. Round-UFP (UFP with rounds) partitions the set of flows into the minimum number of subsets with feasible routing instances [7]. **Bag-UFP** (UFP with bag constraints) divides all the flows in bags and finds the subset of flows of maximum profit such that each flow belongs to a different bag [8]. Though most of the algorithms exploit particular graph structures or make the no-bottleneck assumption (no link has less capacity than the highest demand), some works focus on general graphs [9]. **Round-UFP** is NP-Hard as it contains the Bin-Packing problem. **Max-UFP** and **Bag-UFP** contain the Knapsack problem as a particular case and thus they are NP-Hard too. Another related problem is the **Virtual Network Embedding (VNE)** problem and the problem of placing chains of virtual functions [10]. In our problem, however, (i) we must find paths for all RUs (all flows must be admitted), (ii) the flow demands are specified by another variable (splits), and (iii) both delay and bandwidth requirements must be met.

The problem of routing video flows while optimizing their coding rate (which define the network demands, like BS splits in our case) can also be related. However, given the short timescale in which decisions shall be taken, the related work focus on multipath routing (to exploit coding diversity), single flows and/or make topology simplifications to reduce complexity.

A lot of work on C-RAN has been published lately [11] [12]. An early study of the feasibility of pure C-RAN in packet-based networks is presented in [13]. The authors concluded that frame preemption (802.1Qbu), scheduled traffic (802.1Qbv) and buffers in the receivers would be needed. The work of [14] was the first to propose BS functional split points different from pure C-RAN to relax constraints while retaining some centralization degree. Some other work has studied the trade-offs between qualitative benefits and quantitative costs of different splits for simple scenarios [15] and, recently, the impact of packetization and scheduling in simple setups [16].

Though all this work shows that RAN centralization has many advantages, we are, to the best of our knowledge, the first to study the implications that a flexible BS functional split poses on path computation in large-scale packet-based networks.

3.2 Consolidated design

The architectural software design of RMA is depicted in Figure 9. RMA operates as an application on top of an SDN controller as shown in the figure. Details on the mathematical modelling and algorithmic solutions implemented in RMA can be found in Appendix 17.2. RMA communicates with an SDN controller within the XCI through a REST-based NBI. In this way, RMA can gather topological information from the data plane (using the inventory service and the analytics for monitoring service) and can enforce the computed configuration via the network re-configuration and path provisioning services.

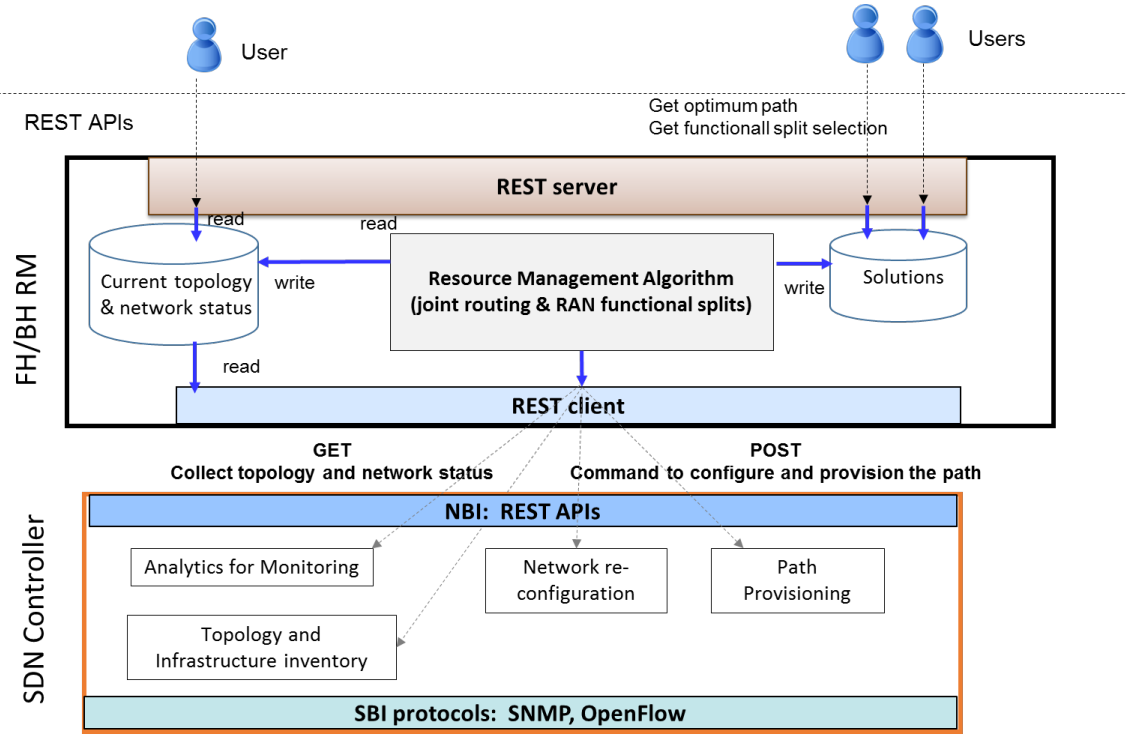


Figure 9: Software architecture design

3.3 Implementation

Figure 10 depicts an illustration of our software implementation. We implement our algorithms as an application on top of Floodlight, a Java-based Apache-licensed SDN controller[2]. We chose this SDN controller to demonstrate the RMA in a small-scale stand-alone test-bed (section 3.5.1) within the scope of WP4, while this RMA is also provided as an application component in WP5 integrated with other SDN controllers under demo 3 (hierarchical multi-domain resource management of 5G-Crosshaul) and will be reported in D5.2. The RMA is developed as an application plugin which does not depend on the underlying SDN controllers. In the initialization phase, the application retrieves (through a REST-based interface) a graph abstraction of the physical topology using Floodlight's Topology Manager which in turn uses LLDP

[2] http://www.projectfloodlight.org/floodlight/

discovery protocol. Then, our application uses the algorithm described in Section 17.2 of the Appendix to jointly compute the optimal paths between radio access points, vEPC and CU (when needed) and the functional split of capable BSs. Upon topology changes, e.g. a link failure or a change on the modulation of a wireless link, our application receives a notification from the SDN controller (which in turn receives a notification from the hardware components via SNMP) and a new configuration is computed (and enforced by the SDN controller via SNMP and OpenFlow for path setup). As depicted in the figure, the different key architectural components of our software application are the following. The HTTP management class is used to create different HTTP request objects that will be used to communicate with the controller's REST API. The bodies of the HTTP methods are filled out with JSON data objects. The Communications management class enqueues all the HTTP objects in a FIFO queue and are processed by several threads associated to different callback functions in charge of managing replies of requested objects. The Crosshaul controller class provides the different data structures to maintain the network state and implement the logic of our algorithms. The Manager class manages the callbacks upon notifications from the SDN controller, and coordinates the above classes for the recovery process (i.e. computation of a new configuration) while it maintains updated data structures and network state information. We rely on existing SDN services in Floodlight (shown in Figure 10) for path provisioning, topology and monitoring information, as well as an extended service to let us (re)configure the CU/RU split in our equipment.

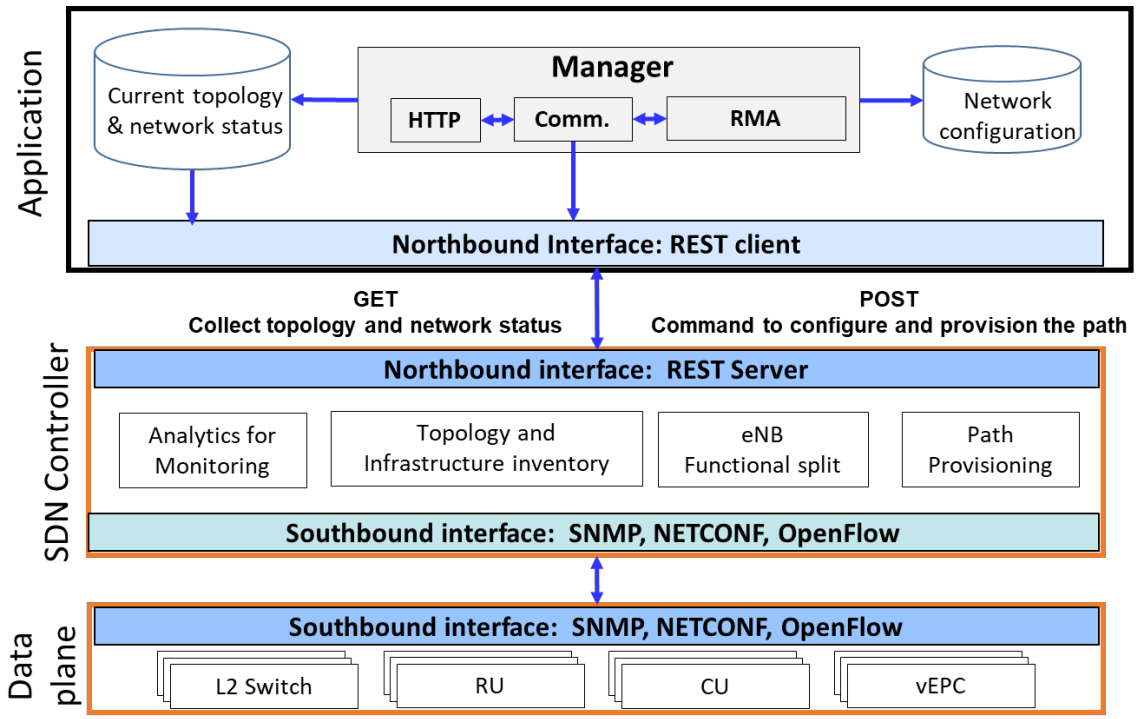


Figure 10: RMA software implementation.

3.4 KPIs

Table 6: list of KPIs addressed by RMA

Application Name	***RMA on joint routing and C-RAN functional splits for 5G-Crosshaul***		
List of related project KPIs	***Obj.6. Design scalable algorithms for efficient 5G-Crosshaul resource orchestration*** *(5GPPP KPI) Increase of total 5G-Crosshaul network throughput by > 20%*		
Measurement of project KPIs as well as application specific ones	**KPI metrics (unit)**	**Description**	**Way of measurement**
	Degree of centralization	This metric is used to account for how much centralized Base Station functions are. The higher the centralization degree, the higher are the gains in terms of OpEx and CaPEx reduction as well as higher spectrum efficiency.	Simulation results (Experimental PoC)
	Throughput (Mb/s)	Throughput requirements (of functional splits) are satisfied.	Simulation results (Experimental PoC)
	Delay (usec)	Delay requirements (of functional splits) are satisfied.	Simulation results (Experimental PoC)
	Signaling latency (usec)	Delay to react upon changing network conditions in the control plane (XCI)	Experimental PoC

List of the State-of-the-Are approach **(used to compare with proposed solutions)**	Baseline approaches based on "first route, then find highest functional splits" based on: - Shortest path routing - Max-Flow routing - Max-Min routing State of the art routing approaches fail to trade off the centralization degree of some flows to benefit larger clusters of Base Stations. This is because they all have simpler goals, e.g., Shortest-Path aims to find low-latency paths without favoring disjoint paths; Max-Flow targets high-capacity paths irrespective of their distance (latency) towards an XPU; Max-Min focuses on both capacity and fairness across RRHs, without balancing centralization towards larger clusters.

3.5 Validation and Evaluation

3.5.1 Validation

We start with an experimental validation in a small-scale testbed and a fault-tolerance use case.

3.5.1.1 *Testbed*

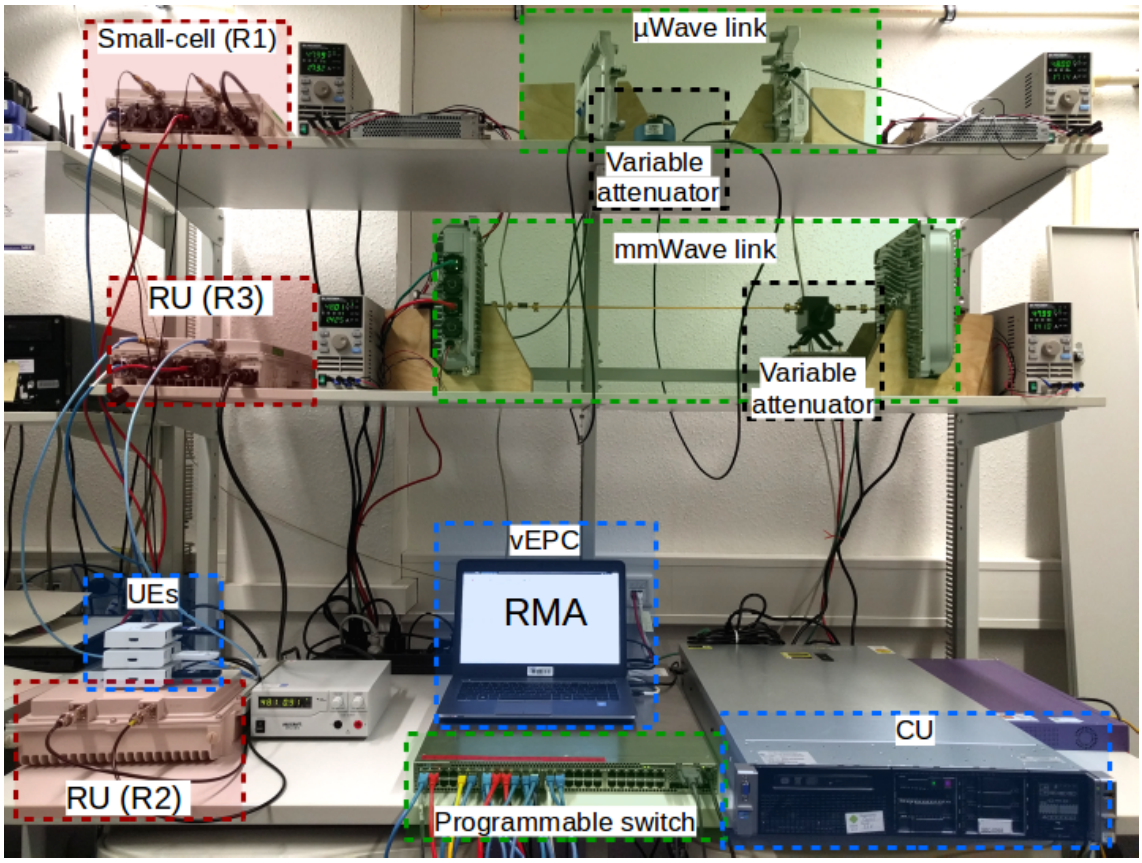


Figure 11: Testbed for RMA validation.

We deploy the testbed shown in Figure 11, illustratively described as a baseline scenario in Figure 12. We use two commercial wireless products in the BH/FH segment, namely a µWave link (connecting nodes 3-5 in Figure 12) and an mmWave link (connecting nodes 4-7). Both support adaptive modulation schemes so the actual capacity varies with the average signal-to-noise ratio (a common measure of wireless channel conditions or quality). For demonstration purposes, the radio links are wired with an SMA cable (µWave RF over coaxial) and a rigid waveguide (E-band RF). Moreover, variable attenuators let us emulate different average channel conditions. The remaining topology of the transport segment is virtualized using a programmable switch via gigabit Ethernet interfaces. Also, a software virtual EPC is deployed on a commodity laptop. The RAN equipment is comprised of R1, a fully-fledged LTE small-cell (i.e., its base station functions are not centralized), and two RUs (R2 and R3) connected to a CU. Given that we do not have yet the ability to configure all functional splits from Table 5 in our CU/RU hardware, we derive traffic flow patterns based on a commercial product, namely a flexible C-RAN solution with available splits 1 and 3 from Table 7, and generate UDP flows accordingly. Table 8 provides more details of the HW components of our testbed. All elements are connected across the same L2 Ethernet-based domain and are synchronized with PTP, which allows us to measure one-way delay.

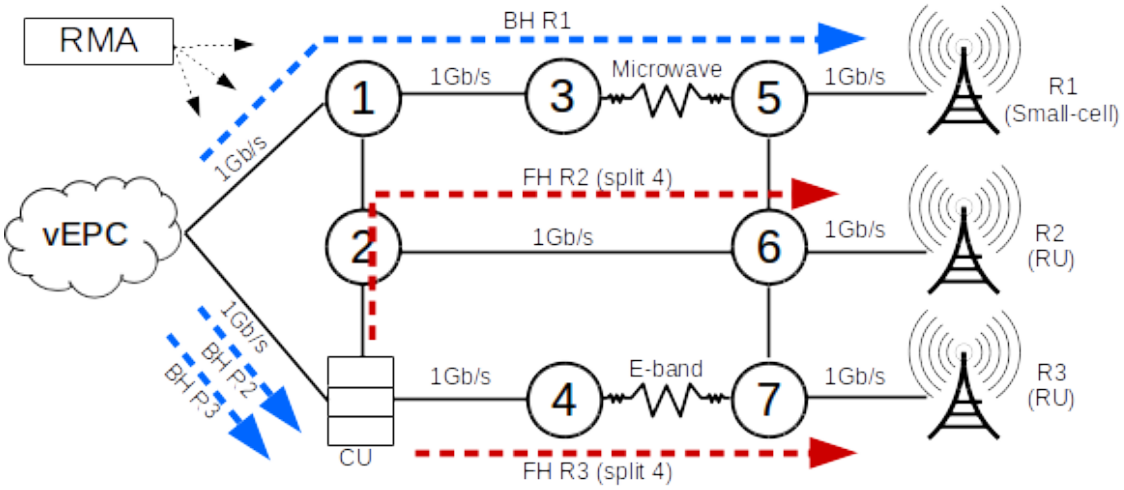


Figure 12: Baseline RMA PoC scenario.

Table 7: Computational costs based on the CPU execution times and split mapping to Table 5.

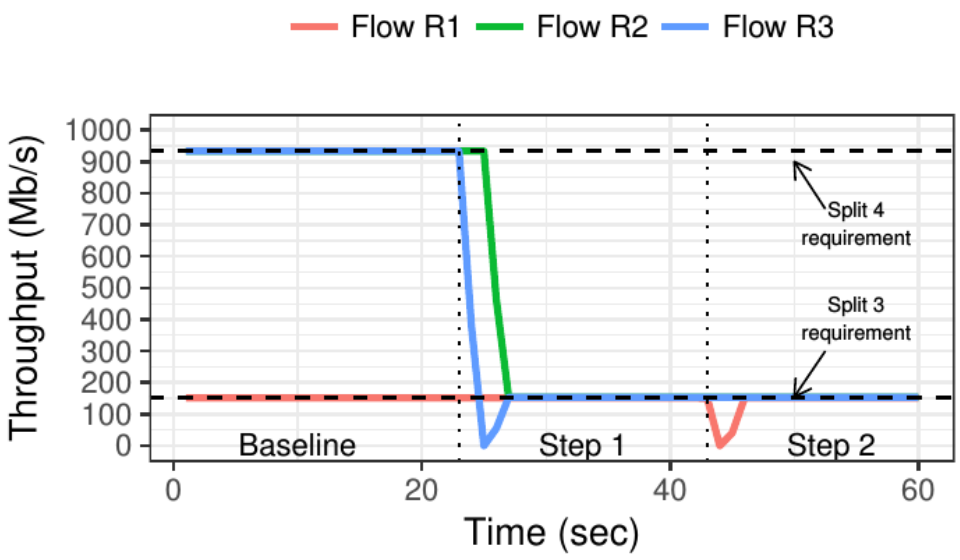
LTE subfunction (f)	Others	RLC	MAC	PHY 2	PHY 1
Relative cost (ς_f)	0.2	0.01	0.14	0.17	0.48
Split 1 (B in Table 1)	CU	RU			
Split 2 (C in Table 1)	CU		RU		
Split 3 (E in Table 1)	CU			RU	
Split 4 (G in Table 1)	CU				RU
Split 5 (J in Table 1)	CU				

Table 8: Testbed components.

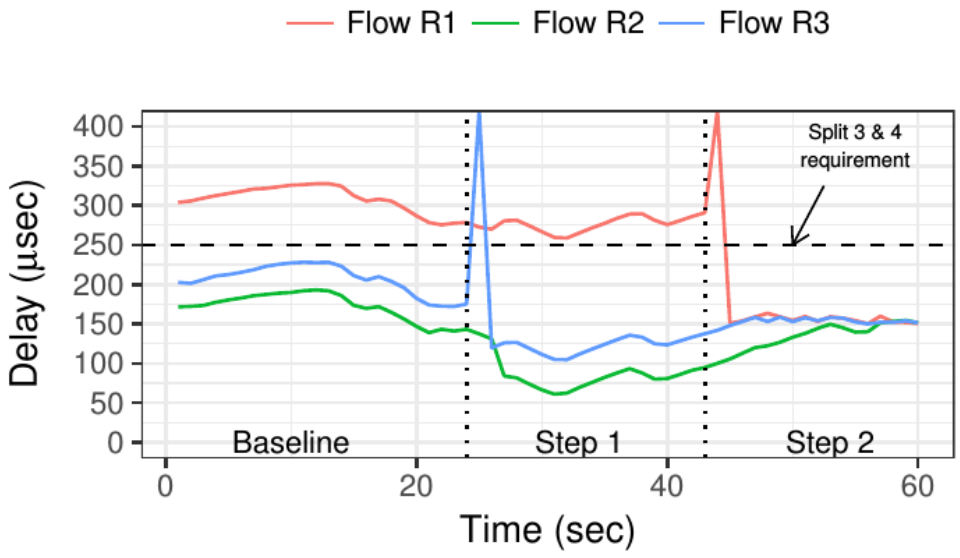
vEPC	OpenEPC Rel. 6
μWave	56 MHz bandwidth @ 7GHz band Adaptive rate $\leq$ 1 Gb/s
mmWave	500 MHz bandwidth @ E-band Adaptive rate $\leq$ 3.2 Gb/s
Switch	OpenFlow switch 48 one-gigabit, 4 ten-gigabit ports
Small-cell	20 MHz channel @ band 3
RU	20 MHz BW @ band 3 Split 1 (PHY, MAC, RLC) and 3 (PHY)
CU	Virtual MAC, RLC, PDCP, RRM, RRC

3.5.1.2 *Use Case Validation*

We present a fault-tolerance use case. Our goal is to validate the ability of our algorithm to maintain maximum centralization degree upon topology changes.



(a) Throughput



(b) One-way delay

Figure 13: Experimental validation.

We start with the baseline scenario shown in Figure 12. Due to the limitation of our testbed to 1-Gb/s interfaces, the algorithm settles with split 4 (from Table 7) for R2 and R3, i.e., no full C-RAN which would require transporting >2-Gb/s flows. After 20 sec, we attenuate the signal of the E-band link, as illustrated in Figure 14. This causes two reactions: (i) the fronthaul flow of R3 is re-routed through switches 2 and 6, and (ii) subsequently the functional split of both R2 and R3 must be softened to split 3 in order to keep a high degree of centralization while guaranteeing that network constraints are satisfied. Finally, 20 sec later, we attenuate the power of the μWave link. This causes the backhaul flow of R1 to be re-routed through link 2-6, which is shared with 2

fronthaul flows. As we can see from Figure 13, the requirements of both fronthaul flows are still satisfied in this step and thus no functional split change is required. In summary, the results shown in Figure 13 validate that our algorithm (i) maximizes the degree of centralization, and (ii) ensures the flow requirements of each split are met at all times.

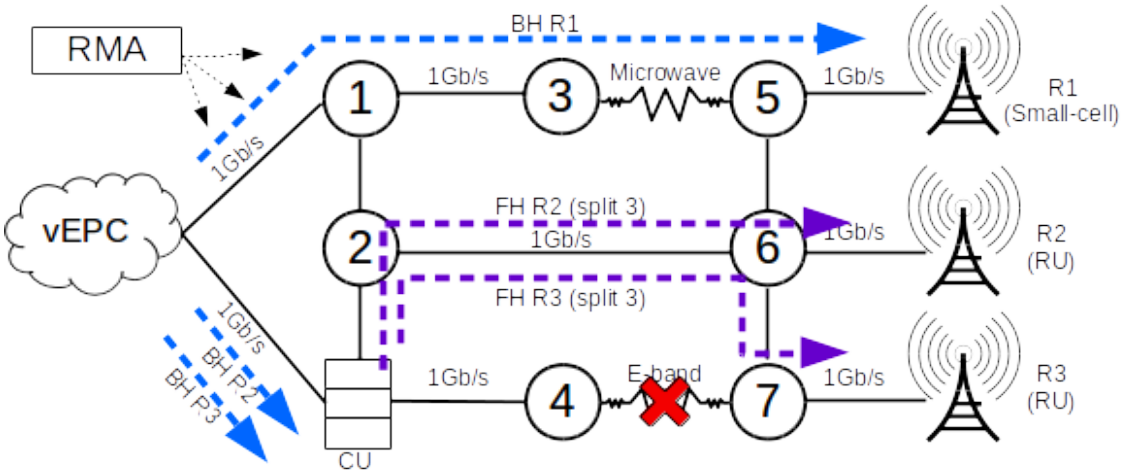


Figure 14: Fault tolerance use case. Step 1.

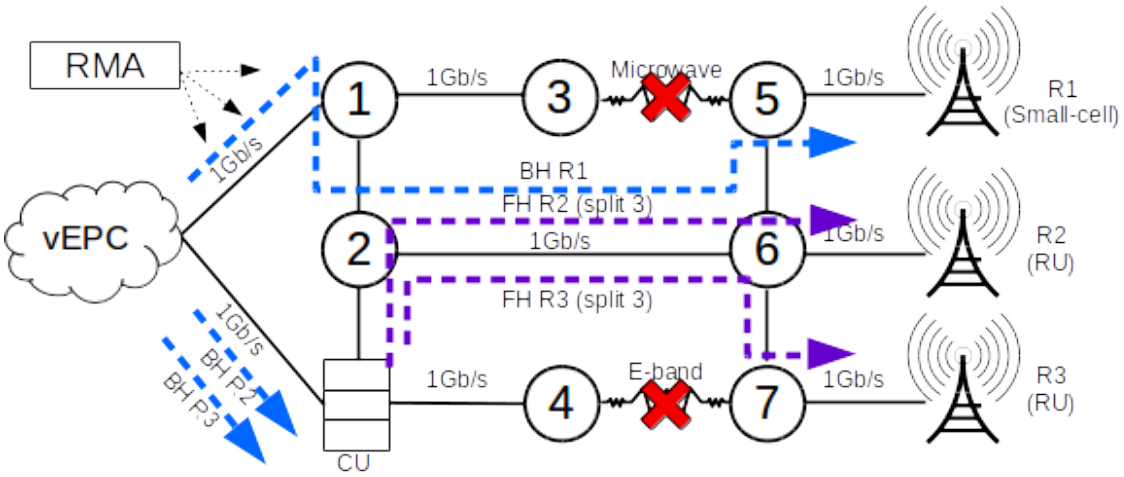


Figure 15: Fault tolerance use case. Step 2.

In Figure 16 we provide insights regarding the nature of the time taken to re-deploy a new configuration when topology changes occur in the use case presented before. The figure shows the amount of time taken on each of the basic operations of our software, namely (i) receiving a notification of the affected hardware elements (labelled "HW reaction"), (ii) running our algorithm (labelled "Algorithm), (iii) install new OpenFlow rules ("Path installation"), and other functions like building/processing messages through the REST interface, etc. (labelled as "Others"). From the figure we see that the overall reaction time is dominated by the delay of the HW components in noticing physical changes. Remarkably, re-computing a new solution barely has a toll given by the simplicity of the topology. We also note that the reaction takes longer in Step 1 (Figure 14) due to path installation process because the flow of R3 has to be managed

by a different physical network card at the CU (as opposed to virtual interfaces in our virtual topology) which results in longer processing.

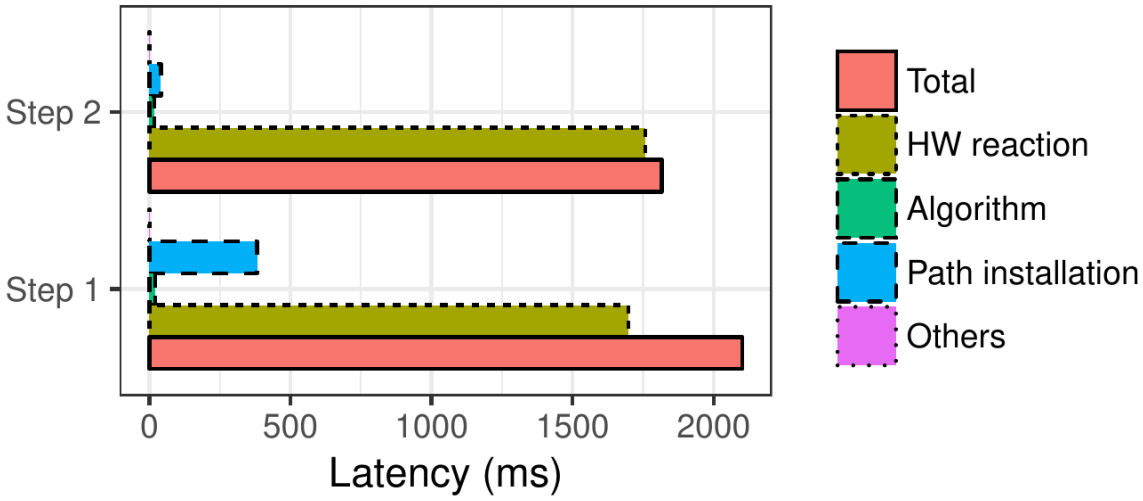


Figure 16: Software reaction time.

3.5.2 Performance evaluation in large-scale topologies

We now assess via simulations the performance of our algorithms with both large-scale synthetic networks and two topologies from two major operators in Europe to assess the performance of RMA as a planning tool.

In order to get statistically meaningful insights, we run each of the algorithms described above over a large set of simulated topologies and extract three parameters for each simulation: degree of centralization γ, whether it is a feasible solution (D-RAN otherwise) and its elapsed time. Each simulation runs on a single Intel i7 core operating at 2.4 GHz. Based on our own estimations, we set $\alpha = 0.5$ to compute the degree of centralization γ i.e. processing functions in an RU is twice as expensive as doing it in a CU. The first and second set of topologies are based on two backhaul networks of existing operators in Romania and Switzerland (up to 900 topologies). The third and fourth sets correspond to random topologies based on tree structures (up to 1800 topologies) and Waxman random graphs (up to 1500 topologies).

3.5.2.1 *Real Topologies*

We create semi-random scenarios based on the backhaul topology of two operators in Romania and Switzerland, illustrated in Figure 17. We know the distance between switching nodes, links connecting them and their capacities. Based on this, we choose the profiles from Table 9 that match better the known capacities. Note that we only consider Ethernet over different PHYs.

We assume classic store-and-forward switching which induces a per-hop latency equal to propagation delay---distance (m)/200μs for copper, distance (m)/300 μs the rest---plus serialization delay (bits/rate).

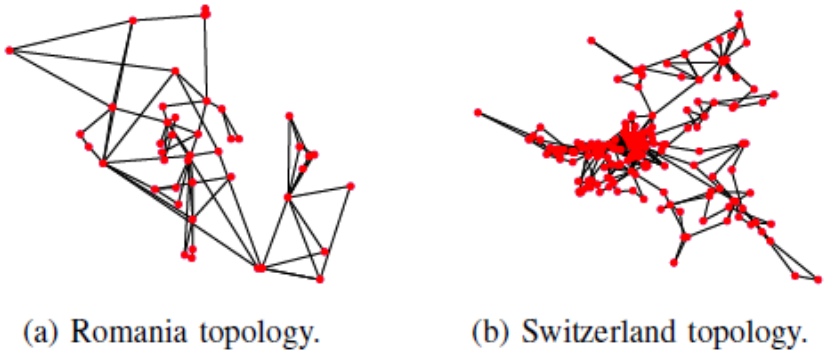
Table 9: Ethernet-based link profiles (proc. delay = 5μs, packet size = 1518 Bytes)

Technology	Bandwidth (Gbps)	Prop. Delay (μs)	Distance (km)
mmWave (60-80 GHz)	0.9, 1.25, 1.5, 2, 3, 4, 8	1-20	0.3-6
μWave (6-60 GHz)	0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 1.25, 1.5, 2	1-100	0.3-30
Copper (1000/10G/40GBASE-T)	1, 10, 40	0.05-0.5, 0.275, 0.15	0.001-0.1, 0.055, 0.03
SMF fiber @ 1310 nm (1000, 10G, 40, 100GBASE-EX, LR, LR-4	1, 10, 40, 1000	1-200, 50, 50, 50	0.2-40, 10, 10, 10
SMF fiber @ 1550 nm (1000, 10G, 40, 100GBASE-ZX, ER, ER-4)	1, 10, 40, 1000	1-350, 200, 200, 200	0.2-70, 40, 40, 40
TbE (*under development)	200, 400	1-50	0.2-10

Table 10: Parameterization of topology generators

Type of topology	Parameters
Romania BH, Switzerland BH	• $\lvert\mathcal{N}_{\text{Romania}}\rvert = 46, \lvert\mathcal{N}_{\text{Swiss}}\rvert = 272$ nodes; • $\lvert\mathcal{R}\rvert=\{\lvert\mathcal{N}\rvert \cdot 0.025i \mid i \in \mathbb{N}, 0 < i \leq 25\}$ RUs; • $\lvert\mathcal{B}\rvert = \lvert\mathcal{Q}\rvert = \{1, 2, 3\}$ CUs/clusters.
Tree-based	• $\lambda_{\text{levels}} = \lambda_{\text{siblings}} = \lambda_{\text{backup}} = 1$; • $\lvert\mathcal{B}\rvert = \lvert\mathcal{Q}\rvert = \{2, 3, 4\}$ CUs/clusters.
Waxman-based	• $\lambda = \{20i \mid i \in \mathbb{N}, 0 < i \leq 10\}, \alpha = 0.4, \beta = 0.1$; • $\lvert\mathcal{R}\rvert = \{1, 2, \cdots 100\}$ RUs; • $\lvert\mathcal{B}\rvert = \lvert\mathcal{Q}\rvert = \{1, 2, 3\}$ CUs/clusters.

Finally, we randomly pick nodes to be RUs and XPUs according to Table 10 and use simple k-means to create as many RU clusters as XPUs. We note that, though we use simple k-means for simplicity (and no generality loss), clustering is important in Cloud RAN and has been object of much study over the years



(a) Romania topology. (b) Switzerland topology.

Figure 17: Snapshot of real topologies

3.5.2.2 *Tree-based Topologies*

To generate topologies with a tree structure, first we generate a set of XPUs connected in a ring. These will be located at the root of the trees. Second, hanging from each XPU, we create a pure random tree using independent Poisson processes with parameters λ_{levels} and $\lambda_{siblings}$ to model the number of levels of the tree and nodes per level. The leafs of the tree correspond to RUs. For each level of each tree, we randomly add a backup link with the upper layer of a neighboring tree, to add additional degrees of freedom for routing, following another Poisson process with parameter λ_{backup}.

Table 10 shows the parameters we used in our evaluation. Once we have the graph structure, we randomize the profile of all the links and their length, using the same assumptions as in the previous scenarios (links in Table 9). For each link and topology, we randomly pick one profile depending on its proximity to a XPU or an RU. To this aim, we group links based on its proximity to a XPU and assign them the same link profile. Additionally, we assign high-capacity profiles to links closer to XPUs with more probability, to capture the fact that aggregation segments typically provide higher capacities. Finally, RU clustering is done as in the previous scenarios.

3.5.2.3 *Waxman-based Topologies*

The last set of topologies consists of arbitrary Waxman random graphs, based on the Erdös-Renyi random graph model, which is popular to evaluate realistic backhaul topologies. To this aim, we used the parameters displayed in Table 10, λ being the intensity of the Poisson process, α the maximal link probability and β a parameter to control the length of the edges. Link profiles, capacities, latencies and clusters are assigned in the same way we did before.

3.5.2.4 *Benchmarking Techniques*

Unfortunately, related literature on QoS routing cannot be used to solve our problem alone. Instead, we compare our algorithms against a few "**best routing, then best split**" strategies (see section 3.1.1) because we can use state-of-the-art tools. We first need to assign RU clusters to XPUs. All our benchmarking schemes use the same heuristic that

we adopt for our algorithm. Then, we use three different baseline routing techniques to compute paths and choose BS splits.

The first algorithm (**Shortest-Path**) greedily chooses, for each RU, the shortest path to its XPU. Then, based on the paths set, we exhaustively search for the best RAN split.

The second algorithm (**Max-Flow**) solves an LP relaxation of the routing problem that maximizes the aggregate flow load; then we use randomized rounding to find single paths and select feasible splits.

The third approach (**Max-Min**) finds a max-min allocation first; then we pick the best functional splits given the allocation of such routing instance.

3.5.2.5 *Test cases*

Table 11 below summarizes the performed evaluation through the execution of the testing procedures described in IR4.3. The results are presented in section 3.5.2.6.

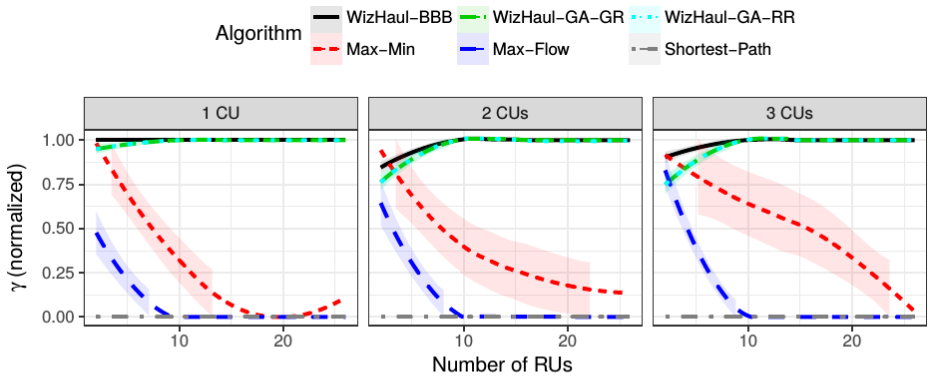
Table 11: RMA algorithm for joint Routing and C-RAN Functional Splits

Test Algorithm	NEC_PCBSSPLIT_01	Execution Status	Passed
Test Name	Branch-and-bound backtracking algorithm (BBB)		
Objectives	Maximizing the degree of BS centralization using a nearly-optimal combinatorial algorithm based on branch-and-bound.		
Test Algorithm	NEC_PCBSSPLIT_02	Execution Status	Passed
Test Name	Greedy based algorithm with Greedy routing (GA-GR)		
Objectives	Maximising the degree of BS centralization using a greedy combinatorial algorithm based greedy routing.		
Test Algorithm	NEC_PCBSSPLIT_03	Execution Status	Passed
Test Name	Greedy based algorithm with Randomized Rounding routing (GA-RR)		
Objectives	Maximising the degree of BS centralization using a greedy combinatorial algorithm based randomized rounding routing.		

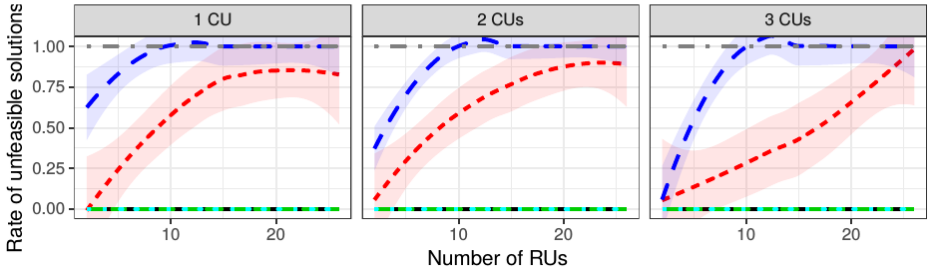
3.5.2.6 *Evaluation Results*

Figure 18 through to Figure 21 depict the normalized degree of centralization, a real value between 0 (D-RAN) and 1 (C-RAN), and the ratio of unfeasible solutions for all algorithms and topologies presented earlier.

We compare our BBB algorithm and two versions of our GA approach, one applying a simple greedy routing approach (GA-GR) and a randomized rounding approach (GA-RR), against the benchmarks presented in section 3.5.1. Importantly, we use GA-RR as initial upper bound for BBB. Their elapsed time is shown in Figure 22. If one simulation reaches 10^5, we stop it and declare the result as unfeasible (which only happens occasionally for Max-Min). Given the large amount of scenarios and heterogeneity of the algorithms, we show curves from a non-parametric local regression fit model in all figures. Shaded areas show the standard error of the fitted model.

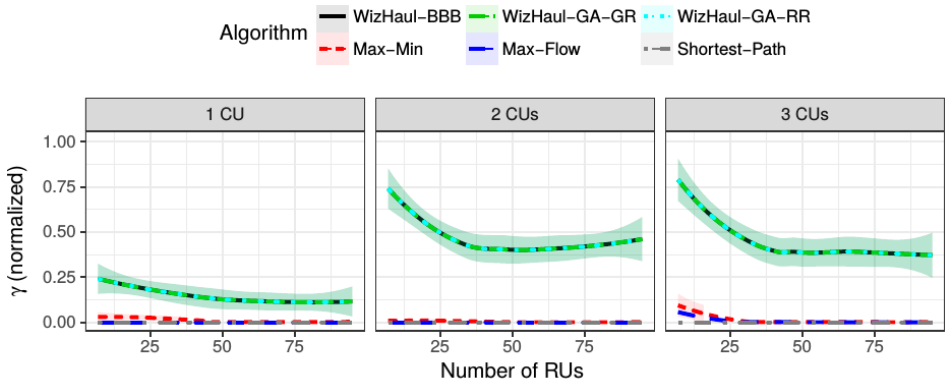


(a) Degree of centralization.

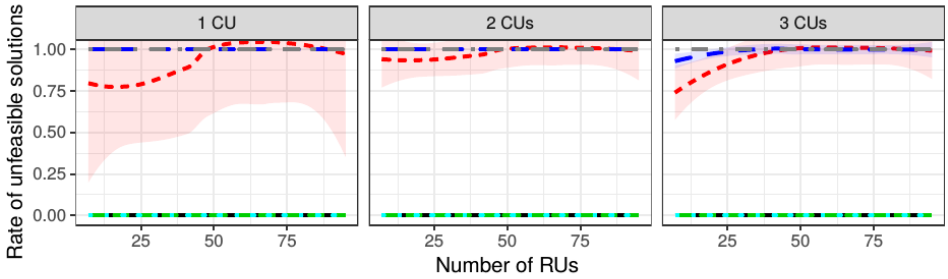


(b) Rate of unfeasible solutions.

Figure 18: Romanian topology

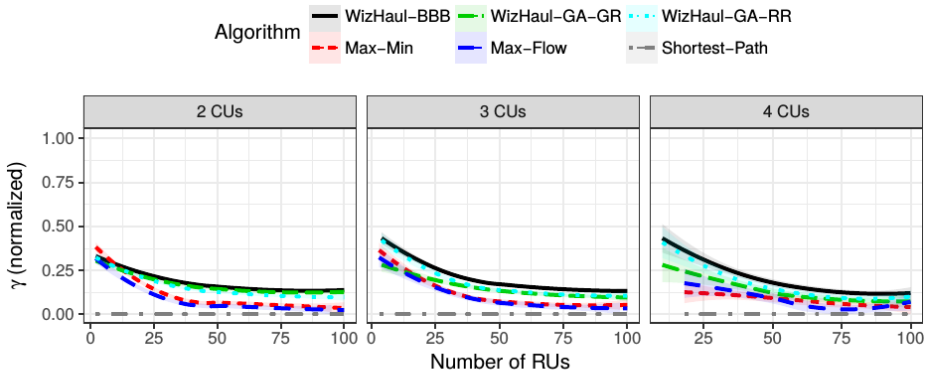


(a) Degree of centralization.

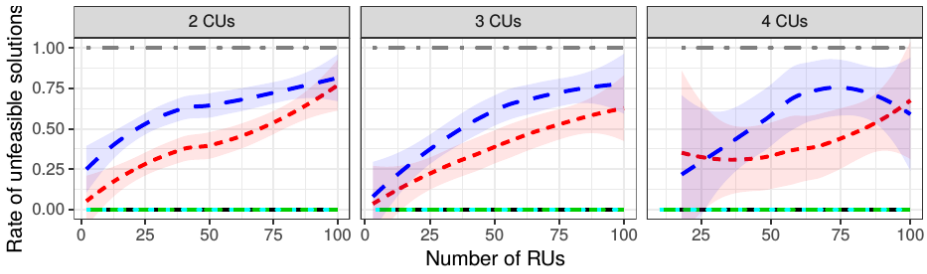


(b) Rate of unfeasible solutions.

Figure 19: Swiss topology

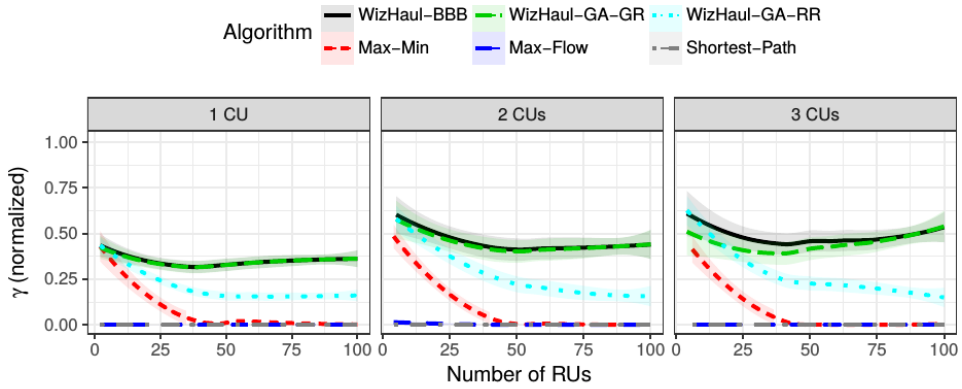


(a) Degree of centralization.

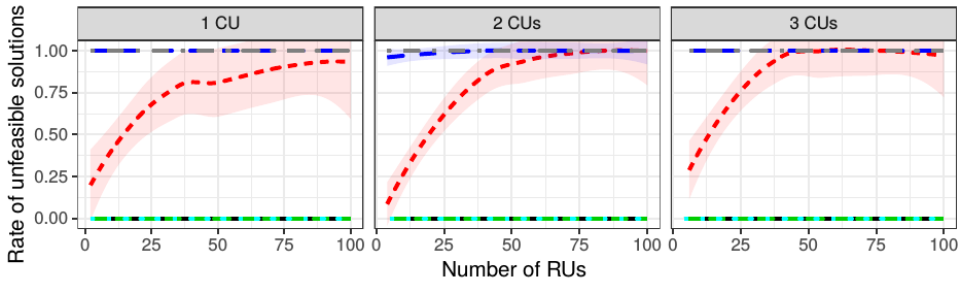


(b) Rate of unfeasible solutions.

Figure 20: Tree topologies



(a) Degree of centralization.



(b) Rate of unfeasible solutions.

Figure 21: Waxman topologies

The first observation is that both greedy approaches (GA-RR and GA-GR) behave very close to the benchmark BBB in most cases. On the contrary, Shortest-Path, Max-Flow and Max-Min render low degree of centralization and a high rate of unfeasible solutions (i.e. no centralization whatsoever) across all types of topologies. The reason is that none of the latter three algorithms have the ability to trade off the centralization degree of some flows to benefit larger clusters. This is because they all have simpler goals: Shortest-Path aims to find low-latency paths without favouring disjoint paths; Max-Flow targets high-capacity paths irrespective of their distance (latency) towards a XPU; Max-Min focuses on both capacity and fairness across RUs, without balancing centralization towards larger clusters.

A second observation is that full centralization can only be achieved in low-scale deployments like our Romanian-based scenarios. This is because XPUs and RUs are generally too far apart. This implies not only that the tight latency demands of full C-RAN can be rarely met but, also, that more links are compromised, limiting the chances of finding disjoint paths.

Our third observation is rather intuitive, that is, we can achieve higher degree of centralization in scenarios with lower number of RUs (less contention) and larger number of XPUs (shorter paths between RUs and XPUs).

In addition, the tree-based topologies render lower centralization degree than the other structures under evaluation. This occurs because these scenarios have less diversity of routes (e.g. no disjoint paths) with many links aggregating multiple flows.

Finally, whereas GA-GR follows very closely the performance of BBB in Waxman-based scenarios, GA-RR deviates as the number of RUs grows. The reason lays in the fact that the linear program used in GA-RR, weights links with their latency but fails to trade longer routes (which could still meet latency requirements) to release additional capacity for other paths that in this case will violate throughput constraints. Note that these are the most complex topologies in terms of diversity and number of links.

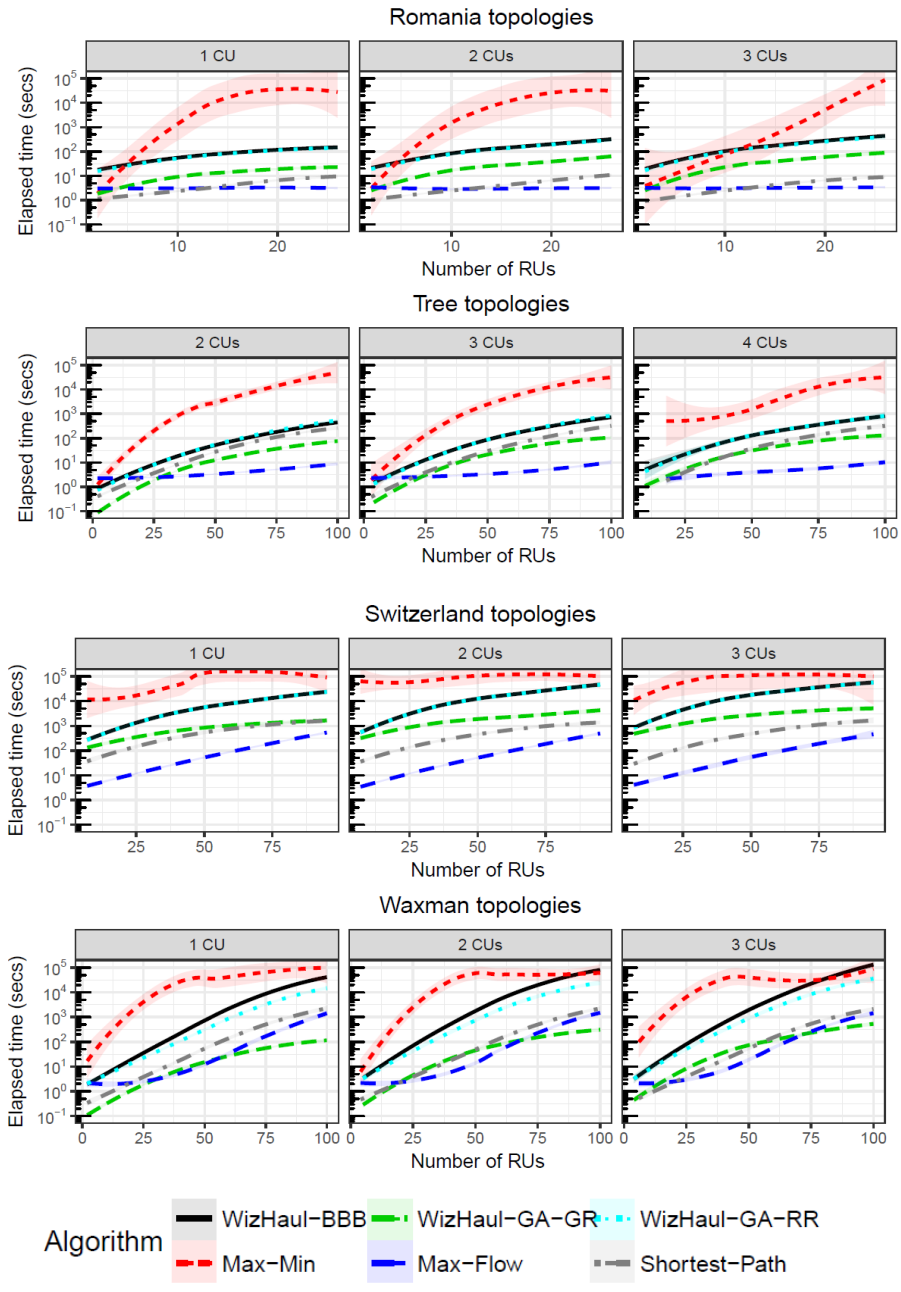


Figure 22: Elapsed time (sec) vs. number of RUs and XPUs

Regarding the running time of these algorithms, shown in Figure 22, we can observe that Max-Min is the most complex algorithm of them all (though it could be accelerated by using dual variables). Perhaps surprisingly, BBB behaves similarly to GA-RR. This is simply because we use GA-RR as an initial upper bound configuration for BBB and, because GA-RR usually finds a good solution, BBB quickly prunes worse candidate solutions out of the whole space. Max-Flow is the fastest because it is based on a simple LP relaxation, faster than Shortest-Path which exhaustively searches for a RAN split after path computation.

3.5.2.7 *Summary*

For the evaluation, we defined a metric to measure the degree of centralization and proposed three algorithms to maximize it while meeting next generation fronthaul network constraints: BBB, GA-GR and GA-RR. BBB provides us with a near-optimal solution which yields a performance upper bound. GA-GR and GA-RR are greedy heuristic solutions where GA-GR achieves the best trade-off between computational load reduction and distance to the optimal solution. The mechanisms designed can be used both at network planning and operational runtime phases to achieve the maximum centralization degree. GA-GR has been implemented in a proof-of-concept with commercial hardware and its properties at operational runtime have been analyzed.

4 Energy management and monitoring Application (EMMA) for XPFE/XPUs

The EMMA application implements two main functionalities: monitoring of power consumption and energy efficient management of resource allocation in infrastructures composed of software-based XPFEs and XPUs.

The monitoring of power consumption is performed for different kinds of physical and virtual entities. In particular, the application manages physical infrastructures composed of software based XPFEs and XPUs, computing both total consumption and contribution of single devices, i.e. XPU servers and XPFE network nodes, based on their computing or traffic load. At the virtual level, EMMA can estimate the power consumption for single network connections or all the network connections associated to a given tenant. This estimation is based on an analytical model that takes as input the flow statistics collected from the XPFEs. Moreover, EMMA is also able to estimate the power consumption associated with virtual Network Services, i.e. collections of Virtual Network Functions which are interconnected through forwarding graphs established through dedicated network connections on the XPFE domain. The EMMA offers a single graphical interface to visualize the current power state and power consumption of

XPFEs and XPUs, as well as historical data on power consumption trends through graphs.

The energy efficient management of resource allocation can operate in three different modes: (i) at the network level only, (ii) at the computing level only, and (iii) jointly for both network and computing resources. In all the cases, the EMMA adjusts dynamically the power state of the target devices based on the current traffic and processing requests, reducing the number of nodes in active states through a suitable resource allocation strategy.

The first option (network only) is adopted in case of computation of network paths among pre-defined end-points. It is based on a routing algorithm which applies power-consumption dependent weights for the different links and nodes in the network graph. This kind of algorithm was described in D4.1, section 3.5.2. When operating at the computing level only, EMMA optimizes the power consumption of the XPUs reducing the number of active hosts, but without considering the underlying connectivity needed to interconnect them. This approach simplifies the computation and provides a good level of energy saving (even if not the optimum solution) since computing nodes are typically responsible for a significant contribution to the overall power consumption of the infrastructure. Finally, the third option optimizes the power consumption in the whole infrastructure since the resource allocation algorithms operate jointly at the XPFEs and XPUs level, considering the combination of computing, network connectivity and bandwidth requirements. These algorithms are detailed in the Appendix, in section 17.1.

4.1 Consolidated design

The EMMA application operates on physical infrastructures including software-based XPFEs and XPUs and it is developed as a Java application implemented from scratch. As shown in Figure 23, it operates on top of the 5G-Crosshaul XCI SDN controller and VIM services, and also implements part of the NFVO functionalities for what concerns the orchestration of resource allocation in network and computing domains. At its north-bound interface, the EMMA application offers REST APIs to enable its integration in broader management systems. Moreover, the operator can interact with the EMMA application using its Graphical User Interface.

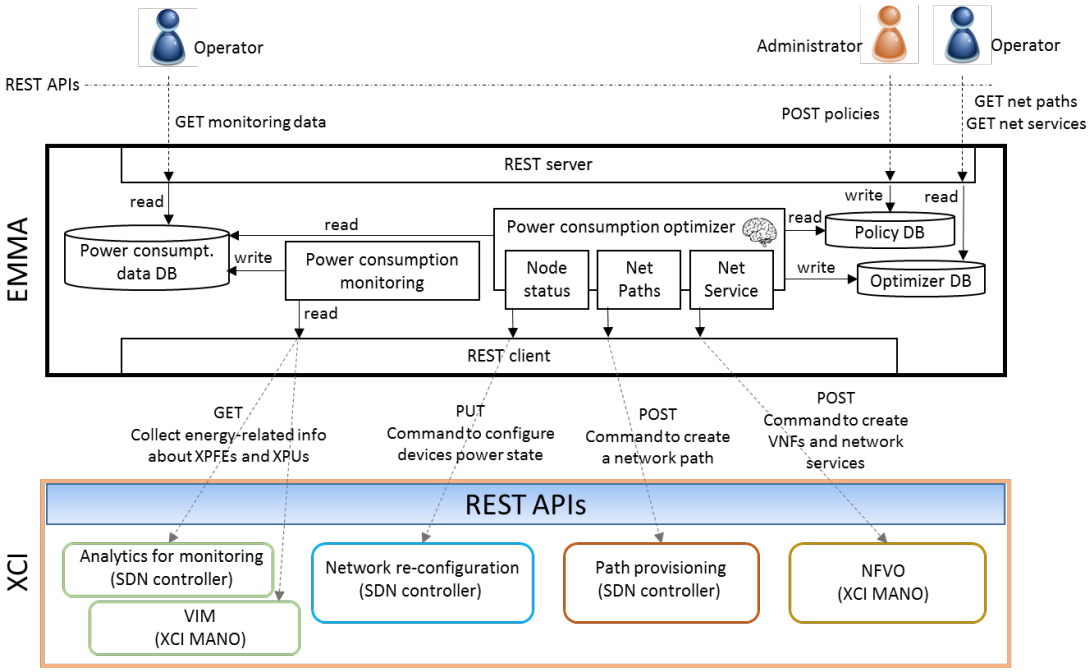


Figure 23: EMMA high-level software design

The EMMA application implements the following main functionalities:

- Correlation of energy-related information for network and IT domains, as exposed by the SDN controller and the VIM components of the XCI, to provide summarized energy consumption data to the administrator of the physical infrastructure. Based on this kind of information, the EMMA also adjusts the power status of the devices (e.g., putting nodes in sleeping mode).
- Optimization of network path provisioning for what concerns power consumption across end-to-end paths and possible re-planning of already established network paths for power consumption re-optimization. As an example, the application targets a specific network technology (i.e., 5G-Crosshaul XPFEs based on Lagopus software switches, controlled via OpenDaylight).
- Optimization of VNFs placement and network paths allocation in network services, still minimizing the whole power consumption while guaranteeing the compliance with the service specification, e.g. regarding disjoint VNFs placement. The implementation targets a specific type of Network Service (i.e., the vEPC) to show a concrete use case.

The application exposes towards the user (e.g., the administrator of the physical infrastructure) a set of REST API for monitoring, operation and management actions.

4.2 Implementation

The main internal components of the EMMA application, also shown in Figure 23 depicting the high-level software design, are summarized in Table 12.

Table 12: Internal components of the EMMA application

Component	Description
REST Client	Entity in charge of interacting with the XCI components via REST APIs.
Power consumption monitoring	Entity in charge of coordinating the collection of power consumption information from XCI components (i.e. VIM for XPUs and SDN controller for XPFEs), processing and aggregating this information and storing and organizing the results in the internal *Power consumption data DB*.
Power consumption data DB	Internal no-SQL DB to store power consumption data, related to single XPUs or XPFEs, end-to-end network paths, VNFs and Network Services, or physical domains at the physical infrastructure level.
Power consumption optimizer	Entity in charge of computing the optimum allocation and status of physical resources, based on the information stored in the *Power consumption data DB* and the new service requests, in terms of single VNFs or network connections and entire Network Services composed of multiple interconnected VNFs. It acts at three different levels: • Regulation of power status of single network nodes and hosts • Computation of energy efficient network paths • Computation of energy efficient VNF placement for Network Services. It also interacts with the XCI components to change the configuration of power status in selected infrastructure nodes (both network nodes and servers) and request the provisioning of the computed network paths and Network Services.
Policy DB	SQL database, storing the policies configured by the administrator.
Optimized DB	SQL database, storing the device status, network paths and Network Services computed by the *Power Consumption Optimizer*.
REST Server	Entity in charge of implementing the north bound REST API of the EMMA application for retrieving power consumption monitoring data and power optimization actions as well as for requesting the instantiation of new VNF-based services and network connections.

The EMMA application interacts with the XCI components, in particular the VIM, the NFVO and the SDN controller, making use of the following services and interfaces (based on REST APIs):

- VIM and NFVO services: instantiation and termination of VMs in given hosts; instantiation and termination of virtual resources for networks, subnets and network ports; collection of power consumption monitoring data for XPUs; collection of information about resource capabilities and availabilities in XPUs; configuration of power states for XPUs.
- SDN controller services: setup and tear-down of network connections with specific paths; collection of network topology information; collection of power consumption monitoring data for the network infrastructure; collection of traffic flow statistics; configuration of power states for network nodes.

4.3 KPIs

The following table lists the KPIs addressed by EMMA for XPFE/XPUs

Table 13: list of KPIs addressed by EMMA for XPFE/XPUs

Application	***EMMA for XPFE/XPUs***		
List of related project KPIs	***Obj.7: Design essential Xhaul-integrated (control/planning) applications*** *(5GPP KPI) Reduce energy consumption in the 5G-Crosshaul by 30% through energy management* ***Obj.2: Specify the XCI's northbound (NBI) and southbound (SBI) interfaces*** *(5GPP KPI) Enable the introduction/provisioning of new 5G-Crosshaul services in the order of magnitude of hours* ***Obj.6: Design scalable algorithms for efficient Xhaul resource orchestration*** *(5GPP KPI) Reducing the network management Operational Expenditure (OPEX) by 10%*		
Measurement of project KPIs as well as application specific ones	**KPI metrics (unit)**	**Description**	**Way of measurement**
	Energy saving (%)	Percentage of energy efficiency/savings (i.e. average power consumption per flow)	In WP5 the power consumption measurements are collected directly from XPFEs and XPUs. In WP4 power consumption data are computed based on an analytical model that adopts parameters obtained from real hardware. The measurements of power consumption are compared with the ones collected with EMMA application disabled,

			i.e. maintaining all the devices always activated, and with the results obtained applying a different energy management strategy derived from literature [17].
	Time for path provisioning	Time required to establish a path, including the time required to compute the resources, activate the required XPFEs (if needed) and configure their flow entries.	This measurement is performed in the context of WP5, through experiments on the 5TONIC testbed where EMMA is integrated with real hardware. The measurement is done capturing and analyzing the pcap files tracing the exchange of control messages at EMMA NBI and between EMMA and XCI SDN controller. The path provisioning time, if compared with the approach where all the XPFEs are always on, increases of a few seconds, due to the time needed to change the XPFEs' power state.
	Time for vEPC service instance provisioning	Time required to provision and configure a vEPC service, including time required for: - Computation of resource allocation in XPFEs and XPUs - Activation of required XPUs and XPFEs - Allocation of VMs to host the VNFs - Configuration of the VNFs - Configuration of flow entries in XPFEs for inter-VNFs connectivity	This measurement is performed in the context of WP5. The measurement is done capturing and analyzing the pcap files tracing the exchange of control messages at EMMA NBI and between EMMA and XCI components (SDN controller and NFVO). Also in this case, the provisioning time increases of few seconds, which is negligible if compared with the

			total provisioning time (in the order of minutes and mainly bounded to the VNFs instantiation time).
List of the State-of-the-Are approach **(used to compare with proposed solutions)**	1. State of the art approach 1: Provisioning of network connections based on shortest path algorithms. Shortest path algorithm for network path computation, with all devices always on. EMMA introduces a mechanism to change the power state of the XPFEs (three different power states supported by Nokia XPFEs) depending on their usage and the characteristics/load of the traffic. 2. State of the art approach 2: Manual management of service provisioning, without energy efficient resource allocation algorithms. Provisioning of vEPC not yet fully integrated with NFV MANO. Some work is currently in progress: https://www.opnfv.org/community/upstream-projects/openairinterface, but not yet finalized. Moreover, in this case, the vEPC instances are created in a traditional network environment (i.e., not over an XPFE network) and without adopting any specific algorithm to optimize the resource allocation (i.e., in an environment with servers controlled by OpenStack, the Nova scheduler is used). 3. State of the art approach 3: Provisioning of VNFs, based on the algorithm proposed in [17]. The main difference with the EMMA procedures is the static deployment of the VNFs at each server, while EMMA adopts a dynamic approach.		

4.4 Validation and Evaluation

This section describes the performance of the EMMA application, applied to XPFE/XPUs domains, when running in an emulated environment. The emulation of the Crosshaul physical infrastructure allows validating the EMMA approach and its algorithms in scalable scenarios. This is particularly suitable to measure the KPIs related to the energy saving with variable traffic loads and topologies, allowing comparing the results with state of the art solutions targeting similar energy efficiency issues. The EMMA functional validation in a real test-bed is performed in WP5 and will be reported in D5.2.

4.4.1 Evaluation Environment and Scenario

In WP4 the reference scenario is based on an emulated environment, using the Mininet tool to emulate a network with a representative Crosshaul topology including both network nodes (XPFEs) and hosts (XPUs). The topology is shown in Figure 25(a).

The software includes the EMMA application and the XCI, with the SDN controller implementing the features for energy monitoring and management over XPFEs domains (see Figure 24).

We demonstrate our optimization strategy in a simple test scenario, depicted in Figure 25(a), with ten edge switches and five core switches. We take into account a period of one hour, and make the traffic generated at each edge switch during that time, change between 0.5 and 3 GByte. We further assume that all traffic traverses the logical graph displayed earlier, that all types of processing require r(v) = 1 computational unit per GB of traffic, and that each core switch has a computational capability of k(c) = 20 units.

Based on the work in [18], we define the network-related power consumption functions as E_{idle} = 90 W • 1hour = 324kJ per active core switch, and E_{sw} = 142.4nJ/Byte. Based on [19], we also set E_0 = 0, i.e., deploying containerized VNFs and not using them has no measurable impact on the energy consumption, and E_{proc} = 11.7 μJ/Byte.

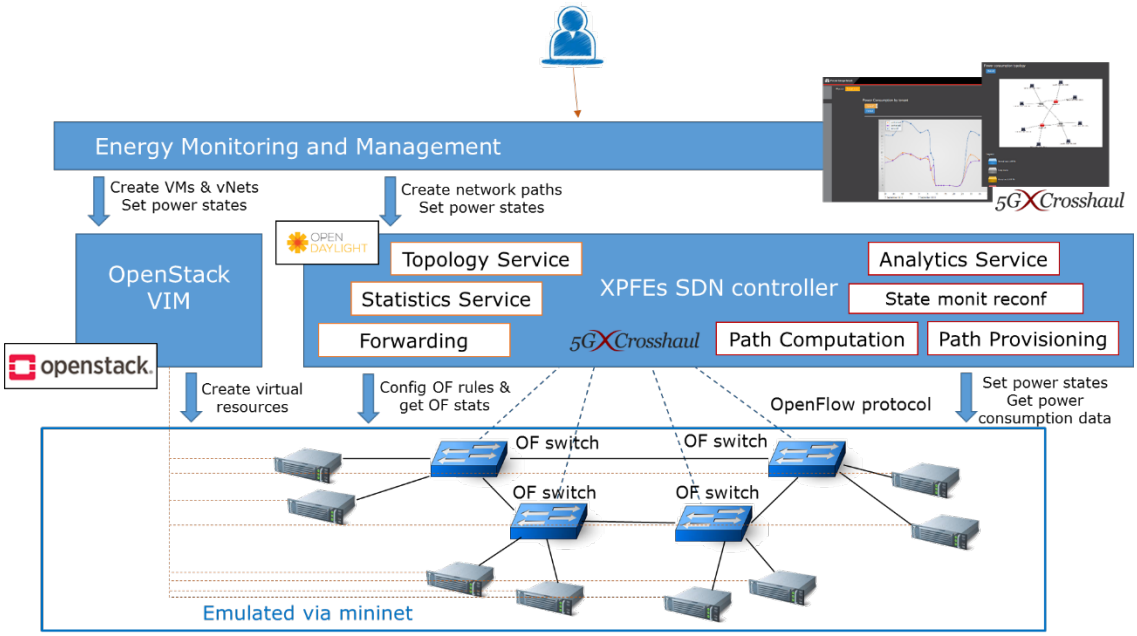


Figure 24: Components of the EMMA system: EMMA application, XPFEs SDN controller and emulated network.

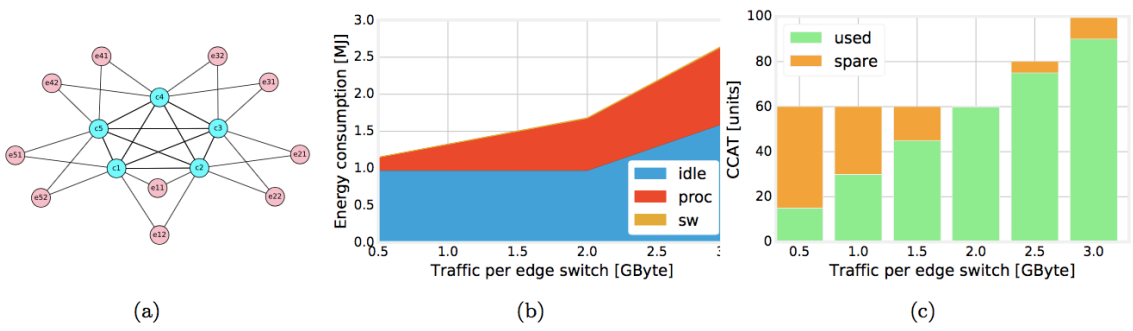


Figure 25: Test scenario (a); energy consumption (b) and utilization of the computational capabilities of the active topology (CCAT) (c) as a function of the traffic.

4.4.2 Test Cases for functional validation

This section describes the EMMA functional validation, performed through the execution of testing procedures. The main focus here is on the internal behaviour of the EMMA software components, their interaction within the EMMA system and the interaction with the external components (REST clients at the EMMA northbound interface and mock REST servers at the interface with the XCI).

Table 14: Test cases for functional validation

Test Card 1	NXW_EMMA_APP_01	Execution Status	Passed
Test Name	Energy Monitoring		
Objectives	Verify the mechanisms to collect XPUs and XPFEs power consumption monitoring data at the XCI's NBI.		
Test Card 2	NXW_EMMA_APP_02	Execution Status	Planned
Test Name	Energy-related policies		
Objectives	Verify the mechanisms to configure, store and retrieve energy-related policies in EMMA.		
Test Card 3	NXW_EMMA_APP_03	Execution Status	Passed
Test Name	Automated configuration of devices power states.		
Objectives	Verify the EMMA mechanisms to detect inactive devices and put them in sleeping mode (through XCI NBI command). Verify the EMMA mechanisms to switch on devices (through XCI NBI command) when required by new services.		
Test Card 4	NXW_EMMA_APP_04	Execution Status	Passed
Test Name	On-demand computation and provisioning of energy-efficient network paths.		
Objectives	Verify the EMMA mechanisms to instantiate on-demand an energy-efficient network path.		
Test Card 5	NXW_EMMA_APP_05	Execution Status	Passed
Test Name	On-demand computation and provisioning of energy-efficient Network Services.		
Objectives	Verify the EMMA mechanisms to instantiate on-demand an energy-efficient Network Service.		

4.4.3 Evaluation Results

A first issue we are interested in is the global energy consumption, and how it is broken down into its components. Figure 25 (b) shows that E_{idle}, i.e., the energy consumed to keep the switches on regardless of how much they are used, is a substantial contribution to the total energy consumption, followed by the processing power E_{proc}, with the latter almost matching the former in high-traffic scenarios. Considering that an idle switch consumes 90 W of power while modern server-grade processors have thermal design powers of roughly 65 W [20], this result makes intuitive sense.

Figure 25 (c) shows how the computational capabilities of the active topology (CCAT) changes as a function of the traffic, and how much of it is actually used. It is interesting to see that when the traffic demand is low, the core switches are idle for a significant fraction of their time; indeed, we would activate fewer of them, were they not needed to ensure connectivity. As the traffic increases, instead, core switches become almost fully loaded, and we need to activate more of them in order to meet the demand for computational capabilities. These results suggest that adding more connections between edge and core switches could be an effective, if somehow unorthodox, recipe for energy savings, especially in networks that are usually faced with low traffic demand.

We compare our solution with two alternatives. The state-of-the-art of what is done in practice, and the state-of-the-art of what is envisioned in the literature. De facto, what is done in real-world cases is always keeping all core switches and all links active; in our model, this corresponds to fixing all x− and y-variables to one and optimizing the resulting problem.

Among the plentiful and creative VNF placement schemes we reviewed earlier, the closest to our scenario and goal is [21]. In Sec. IV.A therein, the authors propose an enhanced consolidation heuristic to decide which servers should be activated in a network. The consolidation procedure starts with all servers disabled; upon arrival of a flow, it tries to serve it with the existing servers, choosing the closest available to the entry point of the flow itself; otherwise, it activates a new server, against trying to minimize the distance between such a new server and the entry point of the flow.

There are two main differences between our system model and that of [21] that could hinder a straightforward comparison, namely:

- in [21], the VNFs deployed at each server are static and given, i.e., there is no equivalent of our δ-variables;
- servers (that run VNFs) are a different entity from switches (that guarantee connectivity).

In order to circumvent the first issue, we adopt a three-stage decision process instead of its two-stage counterpart in [21]: we first look for an already-deployed VNF to serve a flow; if none can be found, we deploy a new instance of the needed VNF at an already-

enabled core switch; otherwise, we activate a new core switch. As far as the second issue is concerned, we enable any core switch necessary to ensure connectivity between edge switches and the core switches serving them. Notice however that we virtually never have to do so, as close-by switches are always preferred if available.

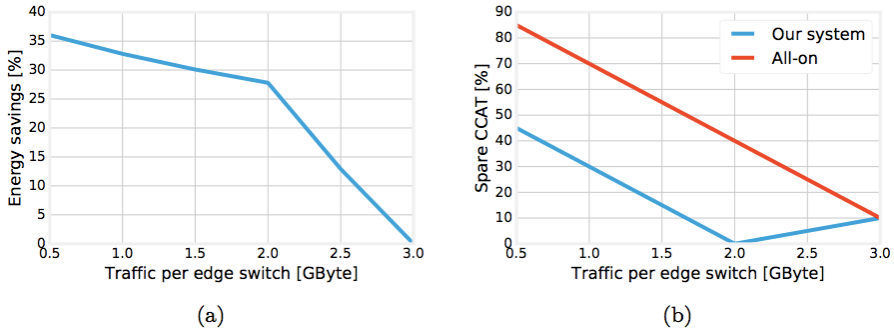


Figure 26: Comparison with the currently-deployed state of the art: energy savings (a) and spare computational capacity in the active topology (CCAT) (b)

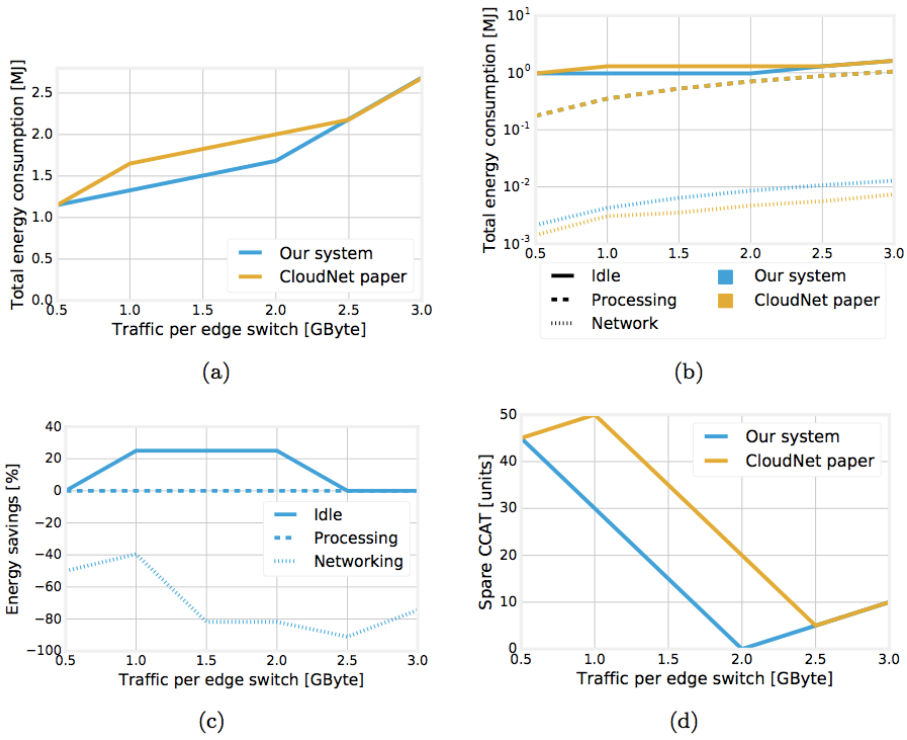


Figure 27: Comparison with state-of-the-art proposals: total energy consumption (a), energy consumption breakdown (b), savings (c), spare CCAT (d)

Figure 26 shows how our algorithms perform compared to what is currently done, i.e., leaving all core nodes on. We can save up to 35% of energy (Figure 26 (a)) when the traffic is low, and therefore there are more opportunities to disable core nodes and/or links. Savings steadily decrease as the traffic increases, and disappear for the highest traffic level, when we have to enable all core nodes. Consistently, Figure 26 (b) shows that the spare computational capacity of the active topology (CCAT) is always higher in the state-of-the-art scenario than in ours.

We then move to the recently proposed energy-saving scheme [21]. Even compared to that proposal, we can consume substantially less energy, as we can see from Figure 27 (a). The breakdown presented in Figure 27 (b) highlights how most of the savings come from the fact that we are able to activate fewer core nodes; on the other hand, we do tend to make the traffic travel a little longer (dotted lines in Figure 27 (b)). The overall savings exceeds 20%; as we can see from Figure 27 (c), such a figure originates from 25% savings in idle power, no difference in processing power, and a higher network power. Finally, Figure 27 (d) shows that our system also corresponds to a lower amount of spare CCAT, a direct consequence of the fact we enable fewer core switches.

It is also interesting to observe how, in Figure 27, we observe little or no difference in the performance when the traffic is very high or very low. This is because in the former case there are no possible savings, and in the latter it is very straightforward to identify the most efficient network configuration. In intermediate cases, when there are multiple possible solutions and it is hard to identify the best one, the superiority of our proposal is more evident.

4.4.4 Scalability

The scalability of the EMMA was tested through emulation in [18]. The results are presented below, while further details can be found in [18] for reasons of space. The behavior of EMMA compared to the No Power Saving scheme, as the network size varies, is presented in Figure 28. The plot confirms the steady performance of EMMA: it reduces the power consumption per flow by a factor ranging from 2 (for 10 core switches) to 8 (for 40 core switches), with the average power consumption/flow linearly increasing.

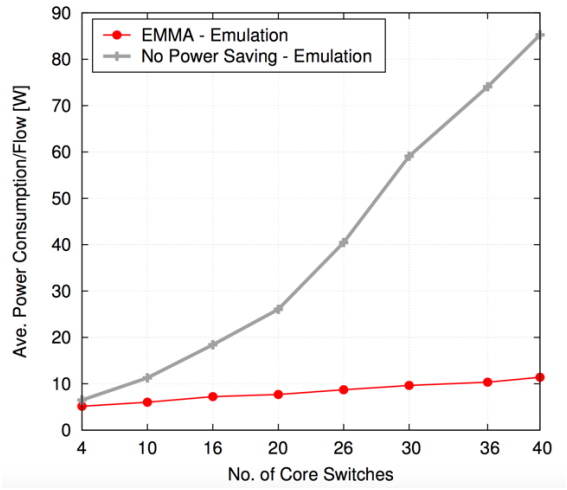


Figure 28: Average power consumption per flow vs. no. of core switches comparison between EMMA and No Power Saving

A similar conclusion is derived from Figure 29, which depicts the gain that we can achieve with EMMA with respect to the No Power Saving strategy, as a function of the

number of core switches and for a flow arrival rate equal to 0.05, 0.1, 0.5, 1. The gain is computed as the difference in power consumption between No Power Saving and EMMA, normalized to the power consumption of the former scheme.

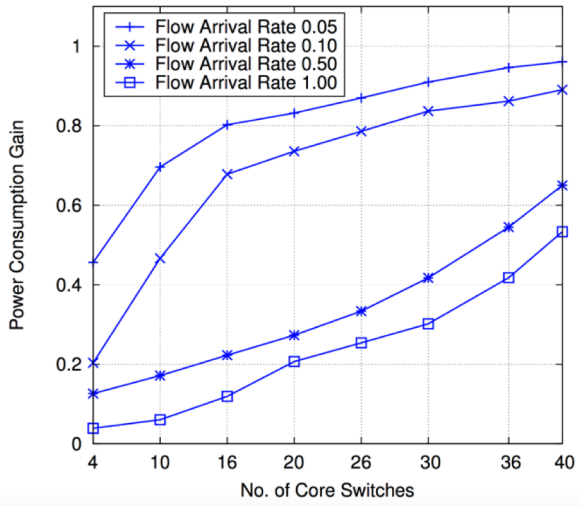


Figure 29: Gain in average power consumption per flow (derived by emulation) provided by EMMA with respect to No Power Saving, as the number of core switches and the flow arrival rate vary.

5 Energy management and monitoring Application (EMMA) for mmWave Mesh Networks

The EMMA for mmWave mesh networks is an over the top application in the 5G-Crosshaul domain. mmWave mesh network considers a dense node deployment of backhaul nodes which provide access locally via a small cell. Since all the backhaul connection are wireless on mmWave a gateway towards the core-network is needed. So all traffic is routed towards this gateway within the created scenario. The goal of the EMMA is optimizing the energy consumption of the given network topology and user distribution while maintaining the user's traffic demand. The simulated behavior of EMMA is focusing on satisfaction of the users, which results in always keeping up with

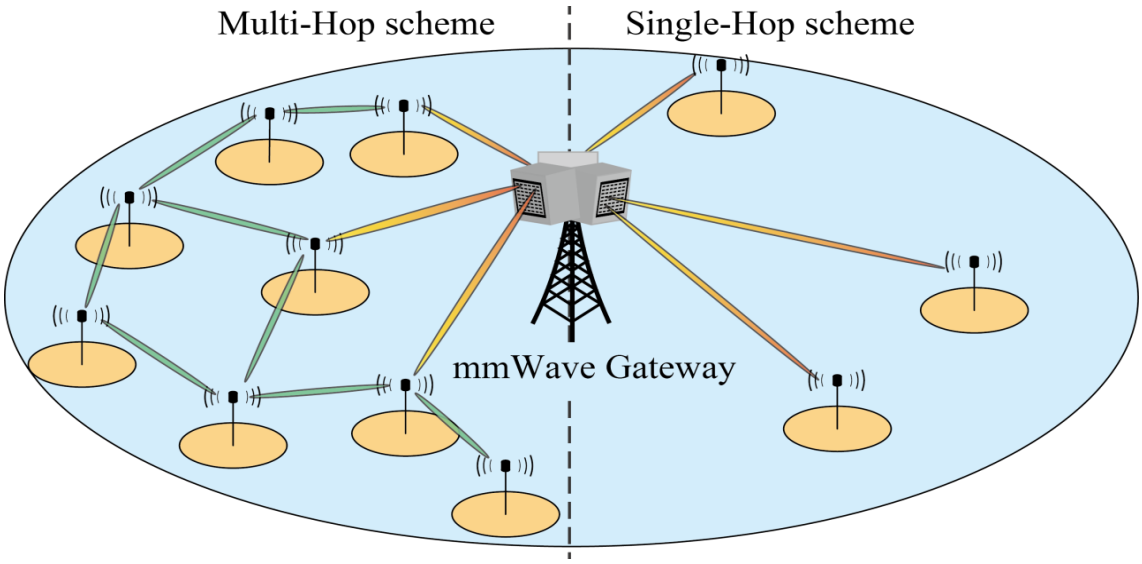


Figure 30: Different routing schemes in dense urban scenario

traffic demand. The simulation is stand-alone and implemented in MATLAB only. Integration with given 5G-Crosshaul elements is studied in WP5, where the algorithm is tested within a real mmWave mesh network, and the results are compared to the simulation results presented here.

5.1 Consolidated design

The Design of the EMMA for mmWave mesh aims at the dense urban scenario (Figure 31) where the user distribution is time-variant and spatially non-uniform. The considered Network topology is a mmWave mesh, overlayed by a LTE-macro BS. This forms a heterogeneous network (HetNet) while the user has access via LTE and mmWave at the same time (Multi-RAT). The focus is the mmWave mesh acting as wireless backhaul serving infrastructure with a mmWave gateway in the network for connecting to remote services. Due to the user scenario, the mesh network needs to be adaptive in routing. Two possible schemes are possible: Multi-Hop and Single-Hop. The Single-Hop design is not preferred due to the high free space attenuation inherent for the mmWave channel. The multi-hop schemes the connection to the gateway is not direct but via multihop. Here relaying the traffic over different hops can compensate the path loss as well as enabling route multiplexing. Forming the routes via the multi-hop

scheme considers only maximum bandwidth mmWave links based on the IEEE 802.11ad standard.

As mentioned the algorithm takes into account the distribution of the users. One can see the user within the network as a moving crowd in some scenarios. Nonetheless, the mmWave mesh has to be adaptive to the time-variant user distribution within the network. So forming mmWave backhaul links and multiplex mmWave based routes is essential to cope with the traffic demand in congestive areas. The goal for EMMA here is maintaining all the service while reducing the energy consumption of the whole network. Disabling and enabling various mmWave small cell base stations as the users are moving is key for satisfying the user's traffic demand.

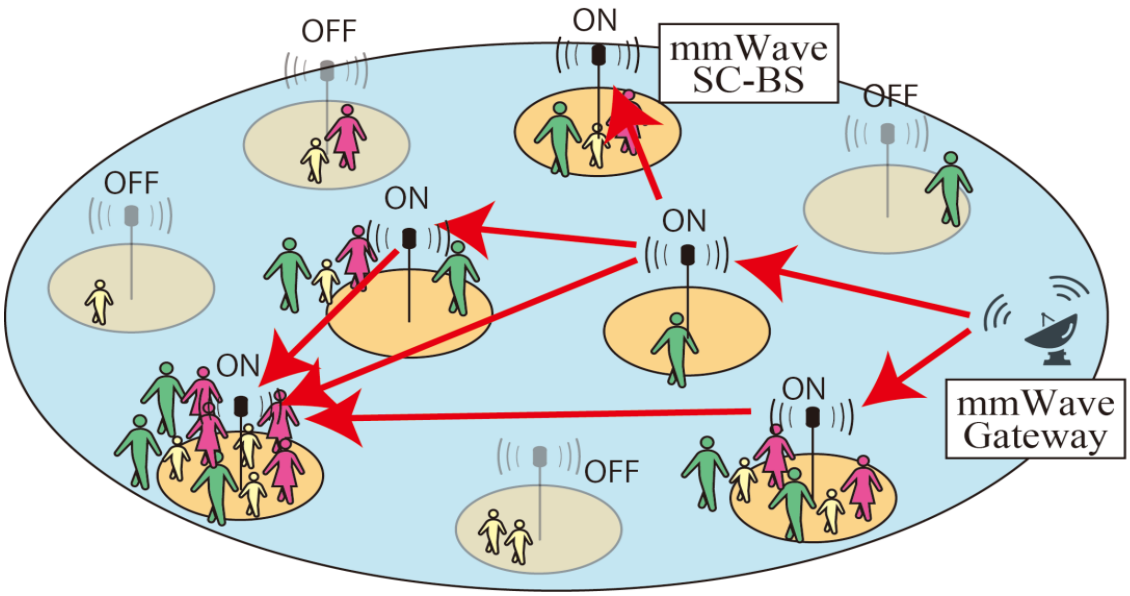


Figure 31: State of mmWave mesh network with multihop routing and disabled mmWave SC-BS

The architecture of the EMMA algorithm within the 5G-Crosshaul framework is straightforward. The application is on top of the XCI. All the necessary communication, i.e., reading topology, getting link status, setting paths can be done via the standardized REST API on the northbound interface of the XCI.

5.2 Implementation

EMMA solves a combinatorial optimization problem that determines mmWave mesh backhaul paths and ON/OFF status of each node. However, this problem is NP-hard, especially when many nodes are deployed. Thus, the EMMA algorithm is divided into 3 steps so that we can find a heuristic solution.

Developed EMMA algorithm in 3 steps

1. Initial mmWave node activation.
   If we assume that network power consumption is proportional to the number of activated mmWave nodes, the goal of minimizing power consumption can be substituted with another problem, determining which mmWave node should be

turned off. When a mmWave node is turned off, users around the node must be accommodated by LTE macro BS in order to maintain services. Thus, the more users LTE macro BS accommodates, the more mmWave nodes can be turned off. Step (1) calculates aggregated traffic load of each mmWave node coverage and sorts by the amount. Then, LTE macro BS accommodates in order from users around nodes with lower traffic load. We can turn off as many nodes as possible by offloading to LTE macro BS within the available bandwidth.

2. Initial path creation over mmWave meshed network.
   mmWave backhaul paths should support all the nodes activated in step (1), and should be shortest path from the point of view of power consumption. We can find the shortest path with the help of graph theory if the source and destination are given. As each sector of mmWave gateway can play a role of a source, and each mmWave node can be a destination, the problem of backhaul paths generation is equal to determining which gateway sector should be the source for each node. Step (2) allocates each node to the appropriate gateway sector which can arrive with the smallest number of hops. If there are large traffic demand in specific area which is larger than the capacity of gateway sector, step (2) distributes them into several gateway sectors. Finally backhaul paths are formed from determined source to destination through activated nodes in step (1).

3. Energy efficient mmWave node re-activation for relay.
   By step (2), almost all nodes can get backhaul path. However, some nodes may be isolated when all surrounding nodes are turned off in step (1). Step (3) re-activates some nodes in order to assure backhaul path for such isolates nodes. It is better to minimize the number of re-activated nodes in terms of power consumption. Thus, step (3) searches all combination of re-activated nodes which is on the shortest path from source gateway sector to the isolated node and selects one which re-activates the smallest number of nodes.

5.3 KPIs

Table 15: list of KPIs addressed by mmWave mesh EMMA

Application	***mmWave mesh EMMA***
List of related project KPIs	***Obj.7: Design essential Xhaul-integrated (control/planning) applications*** *(5GPP KPI) Reduce energy consumption in the 5G-Crosshaul by 30% through energy management*

Measurement of project KPIs as well as application specific ones	**KPI metrics (unit)**	**Description**	**Way of measurement**
	Energy reduction [%]	This KPI is used for evaluating the reduction in energy usage through proposed EMMA algorithm.	Based on simulation result, estimate power-consumption in the network
	system satisfaction ratio [%]	Evaluate average user satisfaction by changing variance of user distribution Proposed algorithm keeps 100% satisfaction ratio even in super densely located user scenario owing to route multiplexing	User traffic demand in simulation has to be satisfied and supported by mmWave mesh
List of the State-of-the-Art approach **(used to compare with proposed solutions)**	1. State of the art approach 1: Always on This is the usual approach for dense networks where all available nodes are on all the time. The energy consumption is at a high base level and does not change much if traffic is send or received. If a user is using the node or not is not of relevance in this approach. 2. State of the art approach 2: user based on This approach takes into account nodes that do not serve any user. These idle nodes can then be turned off to reduce the energy consumption of the network. Here, the traffic demand is not considered. Hence even an idle user within the cell will result in turning on the node.		
Cost Models	Energy model	Energy model is based on SoTA mmWave equipment but can easily adapt for given system, reduction shown by proposed algorithm in relative terms.	

5.4 Validation and Evaluation

The validation and evaluation of the EMMA mmWave mesh algorithm for 5G-Crosshaul is a two-stage process. The first stage is a simulation within WP4; the second stage is an experimental demonstration of the core EMMA functionalities and integration within a real architecture in WP5. For validation and evaluation, a given deployment scenario is created within MATLAB. An example is shown in Figure 32. The mmWave node plays the role of both XFE (relay) and (mmWave) access with three

or four sectors in both access and backhaul/fronthaul. The LTE macro BS plays a role of the mmWave gateway as well in the cell to accommodate time-variant and spatially non-uniform traffic by forming a mmWave mesh network. A principle objective of the EMMA is to reduce energy consumption of mmWave mesh network by switching off mmWave nodes as much as possible in an area with small traffic demand. We compare the results of EMMA simulation with defined SoTA approaches. We compare the multihop-proposal with single-hop approach for different distances from the central gateway unit. Furthermore we show that the energy consumption for different user-distributions and resulting traffic demands are decreased with our EMMA algorithm.

To model the users within the network, a 2-dimensional Gaussian, user distribution function is used. The hotspot of this function in Figure 32 is at x=200m, y=0m and the standard derivation σ=100m.

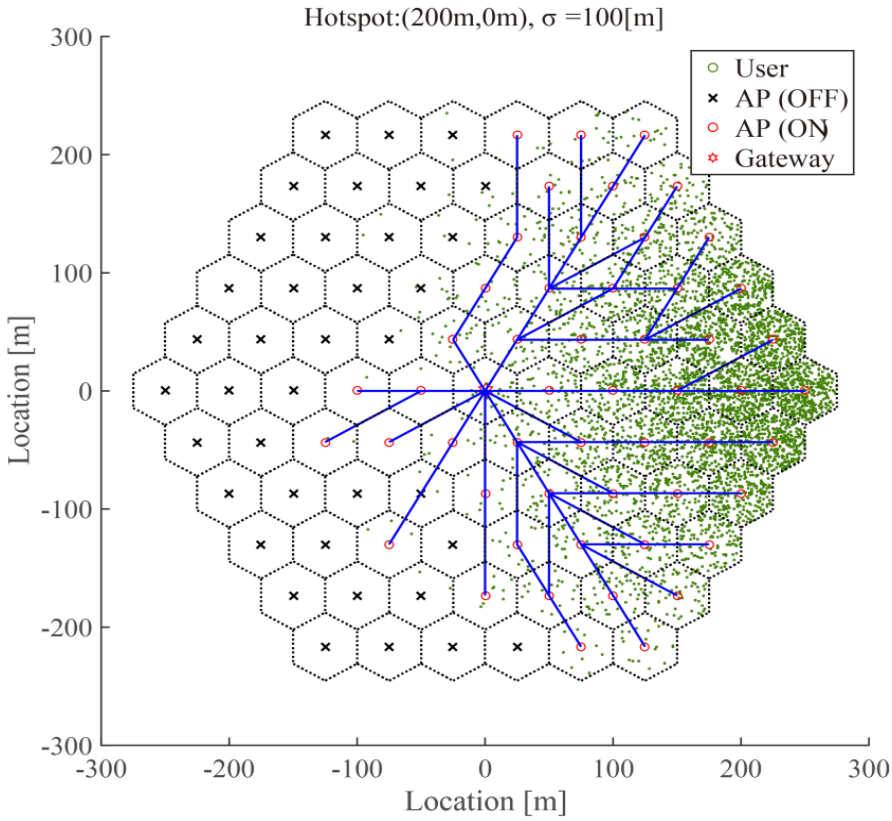


Figure 32: Example of deployment scenario and EMMA solution for given user distribution

For comparing the different approaches, we define a system satisfaction ratio (SSR) by: Supplied Backhaul Data / Demanded Backhaul Data. Changing the σ value of the user-distribution and comparing our multihop solution to a given single-hop solution.

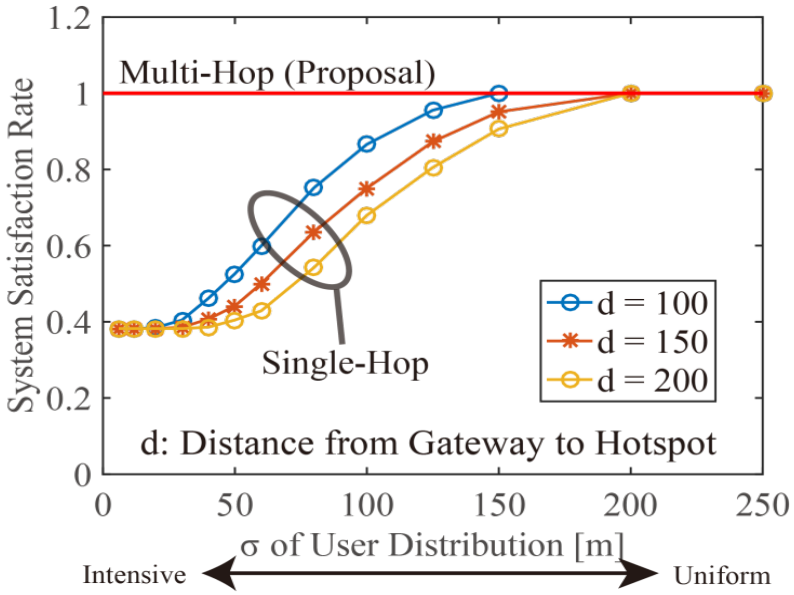


Figure 33: Results of comparing SSR single-hop with multi-hop approach

The results can be seen in Figure 32. Depending on the distance d from the central gateway one can see an improvement in SSR for single-hop. Our multi-hop proposal outperforms the single-hop solution while maintaining an SSR of 100% for any given σ.

Looking at the energy consumption of the mmWave mesh network and comparing it to the always ON and user-centric ON approach the EMMA algorithm outperforms both. In Figure 34, the performance of the proposed algorithm is shown. The energy consumption compared to the always ON approach is reduced by over 39% with EMMA.

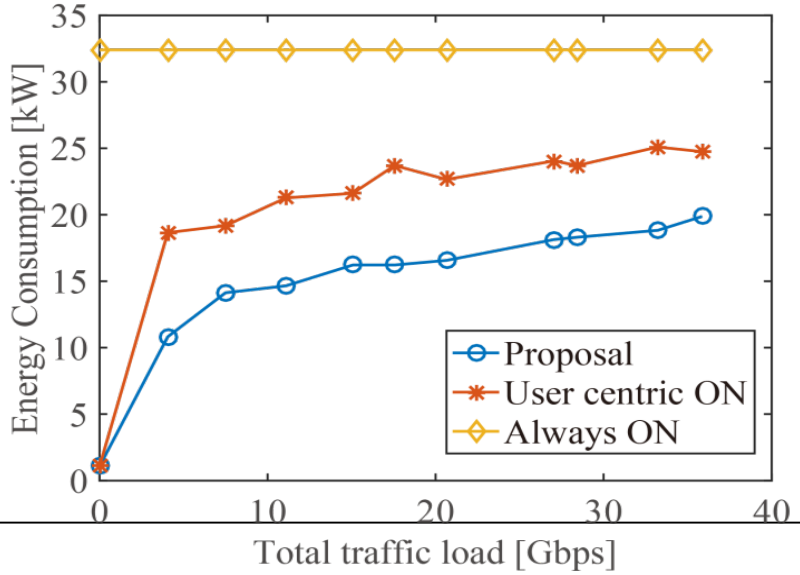


Figure 34: Energy consumption depending on total traffic load of the network

We have shown that even with very high traffic demand and dense user distribution the EMMA algorithm keeps user satisfaction to 100% while reducing energy consumption significantly. It is planned to study the proposed algorithm in more detail in a practical scenario within WP5 and verify the simulation results in real world scenarios and experiments.

6 Energy management and monitoring Application (EMMA) for High Speed Train Scenario

To provide high-speed train communications, we can deploy either general purpose or dedicated systems. In dedicated systems, the transceiving equipment deployed along railway track spend most of the time waiting for passing-by high-speed train to serve, which implies a great deal of waste in energy consumption and operation cost. If all or part of them can be turned on only when needed, system energy efficiency and OPEX can be improved.

Among the transceiving nodes deployed along high-speed rail track, Radio-over-Fibre (RoF) technology is used to extend Radio Frequency (RF) signals from base stations (BS), mostly to tunnels, in a transparent manner. EMMA controls the power state of RoF according to the presence of a high-speed train in close proximity of the nodes reported by cloud database. The application signals to the XCI that idle RoF nodes should be switched off when unused. The goal is to minimize the energy footprint of the deployed distributed RoF nodes in the Crosshaul network without degrading the QoS of ground-to-train communication. EMMA for high-speed train is composed of the following three modules:

- *Context Information Module* is in charge of gathering the context information related to train mobility and determining the current location of the train. It collects the information from the eNBs and stores the information in the database;
- *Statistics Module* which is in charge of storage and retrieval of context information. It allows the XCI to periodically retrieve new records from the database and updates the XCI about the current location of the train;
- *Management Module* which is responsible for the actual control of RoF nodes. It decides to switch on or off the RoF nodes via SNMP protocol based on the received context information.

6.1 Consolidated design

EMMA Algorithm is classified into following two parts:

- Information Parsing: context information module parses the context information receives the log information from each eNB $\boldsymbol{E_i}$ w.r.t Train ($\boldsymbol{T_i}$). Context Information is parsed $\boldsymbol{C_p}$ and relevant information is extracted and transmitted to the database ***db***.
- Statistics Module retrieves the new ***db*** entries and triggers Management module to decide upon the reception of received ***db*** entry whether to switch ON/OFF the connected RoF nodes. Management module switches ON the RoF nodes if Train is within the coverage of RoF nodes $\boldsymbol{R_j}$ and switches OFF if Train is outside the coverage of RoF nodes $\boldsymbol{R_j}$

A pseudocode of the algorithm is illustrated in Algorithm 1:

Algorithm 1: EMMA Algorithm

Input: Train mobility information C
Output: Power status

```
Information parsing;
for each train T_i do
    for each eNB E_i do
        C_p ← parsed context information of T_i;
        w.r.t E_i;
        DB ← Transmit C_p;
Power management;
for each new DB entry do
    if T_i is within the coverage of R_j + Guard distance g
      then
        Power status ← ON;
    if T_i is within the coverage of R_j + Guard distance g
      then
        Power status ← OFF;
return Power status
```

Algorithm 1: EMMA algorithm for High-Speed Train

6.2 Implementation

The EMMA application for the high speed train scenario is developed as a Python application implemented from scratch and operating on top of the 5G-Crosshaul XCI services, utilizing the XCI service by REST APIs. It implements, as a main functionality, the correlation of mobility-related information of the High-Speed Train and energy-related information of Analogue Radio-over-Fibre (RoF), exposed by the XCI, to provide an overall view to the EMMA application. Based on this received information, EMMA will adjust the status of the connected RoF nodes (e.g., switching ON the set of connected nodes if an High-Speed Train is predicted in the coverage area of those connected RoF nodes).

The context information module collects the information from the surrounding eNBs and predicts the current location of the train, which later is posted to the Statistics

Module which analyses the information and triggers the Management module. The management module decides whether to switch ON/OFF the connected RoF nodes based on the received information.

6.2.1 High-level design

The high-level design of EMMA for the High-Speed Train is illustrated in Figure 35.

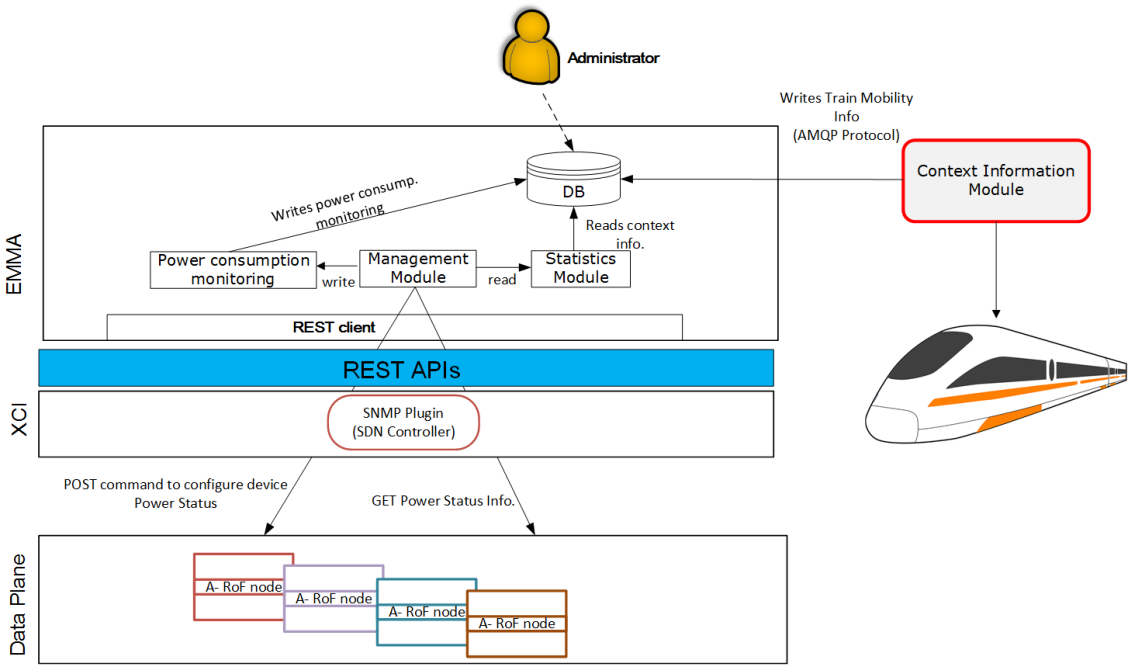


Figure 35: EMMA high-level software design

6.2.2 Implementation Details

The main internal components in Figure 35 are described in the table below.

Table 16: EMMA application components

Component	Description
REST Client	Entity in charge of interacting with the XCI components via REST APIs.
Power consumption monitoring	Entity in charge of retrieving the collection of power consumption information from RoF nodes (via Management module), processing and aggregating this information and storing and organizing the results in DB.
DB	Internal Mysql DB to store power consumption information of RoF nodes and context information.
Statistics Module	Entity in charge of on demand retrieval and analysing the context information from the DB.
Management Module	Entity in charge of computing the power status of RoF node based on the retrieved context information from DB (via Statistics Module).

Context Information	Entity in charge of detecting the location of high-speed train by utilizing the RAN context information.

6.3 KPIs

Table 17: List of KPIs addressed by EMMA for HST

<table>
<tr><td>Application</td><td colspan="3">EMMA for HST</td></tr>
<tr><td>List of related project KPIs</td><td colspan="3">Obj.7: Design essential Xhaul-integrated (control/planning) applications

(5GPP KPI) Reduce energy consumption in the 5G-Crosshaul by 30% through energy management</td></tr>
<tr><td rowspan="2">Measurement of project KPIs as well as application specific ones</td><td>KPI metrics (unit)</td><td>Description</td><td>Way of measurement</td></tr>
<tr><td>Energy saving (%)</td><td>Percentage of energy efficiency/savings (i.e. average power consumption per flow)</td><td>Emulative assessments.</td></tr>
<tr><td>List of the State-of-the-Are approach</td><td colspan="3">State of the art approach:
Current deployment of Analogue Radio-over-fibre, with all nodes always on.
EMMA introduces a mechanism to change the power state of connected A-RoF nodes based on the location of the high-speed train.</td></tr>
<tr><td>Cost Models</td><td>Energy model</td><td colspan="2">Assume there are R_N north-bound and R_S south-bound trains in a day, and the time required to serve each passing-by train is T_S seconds, then the percentage of time a RoF node is idle in a day can be expressed as:
$1 - \frac{(R_N + R_S) \cdot T_S}{86400}$.
The percentage is also the optimum energy saving we can achieve using EMMA.</td></tr>
</table>

6.4 Validation and Evaluation

6.4.1 Scenarios Description

The high-speed train scenario is a special use case of EMMA to control the status of a large number of RoF nodes along the high-speed train track. A typical scenario is shown in Figure 36, where each train pushes its location information to a cloud database, and a central controller utilizes the information to maintain the status of all RoF nodes along the rail track. It is assumed that the RoF nodes are connected to eNB B and C. When the train is in the coverage of an eNB, the communication gateway of the train can receive information, such as Physical Cell ID (PCI), and save it to the IPC (Industrial Personal Computer) server installed on the train. IPC extracts the relevant information and posts it to the cloud database. Cloud database notifies the EMMA upon reception of new entries. After the retrieval of records, the EMMA decides, based on the mapping table, which RoF nodes should be turned ON or OFF. For instance, as the train approaches eNB B, the EMMA via SNMP turns ON the RoF nodes and, as it leaves eNB C, it turns them OFF. In this way, we minimise the energy footprint of the deployed distributed RoF nodes by leveraging the 5G-Crosshaul network without degrading the QoS of ground-to-train communications.

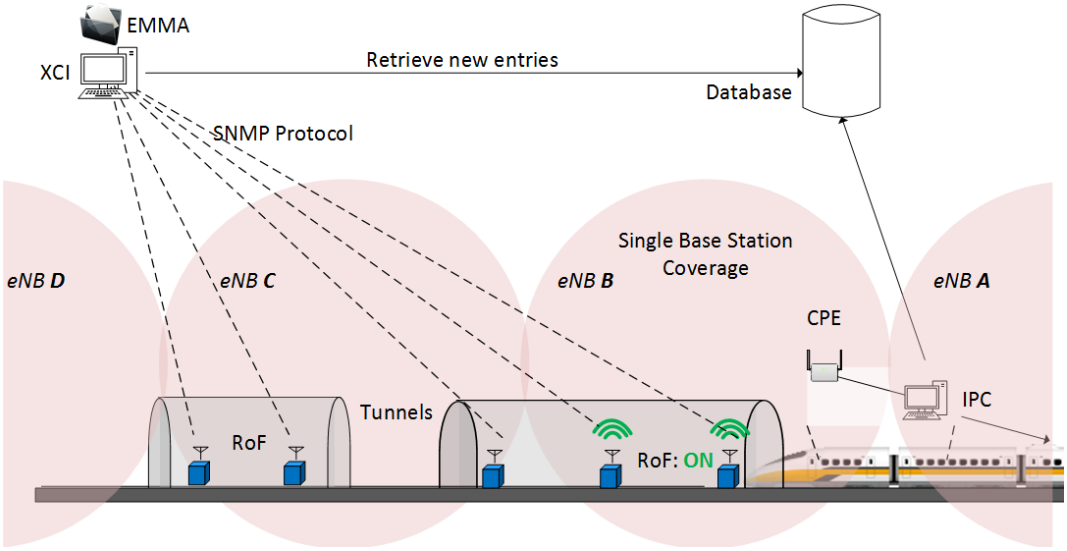


Figure 36: Scenario of EMMA-specific High-speed Train.

6.4.2 Evaluation Environment

Emulation is carried out in lab, as illustrated in Figure 37, with the help of XCI connected to the real RoF nodes and IPC server. A virtual machine acts as an eNB and sends the context information to another virtual machine (IPC). The data will be parsed, and only the relevant information is directed to the database with the help of a

messaging module. In this case, AMQP defined RabbitMQ is used as the messenger. Management module of EMMA embedded or on top of the XCI will retrieve the new values from the Cloud DB and decides (based on the context information regarding mobility) to switch ON/OFF the RoF nodes via SNMP plugin. In the emulated environment, the behavior of RoF nodes is characterized with the help of a computer program as a receiver of the decision of EMMA application and will act accordingly.

The emulated environment consists of:

- ODL (XCI) integrating SNMP Driver
- EMMA application consisting of Management, Statistics and Context information module, sitting on top of ODL (XCI) and using the REST APIs
- RoF nodes

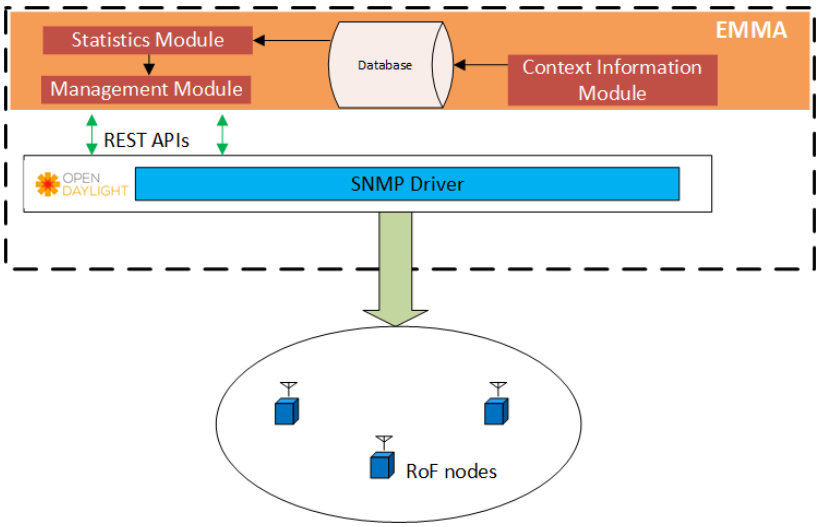


Figure 37: In Lab emulated environment

Figure 38 represents the comparison of energy consumption of RoF nodes with and without EMMA. The x-axis represents the energy consumption per hour in percentage. With the EMMA-integrated solution, RoF nodes will be switched on only to serve the high-speed train when it is approaching, thus saving significant energy. Energy consumption for RoF nodes can save up to 90% compared to the current usage based on RoF usage and train's operational time.

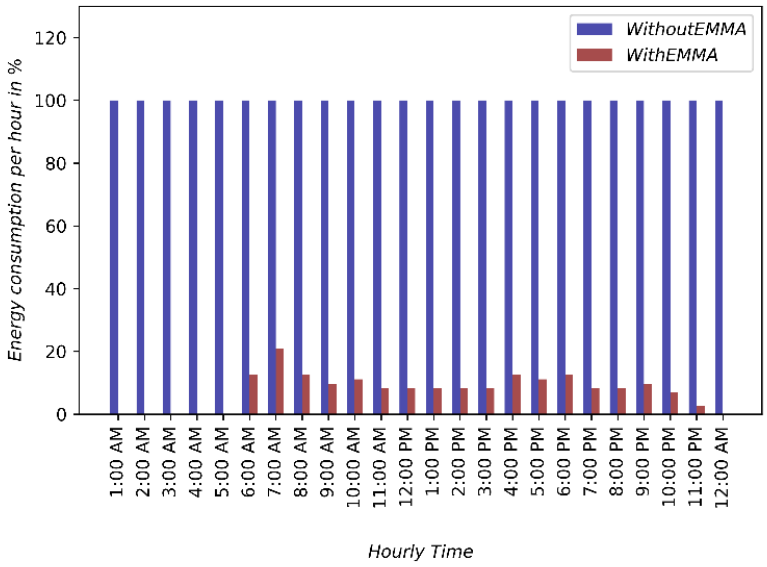


Figure 38: Comparison of energy consumption without (blue) and with (red) EMMA

7 Energy management and monitoring Application (EMMA) for multi-tier networks

The goal of this application is to implement a two-tier architecture for the energy and radio resource management (RRM) of cellular networks with mmWave backhaul and energy harvesting capabilities. In fact, nowadays the greenhouse gases emissions of ICT ecosystems already consume about 1500 TWh of energy annually, approaching 10% of the world's electricity generation and accounting for 2-4% of the carbon footprint due to human activities. The trend is even worst considering 5G, since mobile network operators will have to handle a much higher capacity demand [22], especially within urban areas, where it will need to support up to 1000x more capacity with respect to 4G. To this respect, it is estimated that the energy consumption of ICT might reach the 51% of global electricity production by 2030, which would translate in contributing to up to the 23% of the globally released greenhouse gas (GHG) emissions [23].

Recently, the research community put a lot of effort for reducing the energy consumption of the radio communication networks, as testified by the vivid literature on the topic of energy efficiency (EE) [24]. One of the most promising solutions is represented by the dynamic *sleep mod*e (also called *switch ON-OFF*). However, this technique can guarantee only reduced energy savings [25]. According to this and considering the high-energy demand, the introduction of energy harvesting (EH) capabilities can represent a viable paradigm for enabling higher energy savings [26]. These works integrate the problem of EE by considering the erratic and intermittent nature of the renewable energy sources (RES) and the constrained amount of energy that can be stored. Up to now, the efforts concentrate in providing guidelines for dimensioning the network and the energy harvesting systems, while on-line approaches to control network elements have appeared only recently. In [27] the ski rental problem has been adapted to the ultra-dense small BS scenario for determining when to switch OFF. The analysis is carried out considering Poisson arrivals for energy and traffic, which may provide a non-realistic approximation to these processes. A solution based on Reinforcement Learning (RL) is presented in [28], where the performance of a single SBS has been optimized as a function of the local RES and storage conditions. However, the impact of the backhaul technology adopted is not being assessed. Moreover, the effect of multiple SBSs simultaneously switching OFFs within the same area is not considered.

In order to deal with this, we propose a solution considering a two-tier architecture where macro BS are supported by small BS (SBSs). SBS network is modeled as a multi-agent system where each local agent (SBS) makes autonomous decisions, according to a Decentralized Self Organized Network (D-SON) paradigm. SBSs equipped with solar energy and batteries constitute an overlay layer in a two-tier network where macro BSs are powered by the electricity grid. In WP2, the behavior of SBSs has been optimized to offload the traffic from the macro BS according to the

energy income and the traffic demand. To this purpose, we proposed a distributed on-line solution based on multi-agent reinforcement learning (RL), known as *distributed Q-learning* (QL) [29], which allows SBSs to independently learn a radio resource management (RRM) policy. Therefore, the local agent in each SBS dynamically learn when to switch ON and OFF both the backhaul and the access according to the available harvested energy budget, the user traffic demand and the energy consumption of the node. QL is designed to learn a policy for optimizing the system performance in terms of throughput and energy efficiency by directly interacting with the observed environment. In WP2, we analyzed the behavior of this solution in a single small BS scenario obtaining encouraging results. Thanks to energy monitoring functionalities of EMMA, we can design a dedicated EMMA application for multi-tier networks with energy harvesting capabilities (EMMA-EH) that considers network-wide statistics for designing more energy efficient (EE) RRM policies interacting with the local ones. The goal of EMMA-EH is to improve the system EE and reduce its drop rate by assisting the SBS local agents in their learning process. In detail, EMMA-EH can influece the learning and decision making processes of the local QL agents according to the actual network conditions (e.g., system load, batteries level). This allows to mitigate the well known conflict problem typical of the distributed learning systems. For doing this we rely on the Heuristically Accelerated QL (HAQL) framework [30]. The idea of HAQL is to use additional knowledge from the environment which is not included in the learning process for influencing the action choices of the learning agent in order toward a more efficient global system policy.

7.1 Consolidated design

One of the main task of the EMMA-EH application is to monitor the status of the system for detecting whether the system might be in outage or not (i.e., it is not able to satisfy the UEs demand) according to the local policies selected by the distributed SBS agents. In detail, it is charge of estimating whether the system will be able to manage the requested traffic according to the number of the SBSs that will be switched OFF in the next time frame. For doing this, we propose to use machine learning techniques (ML) in order to be able to consider the environmental variables of each SBSs (e.g., different positions respect to the macro, different traffic profiles, different coverage ranges, etc.). To this respect we considered the use of Multilayer Feedforward Neural Networks (MFNN) [31] [32]. The basic element of the MFNN is represented by the neuron (also called perceptron) which consists of a linear combination of fixed nonlinear functions $\theta_j(x)$, in detail for a vector of input $x_i, i = 1 \dots M$, takes the form:

$$y(x, w) = f\left(\sum_{j=1}^{M} w_j \theta_j(x)\right) \quad (1)$$

where w_i are the weights associated to each input and $f(\cdot)$ is a nonlinear activation function, typically the sigmoid function $f(x) = 1/(1 + e^{-x})$. A MFNN is composed by a series of neurons organized in *N* layers in a way to allow the input information to move only in one direction (there are no cycle in the networks like in a recurrent neural network). Let define N_l as the number of neurons in layer *l*. The bottom layer, N_0, is the input layer and it contains *M+1* neurons which are the inputs plus the "constant" neuron always at 1. The last layer is composed by only one neuron and represents the output of the neural network. Each neuron in a layer $l = 2, \dots, N$ has $M_l = N_{l-1}$ inputs, each of which is connected to the output of a neuron in the previous layer. Layers *2,..N-1* are called *hidden layers*. A MFNN can approximate arbitrary continuous functions defined over compact subsets of $\mathbb{R}^M$ by using a sufficient number of neurons at the hidden layer. In order to achieve this it is necessary to determine the values of the weights correspondent to the function to be approximated, operation called *network training*. In this work, MFNN are used with supervised learning, since a training set of input-output is used to train the network according to the *backpropagation algorithm* [31].

The approximation capabilities of the MFNN have been used for evaluating the system load conditions. More in detail, we want to estimate the macro load normalized factor L^{macro} as function of the load of each SBS L_i and of their policy π_i. After testing different input configuration solutions, the one that returned the best performance was obtained by combining the two input variables L_i and π_i in one load-policy factor LP_i defined as follows:

$$LP_i = \begin{cases} L_i & \pi_i = 1 \ (SBS \ is \ ON) \\ P_f \times L_i & \pi_i = 0 \ (SBS \ is \ OFF) \end{cases} \quad (2)$$

where P_f is a policy factor used to consider the effect of the load of the small BS when it has to be managed by the macro. We considered a two hidden layers MFNN and we analyzed it for different number of neurons per layer in order to obtain the best estimation performance further details will be provided in Section 7.4.

Defining as L_m^t as the load of the macro BS at the time *t*, we can identify two different operational cases for the EMMA-EH for avoiding the system to have suboptimal performance: i) the system is under-dimensioned (i.e., $L_m^t > L_m^{thrHigh}$) and ii) the system is over-dimensioned (i.e., $L_m^t < L_m^{thrLow}$) and it will use more energy respect to the optimum. In both cases the EMMA-EH can help the learning process guiding the system toward a more energy efficient global solution by influencing the learning process of the local agent at each SBS to adopt a different solution, when possible (i.e., SBSs with scarce energy cannot be used anyway). It is to be noted that, while the former is aimed at reducing the system drop rate, the second, affecting the energy, might have a strong impact on the distributed QL layer. In fact, the main goal of the QL algorithm is to dynamically switch on and off the SBS according to the available harvested energy budget, the user traffic demand and the energy consumption of the SBS. For doing this, QL learns a policy for optimizing the system performance as the

throughput and energy efficiency by directly interacting with the observed environment. In distributed QL each agent *i* maintains a local policy and a local Q-function $Q(x_t^i, a_t^i)$ that only depends on its state x_t^i and actions a_t^i, with *t* being the decision epoch (time). Briefly, the QL algorithm of an agent *i* at time *t* is defined as follow:

- *State:* $x_i^t = \{S_i^t, B_i^t, L_i^t\}$
- *Action set:* $a_i^t = \{ON, OFF\}$
- *Reward function:*

$$r_i^t = \begin{cases} 0 & B_i^t < B_{th}^{LOW} \text{ or } D < D_{th} \\ kT_i^t & B_i^t \geq B_{th}^{LOW} \text{ and } D \geq D_{th} \text{ and SBS is ON} \\ \dfrac{1}{B_i^t} & B_i^t \geq B_{th}^{LOW} \text{ and } D \geq D_{th} \text{ and SBS is OFF} \end{cases}$$

where S_i^t is the harvested energy, B_i^t is the battery level and L_i^t is the load requested in the SBS. *D* is the traffic dropped by the network (i.e., non-served users), T_i^t is the throughput of the SBS. D_{th} is the threshold on system drop-rate (i.e., macro and SBS), while B_{th}^{LOW} is the security threshold of the battery state of charge (SOC). Interested readers can refer to [33] for a more comprehensive description. According to this, a proper mechanism has to be adopted for interacting with QL, such as the HAQL) [30]. In QL the optimal action $\hat{a}$ to be taken by the SBS *i* at time *t* in a certain state x_i^t is defined as $\hat{a} = \underset{a_i}{\text{argmax}}(Q(x_i^t, a_i^t))$. The idea of HAQL is to use a heuristic function $H(x_i^t, a_i^t)$ derived from additional knowledge which is not included in the learning process of QL (i.e., in its state variables x_i^t) influencing the action choices of the learning agent in order to modify its current policy $\pi(x_i^t)$ for guiding the convergence. The new combined policy selection formula is:

$$\pi(x_i^t) = \underset{a_i}{\text{argmax}}(Q(x_i^t, a_i^t) + H(x_i^t, a_i^t))$$

At each time step 1 hour long, EMMA-EH receives the most important statistics that the local SND controllers collected from the local agents (i.e., the load of the SBSs L_i, their battery level, their policy π_i and representative parameters of Q-learning agents like the maximum values of the Q-functions Q_i^{max}). According to these parameters, the MFNN can estimate whether the system is well dimensioned or not. In case the system is under-dimensioned, EMMA-EH at each time epoch *t* defines the set SBS_{ON}^t with the SBSs that are going to be switched OFF by the QL that have enough battery (i.e., the intantaneous batter level of the SBS *i* at time *t* is above a specific threshold: $B_i^t \geq B_{th}^{LOW}$). From SBS_{ON}^t , it selects the sub-set $SBS_{ON \to OFF}^t$ which includes the ones with the lowest Q_i^{max} values that if switched ON would avoid dropping in the system and stores the corresponding Q_i^{max} values. The idea behind this solution is to promote the system

sustainability by influencing the SBSs that have the lowest level of convenience in selecting the action of switching OFF in order to follow as much as possible the QL learned policies. Alternatively, when the system is over-dimensioned, EMMA-EH defines the set SBS_{OFF}^{t} with the SBSs that are going to be switched ON by QL that have scarse energy reserve (i.e., $B_i^t \leq B_{th}^{OFF}$). From SBS_{OFF}^{t} it selects the sub-set $SBS_{OFF\rightarrow ON}^{t}$ containing the SBSs with the lowest lowest Q_i^{max} values that would not over-load the system and stores the corresponding Q_i^{max} values. In this case, EMMA-EH will influence the subset of SBS with scarse battery reserve to switching OFF taking care to avoid to overload the macro BS. In case the system configuration can be improved respect to the one proposed by the local agents (i.e., either $SBS_{ON\rightarrow OFF}^{t}$ or $SBS_{OFF\rightarrow ON}^{t}$ is a non empty set), EMMA-EH will return the most appropriate set of $H(x_i^t, a_i^t)$ that can influence the SBS policies according to the QL parameters of the local agents. In detail, $H(x_i^t, a_i^t)$ will contain the value $-Q_i^{max}$ for each SBS that needs to be influenced and 0 elsewhere. In fact, $H(x_i^t, a_i^t)$ must be the lowest value that can influence the choice of action in order to minimize the loss in the Q-functions due to the use of the heuristics. Finally, EMMA-EH communicates to the local SND controllers the vector $H(x_i^t, a_i^t)$ which will influence the local SBS agents when running the local QL algorithm.

A sketch of the proposed architecture is provided in Figure 39.

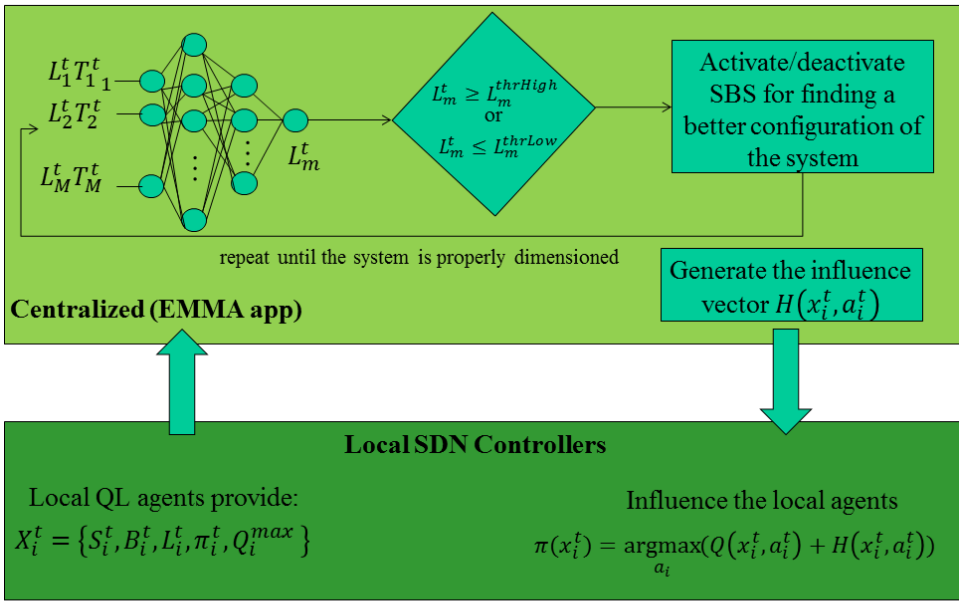


Figure 39: EMMA-EH architecture overview.

7.2 Implementation

EMMA-EH has been evaluated by simulation in order to reproduce scenarios with multiple SBS operating simultaneously with realistic traffic conditions over large time ranges (e.g., one simulated year) for collecting the behaviour of the system and of the algorithm when varying the energy harvesting conditions. More in detail, the evaluation environment is an octave based simulator which carefully models the channel condition, the LTE system and the energy harvesting phenomena. We adopted this solution since we need a tailored tool able to balance the trade-off between the accuracy and the computational complexity, especially considering the large simulated time requirements. The simulator has been already used in the framework of the EU H2020 project SANSA for evaluating backhaul routing strategies in network of SBSs with energy harvesting capabilities [34]. In this paper, a similar version of the local QL agent algorithm has been integrated with a backhaul routing protocol. The proposed solution has been evaluated both with the octave simulator and the well-known Network Simulator 3 (ns-3) [35] . The results presented show that the two simulation tools return comparable results for what concern the LTE behavior of the scenario investigated.

7.3 KPIs

Table 18: List of KPIs addressed by EMMA-EH

<table>
<tr><td>List of related project KPIs</td><td colspan="3">Obj.5: Increase cost-effectiveness of transport technologies for ultra-dense access networks

(5GPP KPI) Reduce energy cost per bit by a factor of 10
Obj.7: Design essential Xhaul-integrated (control/planning) applications

(5GPP KPI) Reduce energy consumption in the 5G-Crosshaul by 30% through energy management</td></tr>
<tr><td rowspan="3">Measurement of project KPIs as well as application specific ones</td><td>KPI metrics (unit)</td><td>Description</td><td>Way of measurement</td></tr>
<tr><td>Energy efficiency [%]</td><td>This KPI is used for evaluating the efficient utilization of the energy resources as function of the overall amount of data transmitted.</td><td>Simulative assessments.</td></tr>
<tr><td>Average link and RAN capacity utilization [% and normalized values]</td><td>This KPI is used for evaluating and comparing the resource utilization when different energy management policies</td><td>Simulative assessments.</td></tr>
</table>

		are considered.	
	System drop rate [%]	This KPI is used for the evaluation of the blocking probability of the flows in the context of limited energy resources (i.e., SBSs switched OFF or without energy in the battery)	Simulative assessments.
List of the State-of-the-Are approach **(used to compare with proposed solutions)**	3. State of the art approach 1: grid This represents the current deployment situation, where both the macro and the SBSs are powered with the standard energy grid. This solution does not provide any energy effiicency reference value, but it is important for understanding the gap of introducing the energy harvesting capabilities. 4. State of the art approach 2: greedy This approach represents the baseline solution where the SBSs is which is switched off when its battery is below the security threshold. This solution helps in understanding the gap with a solution that instantaneously uses the harvested energy when available in the storage system respect to an optimized one. 5. State of the art approach 2: q-learning This approach is the standard distributed solution evaluated in WP2 based on distributed Q-learning, where the local agents have only an indirect view of the whole system performances.		
Cost Models	Energy model	The energy model used for the RAN will be the one presented in EARTH project [36], while for the mmWave backhaul link we will consider a static value based on product datasheets.	

7.4 Validation and Evaluation

7.4.1 Scenarios Descriptions

In this report we start analysing the EMMA-EH by evaluating the behaviour of its MFNN estimator considering different neurons configurations in order to find the best solution to be used in the application. We further continue the evaluation by presenting the network performance of the EMMA-EH. The scenario considered in this analysis is

composed of a macro BS with a varying number of SBSs equipped with an mmWave backhaul link to the core, which is providing capacity extension to a macro cell. Adopted traffic demand is of an urban business district and it is based on the model proposed in [36], with higher intensity during the day-time hours (11am to 6 pm). The power consumption depends on the traffic load and/or transmission power. The approximated linear model defined in [36] has been adopted. The electrical grid powers the macro base station. The SBS is supplied by an array of 16x16 solar cells of Panasonic N235B solar modules (area 4.48 m^2), which have single cell efficiencies of about 21%, delivering about 186W/m^2 and a lithium ion battery of 2KWh. The size of the energy supplier/storage has been dimensioned based on the typical power consumption of the LTE interface and of the IEEE 802.11ac/ad cards present in the wireless segment of the Hierarchical multi-domain resource management of the 5G-Crosshaul proof of concept described in D5.1 [37]. The evaluation environment is an octave based simulator which carefully models the channel condition, the LTE system and the energy harvesting phenomena. We adopted this solution since we need a tailored tool able to meet the trade-off between the accuracy and the computational complexity, since we need to simulate at least one year of lifetime of the system in order to capture the behavior of the algorithm when varying the energy harvesting conditions. The simulator has been already used in the framework of the EU H2020 project SANSA for evaluating backhaul routing strategies in network of SBSs with energy harvesting capabilities [34]. In this paper, a similar version of the local QL agent algorithm has been integrated with a backhaul routing protocol. The proposed solution has been evaluated both with the octave simulator and the well-known Network Simulator 3 (ns-3) [35] . The results presented show that the two simulation tools return comparable results for what concern the LTE behavior of the scenario investigated.

7.4.2 MFNN Validation Results

In this section we start reporting the results obtained when analyzing the behavior of different MFNN configurations. We tested the MFNN with different values of P_f obtaining that the value that provides the best approximation results is 20. In parallel, we tested also different number of neurons in each hidden layer considering MFNN with only few neurons per layer up to MFNN with twice the number of inputs, which is the number of small BSs N_{SBS}. The configuration that showed the best approximation has been the one that uses $N_1 = \lceil {}^3/_2 \, N_{SBS} \rceil$ and $N_2 = \lceil {}^2/_3 \, N_{SBS} \rceil$. This configuration has been used for obtaining the results presented in what following. We run the simulation for one simulated year and with different number of deployed SBSs.

Figure 40 presents the overall mean square error as a function of the time-step of the MFNN, which is of 1 hour as defined in Section 7.1. After a preliminary phase of 500 hours, all the MFNNs reach a stable behaviour. However, when the number of SBSs increases, the function to be approximated becomes more complex since its domain increases in dimension. This effect is reflected in the value of square mean error for different number of SBSs deployed, where the one with the worst average error is for 9

SBSCs with an error of 15%, while the one with best approximation performance is for 3 SBSs with an error of less 10%. According to this, in the next figures we will report the results for 9 SBSs for analyzing the performance of the worst case scenario.

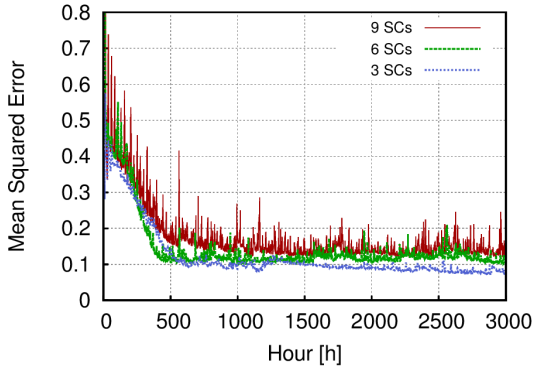


Figure 40: Mean square error of the MFNN for different number of SBSs.

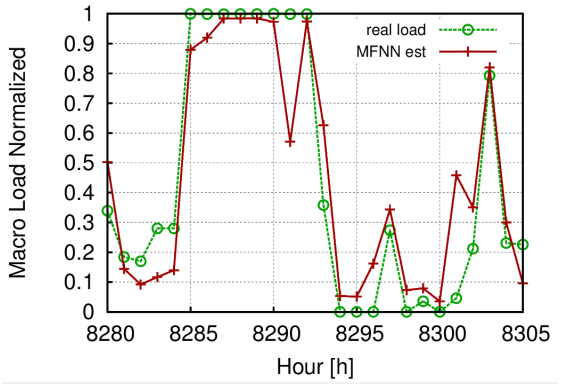


Figure 41: Macro load estimation example with the MFNN

Figure 41 shows the real macro load and the estimation performed by the MFNN. It can be noticed that the MFNN is able to follow the system dynamics but for a single point at hour 8291, which confirms the error reported in Figure 40. Thus, we evaluated the effect of this approximation error for what concern the switch ON-OFF influencer algorithm of the EMMA-EH app. To this respect, the most critical error would be to activate/deactivate SBSs that in reality should not be influenced. In fact, in case for instance a SBS that will be switched OFF is wrongly influenced for doing the opposite, the local agent will experience a delay in process of learning or might take erroneous actions in the future. For doing this, we tested two different operational cases that would generate an error in the EMMA-EH decision making process (Figure 42): the false negatives and false positives occurrences. We defined the former as the cases when the MFNN returns a $L^{macro} \leq L_m^{thrHigh}$ while the real system dropped some traffic due to overload (i.e., the MFNN has not been able to detect the overload of the system). Alternatively, the false positive has been defined as the cases when the MFNN expects a $L^{macro} > L_m^{thrHigh}$ while the system do not drop any traffic. (i.e., the MFNN wrongly detect an overload of the system). In order to maintain a safeguard margin on the drop case, $L_m^{thrHigh}$ has been chosen as 85%; this should also help in mitigating the effect of the maximum 15% of mean square error.

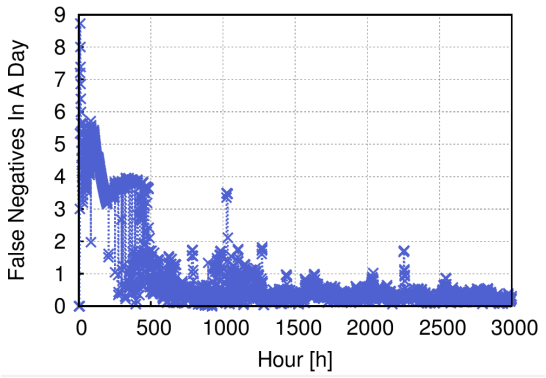


Figure 42: False negative occurrences with the EMMA-EH MFNN

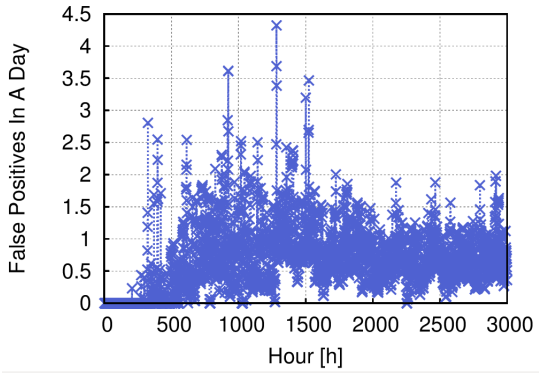


Figure 43: False positive occurrences with the EMMA-EH MFNN

Figure 42 and Figure 43 provide respectively the occurrences per day of false negatives and false positives at each step of the learning process evaluated on all the known past inputs at that stage. As we can see, after the preliminary instable phase (around 1500 hours), the MFNN is able to reduce the false negative to less than 1 per day, while the false positives are between 1 and 2. The reason behind this is that we used a guard margin in the definition of false positives for guaranteeing that the macro will not be overloaded. Therefore, some false positives are estimations that fall between $L_m^{thrHigh}$ and 100%, which may not represent a drop situation. It is to be noted, that the behaviour of the false positives is complementary respect to the one of the false negative. This is due to the fact that at the beginning the MFNN tends to return very low values of the estimated L^{macro} producing therefore a high number of false negative. Then, from 500 to 1500 hours the MFNN reaches a high level of accuracy (i.e., the mean square error is already of 15%), so false positives might happen more frequently. Finally, at 1500 hours, the MFNN arrives to the global stability by reducing the number of false positives while maintaining the false negatives and the mean square error to low levels.

7.4.3 EMMA-EH Network Performance

In what following we move to the results at networking level. We tested the network with different configuration parameters and we obtain the best results for $L_m^{thrHigh} = 85\%$, $L_m^{thrLow} = 5\%$, $B_{th}^{LOW} = 50\%$. While B_{th}^{OFF} is set to 20% for maintaining the batteries in the correct SOC operative range, in order to avoid to rapidly jeopardize the battery performance [38]. EMMA-EH will be compared with the standard QL solution and with respect to a greedy (GR) algorithm, which is switching off the SBS when its battery is below the security threshold B_{th}^{OFF}. Simulations are run during 365 simulated days spanning over different months. Considering the high difference in the harvesting process among the seasons, two representative periods are considered for presenting the results: *winter* and *summer*, respectively termed "Win" and "Sum". January, February, October, November and December are considered as *winter* months. The energy harvesting profiles have been synthetically generated with the SolarStat tool [39].

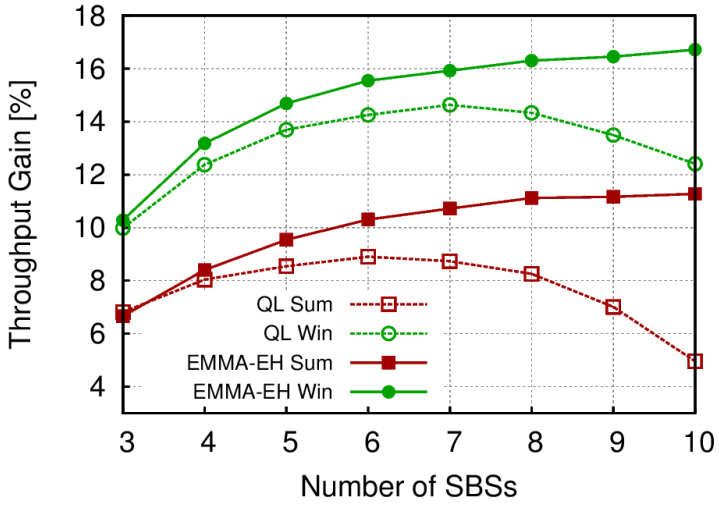


Figure 44: Average throughput gain of EMMA-EH and QL with respect the greedy scheme.

Figure 44 presents the average percentage gain in throughput of the SBSs of the EMMA-EH and QL schemes respect to the GR one by varying the number of SBSs. The throughput gain of EMMA-EH outperforms both QL and greedy. Moreover, with HAQL the throughput gain is not affected by the number of SCs. In fact, the QL shows a degradation of the performances when increasing the number of SCs. This is a typical problem of the distributed RL solutions, in which the agents might start generating policies that conflict among each other. This fact is remarked in Figure 45 where the traffic drop rate of the three schemes is presented. The QL solution already halves the drop error rate respect to the greedy which reaches 25% in *winter* and 15% in *summer*. However, EMMA-EH is able to further reduce the traffic drop rate of QL by reaching the 5% in *winter* and the 1% in *summer*.

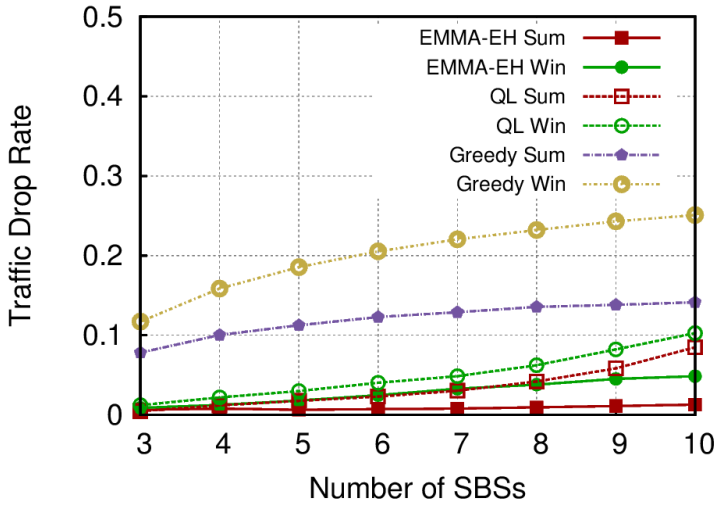


Figure 45: Traffic drop rate for greedy, QL and HAQL.

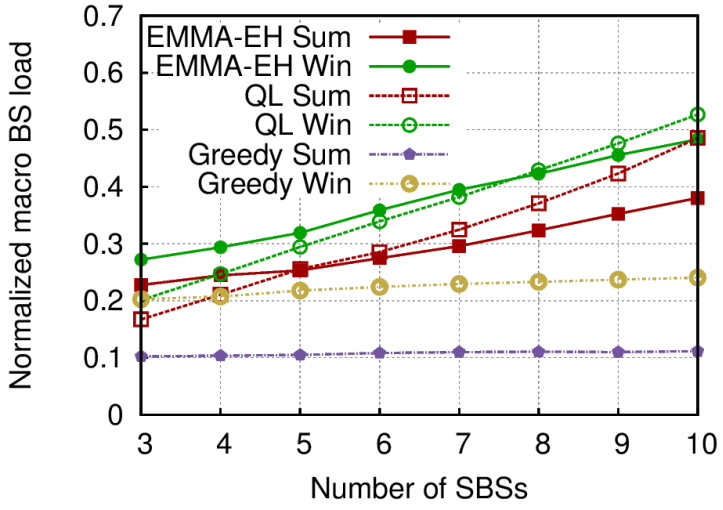


Figure 46: Average normalized macro load for greedy, QL and HAQL.

Figure 46 presents the average normalized macro BS load. Both QL and EMMA-EH generate more traffic in the macro BS in order to avoid the high drop rate of the greedy. To this respect, EMMA-EH generates less traffic respect to QL while guaranteeing better performance in the terms of drop rate. This is due to the fact that, EMMA-EH exploiting his network wide knowledges can better fit the available resources with the actual network needs. On the contrary, QL is experiencing problems of conflicts among the agents at high number of SBSs, which are not able to coordinate efficiently thus jeopardizing the performance.

Regarding the energy, we defined the EE metric as $EE = T_S/E_S$, where T_S is the system throughput and E_S is the total energy drained by the macro BS (from the power grid). Figure 47 reports the EE improvement of EMMA-EH, QL and greedy with respect to the baseline solution where both the macro BS and the SBS are powered with the grid. The EMMA-EH outperforms both greedy and QL in winter. Moreover, EMMA-EH well scales with the number of SBSs, while standard QL solution experiences degradation till performing even worst respect to the greedy one. It is to be noted that, during summer the gain is lower since the renewable source system has been dimensioned to provide the necessary energy in the worst case, which is a winter day. This implies during summer the energy reserves are abundant and both EMMA-EH and QL have fewer margins when optimizing the policy. Finally, when we look at the total amount of energy spent by the system in the worst case scenario of 10 SBSs (it is proportional to the served traffic). It approximatively amounts to 7.5 MWh in a year for a greedy solution, while it varies from 9.3 for QL to 8.8 MWh when using EMMA-EH is adopted. As a final remark, it is worth mentioning that the same system implemented without energy harvesting capabilities (i.e., where the SBSs are grid-connected) would consume 17 MWh in a year, which implies an increment in terms of used energy of more than 50%. According to this, the abundant energy has to be discared by the SBSs,

since it can neither be used for transmitting nor stored in the battery. To this respect, Figure 48 reports the abundant energy of the three schemes. As expected in summer period, the amount of the energy in excess during the day is higher for all the solutions. However, greedy has always less abundant energy. While EMMA-EH, despite guaranteeing better system performance, has less abundant energy respect to the greedy solution. This confirms that HAQL is able to better use the available energy reserves.

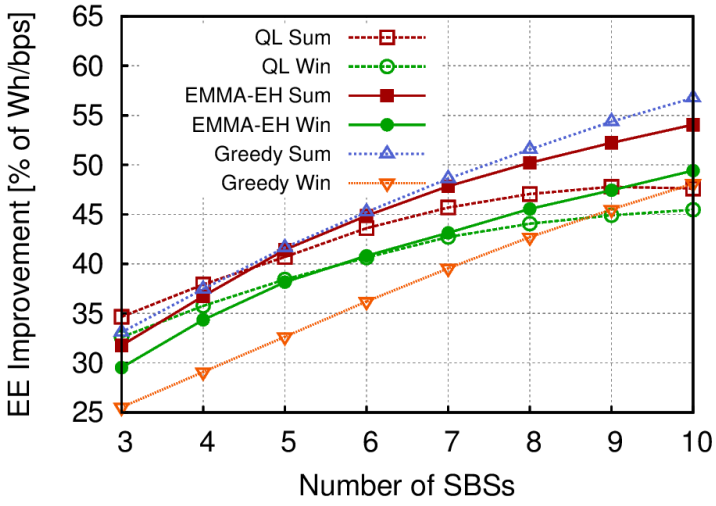


Figure 47: Average EE improvement [% of Wh/bps] of HAQL, QL and Greedy with respect to the case in which both macro BS and SBS are powered with the grid.

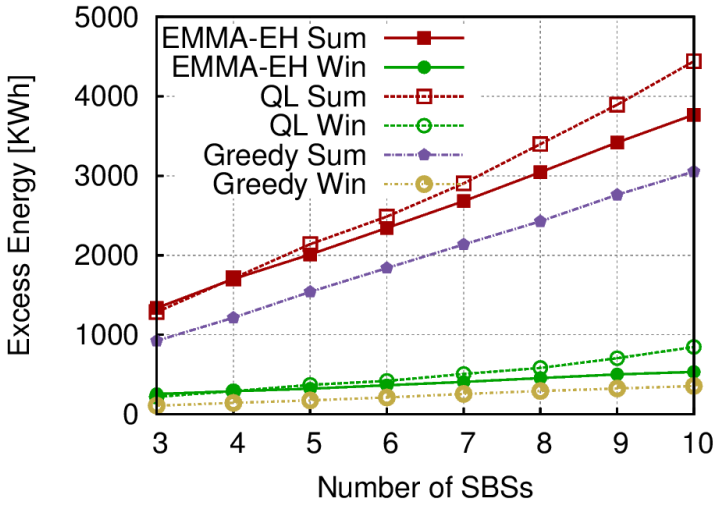


Figure 48: Average redundant energy for HAQL, QL and greedy

8 Virtual Infrastructure Manager and Planner (VIMaP)

The Virtual Infrastructure Manager and Planning application (VIMaP) is logically part of the 5G-Crosshaul XCI and stays at the lowest level of the application hierarchy. The VIMaP enables other applications (such as the Multi-Tenant Application, MTA) to

request the constrained allocation of physical and virtual Crosshaul resources (i.e., computing and networking resources) represented as an abstracted construct and proceeds to instantiate, deploy and provision them over the Crosshaul infrastructure.

The VIMaP itself, consequently, exports an API to applications, and this API constitutes its NBI. The goal of this application is to plan and optimize the physical and virtual Crosshaul resources (i.e., computing and networking resources) and to instantiate over the Crosshaul infrastructure the decisions taken. VIMaP consists of two components: the planner and the Virtual Infrastructure Manager (VIM). The planner component is in charge of running the required resource allocation algorithms for the planning and (re-) optimization of Crosshaul resources. The VIM is the component responsible for the dynamic provisioning and instantiation of Crosshaul resources. The planner runs the resource allocation algorithms with the aim of (re-)optimizing the Crosshaul resources. It is able to perform the constrained allocation of interconnected endpoints (including VMs, VNFs, etc) and network connectivity services.

The VIM is responsible for handling jointly the different IT and Network resources. Virtualization of IT resources is provided by means of a Cloud Controller, which is able to control the Crosshaul Processing Units (XPU), where the computing and storage resources are allocated. A VM might be requested based on its availability zone, its hardware resources (i.e. flavor), or the disk image/container/… to be loaded.

8.1.1 Consolidated design of the VIMaP application

The high level design of the VIMaP application is shown in the Figure 49 below, aligned (and augmenting) the notion of VIM within the ETSI NFV architecture. The Planning component of the application is in charge of running the required resource allocation algorithms for the planning and (re-)optimization needed to perform the placement of virtual machines within, and may also include the optimization of the network paths that provide connectivity. The VIM-P interface is between VIM and Planning component and, for deployments without “P” component, it is internal.

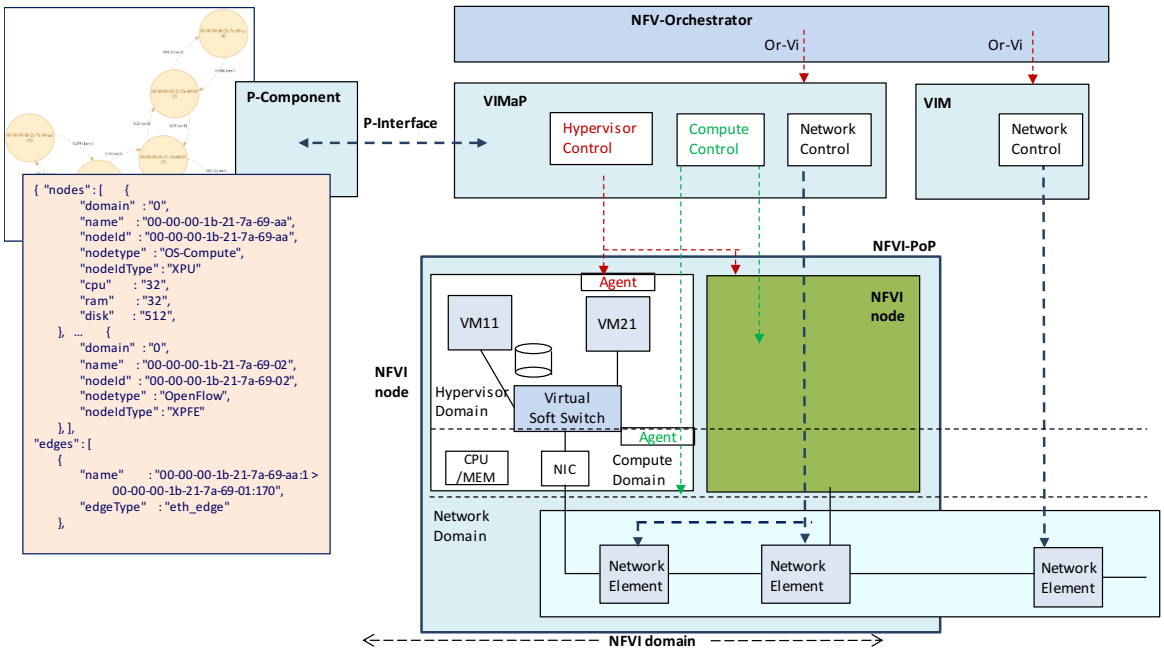


Figure 49: High level design of the VIMaP application scoped to the ETSI NFV framework.

8.1.2 Consolidated design of VIMaP algorithms

Our main driver is the design and implementation of the application, providing an architectural, interface and protocol framework, let us note that different resource optimization policies may be applied. Provisioning VNF graphs in an 5G-Crosshaul transport network is challenging when the constraints on compute and bandwidth resources both need to be addressed [40] [41]. There is an increasing number of recent works related to the algorithmic aspect of function placement, and we briefly mention some of them (and see references within each one). In particular, authors have considered the problem of VNF placement in [42] [43], while the studies in [44] [45] [46] [47] have investigated how to deploy VNF graphs in the form of chain or tree.

The problem of provisioning VNF graphs with arbitrary topologies in generic (e.g., inter Data Center networks) under compute/storage and bandwidth resource constraints is still open in multiple aspects, despite the increasing number of works addressing it. In [45], the authors assessed the problem of Virtual Machine Graph (VMGs) resource allocation in distributed Data Center (DC) scenarios by finding the minimum diameter graph (in terms of distance) to minimize the latency between VMs, proposing a resource allocation approach taking into account the distance between DC and the network load to select the connection path. In [48], an ILP model is formulated to minimize the total resource cost of VNF graphs provisioning, proving that it is an NP-hard problem. Authors leverage the minimum k-cut problem to design two time-efficient heuristics.

8.1.2.1 *Baseline algorithm*

The details of the baseline algorithm are provided in the appendix, Section 17.3.

8.1.2.2 *Net2Plan-based algorithms*

The modular design of the VIMaP application allows to integrate additional placement algorithms in the scope of the P-component. In particular, specific algorithms have been deployed using the facilities available in the net2plan Open Source software [49].

The latest available version of net2plan incorporates new modeling possibilities through the introduction of the concept of resource, allowing to model NFV scenarios. In this context, demand inputs in net2plan can represent service chain requests (i.e., a set of ordered connections among VNFs), and routes are service chains that can traverse links and resources, and can occupy a different amount of capacity in each link/resource traversed.

The net2plan software includes algorithms able to produce designs handling service chains like in NFV scenarios. Specifically, it includes a novel version of the *k-minimum* cost service chain algorithm, which computes the minimum cost paths that satisfy a service chain request.

The tool developed to perform the implementation of the P-component functionalities has been denominated as Comp2Plan. The schematic of the lab environment presents an internal tunnel between both partners to do the communications and complete the testing phase.

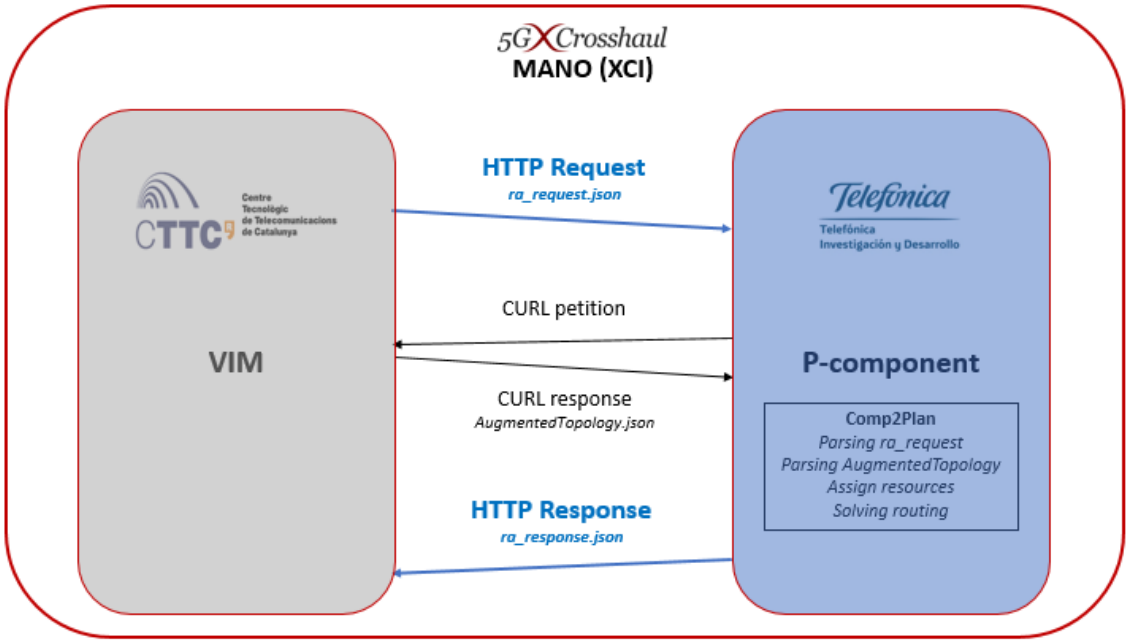


Figure 50: P-component functional schema

Figure 50 above shows a high-level representation of the functionality designed for the planner. The execution begins once the VIM need to solve a new VNF petition. This first step supposes a communication via HTTP Request that includes the VNFs needed and the computation requirements of the different nodes to assign (XPU).

Comp2Plan is a stateless algorithm, so after the reception of this VNF petition the P-component need to know the current state of the network/domain via cURL. Once the processing has been completed the results are returned to VIM domain (HTTP Response) and the operation is closed.

8.2 Implementation

The *VIMaP API* is implemented to provide a virtual infrastructure slice to a dedicated tenant or user. The API involves a request including a set of virtual instances interconnected forming a Virtual Machine Graph (VMG). The VIMaP architecture allows the *VIMaP logic* component to select the preferred algorithm depending on the desired resource allocation policy and delegate it to the P-component. The algorithm receives the resource allocation requests from the VIMaP logic and it obtains all the substrate infrastructure information from the *Resource Manager* component which maintains up-to-date information of both the cloud and the network underlying infrastructure.

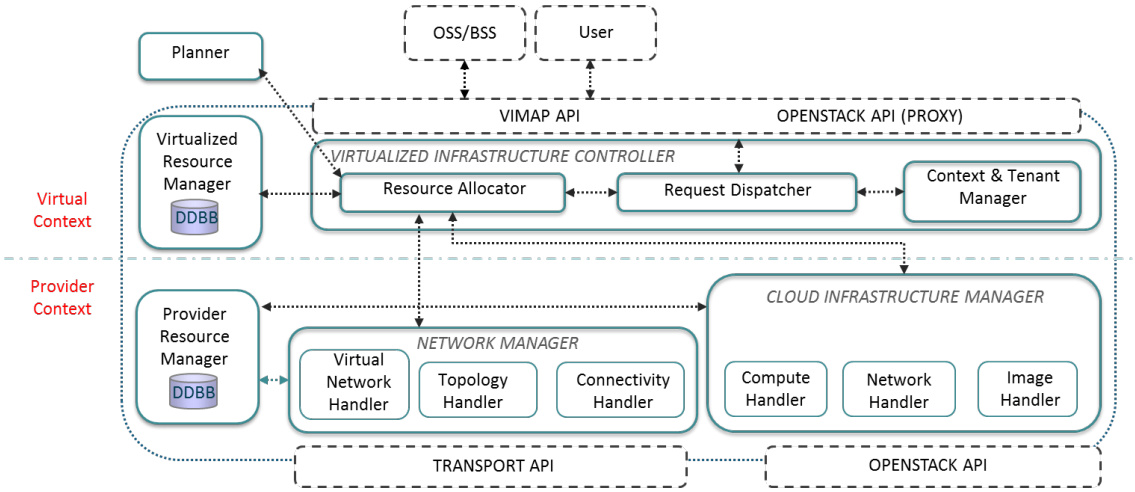


Figure 51: VIMaP internal block structure overview

The VIMaP includes a *dedicated configuration interface* for slice provisioning which is exposed to OSS/NMS management systems through a RESTful API. The *VIMaP logic component* is the responsible of orchestrating the workflows among the different architectural components in order to provision the cloud and network resources for an upcoming request. It is responsible for performing context-aware orchestration, exposing to each tenant only those resources allocated to the tenant itself by means of virtual representation. It includes a NBI which exposes the custom set of VIMaP programmable resources to each tenant.

The *Resource Manager* is responsible for storing and maintaining up-to-date state of all virtual and physical resources controlled by the VIMaP. It is also responsible for

maintaining the resource allocation relationship between the requested virtual resources and the allocated physical resources. The *Network Manager* functions are two-fold: first it provides the southbound interface towards network infrastructure controllers including the necessary APIs or protocols implementations. Secondly, the Network Manager is responsible for managing the virtual network resources of each tenant. The *Cloud Infrastructure Manager* is responsible for distributed cloud orchestration. Differently to the Network Manager, it is responsible of the partitioning and aggregation of cloud resources which might be distributed across different clouds (private, public). Once the selected DCs are allocated for a given tenant, it is responsible of creating a tenant session on each child cloud system and mapping all these client sessions to the corresponding VIMaP TenantID. Once this initial abstraction is performed, it is responsible for aggregating all the resources distributed among different clouds into a single unified view accessible by the tenant through the VIMaP NBI. This is performed populating the Resource Manager database with virtual representation of the resources deployed in the underlying infrastructure, these resources are segmented by its corresponding VIMaP global TenantID.

8.2.1 Decoupling of the planning function

As stated, it may be possible that the planning component (P-component) needs to be decoupled from the VIM component (VIM-component): the P-component may need dedicated computational resources, or may need to interact with other operators' business and support systems. The VIMaP application needs to define an internal interface between the VIM and the P components. This interface is designed to address the planning function and allow: i) to retrieve the topology of the network and the placement of the XPUs, and ii) to perform resource placement upon request from the P-c, allowing a P-c to request a (grouped or not) placement of a VM and its interconnection with one or more endpoints.

8.2.2 Planner description

Following the theoretical design discussed in previously, the planning tool consists on an application within the client/server architecture making in this case the P-component the server action. When a new connection arrives from VIM the process begins.

The logic of Comp2plan entails several steps that comprises the parsing of the information in JSON (ra_request.json and AugmentedTopology.json) to the Net2plan (n2p) object format. After that, the processing resources are assigned to the available nodes (XPU) doing now the placement. When each VNF are correctly located, the tool solves the routing and returns the path found using a Shortest Path protocol.

Figure 52 presents the flow chart of Comp2Plan algorithm used in the test phase.

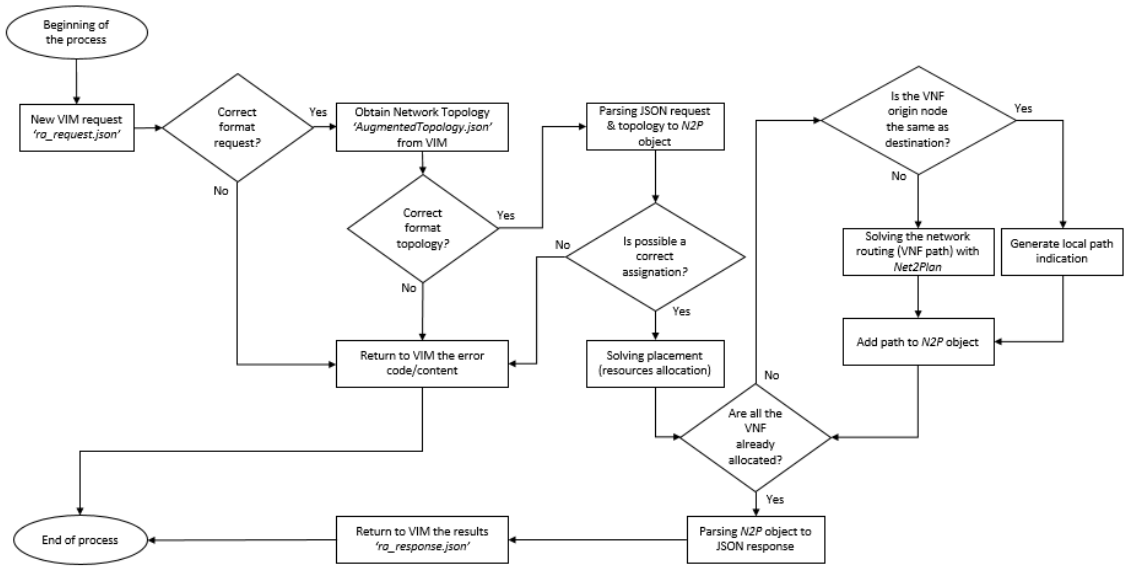


Figure 52: P-component flow chart

The execution of the process has been defined by the next steps:

1) Client – Server Communication (HTTP): First, the server is listening in an open port for new requests. When a petition arrives, the process begins.
2) Parsing of the Input Information: After the format check-up, the info is parsed to the Net2Plan object format (object <n2p>).
3) Solving Routing: This method comprises the placement of the resources requested and solves the needed paths with the functionalities of Net2Plan.
4) Parsing the Output Information: After the resolution of the network assignation, the data is parsed back from n2p to JSON.
5) Sending Response (HTTP): This function sends the solved info to the VIM within content of the HTTP response (JSON).

During the distinct interactions between the VIM and planning element, some inadequate situation may occur which must be notified in both parts. The available responses returned by Comp2Plan are included in Table 19.

Table 19: VIMaP Comp2Plan Return values

Code	Default Message	Description	*Comp2Plan* Context
200	OK	Standard response for successful HTTP requests. The actual response will depend on the request method used. In a GET request, the response will contain an entity corresponding to the requested resource. In a POST request, the response will contain an entity describing or containing the result of the action.	In Comp2Plan context, the reception of this message supposes a correct transaction and service resolution, all right.
400	Bad Request	The server cannot or will not process the request due to an apparent client error (e.g., malformed request syntax, size too large,	The reception of this code is related with a wrong syntax in the JSON supplied.

		invalid request message framing, or deceptive request routing)	
422	Unprocessable Entity	The request was well-formed but was unable to be followed due to semantic errors.	The reception of this code supposes that there is some problem with the content related (e.g., resource allocation petition impossible to resolve).
500	Internal Server Error	A generic error message, given when an unexpected condition was encountered and no more specific message is suitable	This message is sent when there is some problem in the internal execution of Comp2Plan.

8.3 KPIs

Table 20: list of KPIs addressed by the ViMAP

Application	***VIMaP***		
List of related project KPIs	***Obj.2: Specify the XCI's northbound (NBI) and southbound (SBI) interfaces*** *(5GPP KPI) Enable the introduction/provisioning of new 5G-Crosshaul services in the order of magnitude of hours*[3].		
Measurement of project KPIs as well as application specific ones	**KPI metrics (unit)**	**Description**	**Way of measurement**
	VIMaP Service Provisioning Latency (minutes)	The time it takes to deploy a VIMaP service, composed of a set of interconnected Virtual Machines	Experimental assessment.From Command Line Interfaces; Wireshark Capture / PCAP with NTP synchronized machines, Logs from the functional components and Statistical processing after a set of experiments

[3] KPIs related to the maturation of Network OS and NFV concepts, increased number of controllable resources, etc., still remain qualitative more than quantitative. In particular, baseline and benchmarking aspects refer to the increasing number of real deployments, reference implementations, contribution to standards and so on. That said, in the scope of VIMaP we are considering benchmarking in two main aspects: i) Algorithmic one and how advanced algorithms may improve efficiency in the use of resources, ii) Operational aspects related to the automation of processes.

	Max number of simultaneous services	Maximum number of simultaneous services that the transport infrastructure can support	Evaluation of blocking probability given a set of assumptions (e.g. Poisson requests).
	Control Plane Required Bw	Evaluation of the trade-off of message rate, required and available DCN bandwidth	Experimental assessment
	Control Plane Latency	Decompose to the VIMaP Service provisioning at different reference points	Experimental assessment
List of the State-of-the-Art approach **(used to compare with proposed solutions)**	**State of the art approach 1: Manual Provisioning of Network Services** Both benchmarking aspects are still not fully refined. On the one hand, algorithms still remain mostly in terms of theoretical studies, simulations and very specific assumptions, rendering practical algorithms complex. On the other hand, a baseline for improvements in automated network control and reduced network OPEX is considered and measured with regards to current human operator tasks. When available, a number of representative cases may be used to quantify human operated tasks and the effort it requires. Macroscopically, we consider: • VIMaP Services (NS) are provisioned in scoped and geographically constrained locations and currently most deployments are using private clouds in a single location. • No major adoptions of SDN/NFV integrated solutions with TRLs beyond 6. • Lack of integration of transport segments for distributed Data Centers. • Lack of adopted standard models for network topology, network services, The weakness of the method, the gap to the proposed solution: Data plane connections provisioned either manually, or via management planes (OSS/BSS). Limited deployment of GMPLS as transport network control plane. Even more limited of GMPLS for optical transport networks.No deployments of SDN control plane for fronthaul / backhaul / Crosshaul segments. **State of the art approach 2: Allocation of Virtual Machines in centralized data-centers** Use of mainly Layer 2 and Layer 3 technologies (Ethernet, IP), with limited deployments in geographically disperse locations. No consideration of transport / Crosshaul network segment. Need to account for multi-layer networks with coordinated orchestration of the packet (IP/Ethernet) and		

	Circuit (Optical) layers. For references, please see: [50] [51] [52] [53] [54] [55].

8.4 Validation and Evaluation

8.4.1 Scenarios Descriptions

The VIMaP application is being evaluated in two main scenarios. In both cases, the storage & computing controller is implemented in terms of a single OpenStack deployment, which controls several compute nodes (XPUs) distributed in different geographical locations within a single 5G-Crosshaul domain (NFVI administrative domain). The relevant interfaces to characterize in the scenarios are i) the VIMaP NBI , ii) VIM interface towards the OpenStack, iii) COP protocol towards the ABNO controller and iv) the P-interface.

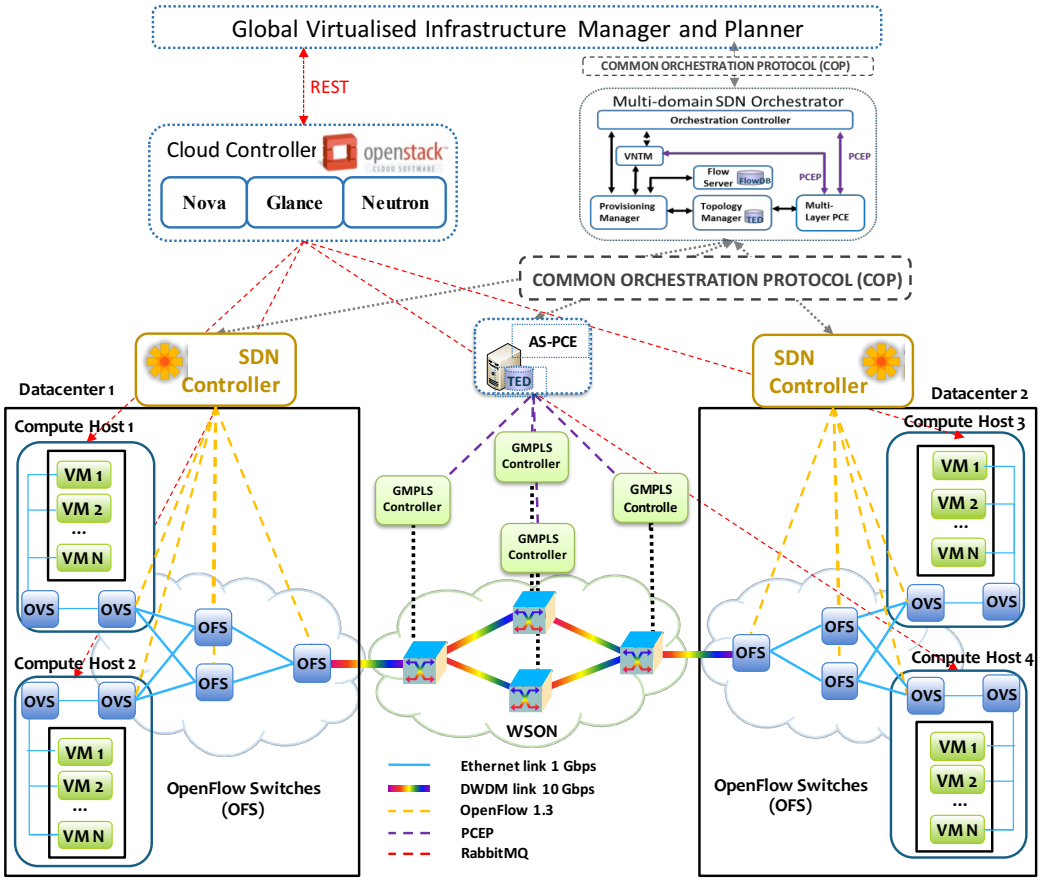


Figure 53: VIMaP testing scenario 1, including multi-layer and multidomain transport network

8.4.1.1 *Scenario 1: integration with the Transport Network SDN Controller*

Within Scenario 1 (Figure 53), the cloud computing platform is controlled using OpenStack (Havana release), which has been deployed into servers with 2 x Intel Xeon

E5-2420 and 32GB RAM each. An Openstack controller node and four compute nodes have been setup in different network locations. Each DC network is composed of four OpenFlow switches deployed on COTS hardware and using OpenVSwitch (OVS) technology. Two hybrid packet/optical aggregation switches based on OVS as well and with a 10 Gb/s XFP tunable transponder connecting to the DWDM network as alien wavelengths. Finally, the GMPLS/PCE-controlled optical network is composed of an all-optical WSON with 2 ROADMs and 2 OXCs. The COP has been employed as a Transport API for the orchestration of: two SDN OpenDaylight Helium controllers responsible of controlling the Ethernet intra-DC domains via OpenFlow 1.3; and the optical transport network via an ASPCE.

8.4.1.2 *Scenario 2, Integration with the P-Component*

The second scenario (Figure 54) has been deployed to ease the integration of P-components developed by third parties, and to perform performance evaluation.

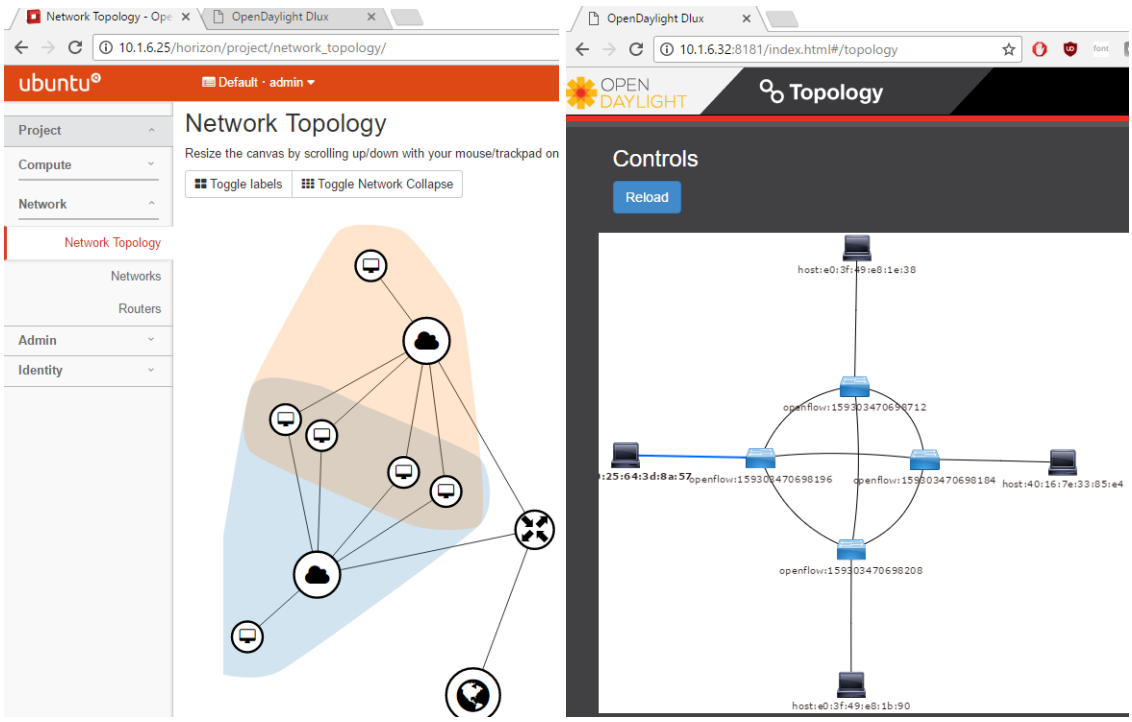


Figure 54: View form OpenStack and OpenDaylight of the infrastructure used to validate the decoupling of the P-Component

8.4.2 Test Cases

The main objectives of the planned tests to be carried out are the following: i) functional validation and experimental assessment of the VIMaP NBI interface; ii) functional validation of the interaction with the cloud controller (OpenStack); iii) functional validation of the interaction with the SDN controller and iv) functional validation of the P-interface.

Table 21: test case for functional validation of VIMaP

Test Card 1	CTTC-VIMaP-T001	Execution Status	Passed
Test Name	Topology detection		
Objectives	Retrieve the transport network topology from the SDN controller or orchestrator, following COP protocol, topology management service.		
Test Card 2	CTTC-VIMaP-T002	Execution Status	Passed
Test Name	XPU status and capability discovery		
Objectives	Retrieve the resources controlled by the OpenStack controller.		
Test Card 3	CTTC-VIMaP-T003	Execution Status	Passed
Test Name	Instantiation of Virtual Machines in remote locations		
Objectives	Allocate a set of Virtual Machines as provided in a request.		
Test Card 4	CTTC-VIMaP-T004	Execution Status	Passed
Test Name	Flow configuration across Single- and Multi-domain networks		
Objectives	Setup flows using the COP call and connection control services.		
Test Card 5	CTTC-VIMaP-T005	Execution Status	Passed
Test Name	Functional assessment of the VIMaP NBI		
Objectives	Verify API correct use by consuming applications.		
Test Card 6	CTTC-VIMaP-T006	Execution Status	Passed
Test Name	Enable external placement computation of VMs and flows		
Objectives	Delegate the placement computation of the VMs.		

8.4.3 Scenario 1: VMG algorithm allocation results (Simulation)

The proposed heuristic baseline solution is compared with a Random Fit based algorithm. The random solution differs on the DC selection strategy but keeps CSPF to assure path feasibility in the virtual link selection stage. The substrate infrastructure scenario employed for the experiments is an extended version of the NSFNET of 14 nodes and 42 unidirectional links and 6 DCs. For simplicity, the DCs are co-located

within the same network node locations and the connectivity between DC's and its corresponding network nodes is modelled to have infinite bandwidth. The substrate infrastructure is initially configured with pre-defined capacities which are maintained along all the experiments. The values of the capacities of each DC are uniformly distributed among the values included in each range. In the VMG requests, the number of virtual nodes is randomly determined by a uniform distribution between 2 and 20. Each pair of nodes are randomly connected with probability 0.5. The capacities of the virtual hosts and the virtual links are also selected randomly following a uniform distribution. The VMG requests arrives to the VIMaP following a Poisson process on which the arrival rate is varying. The holding time of the VMG requests in the system follows an exponential distribution with 10 time windows on average. We run all the simulations for 10000 requests for each instance of the simulation. The results of the simulation for different loads can be seen in Figure 55. The results show a slightly better performance of the greedy approach compared with the Random. This result is explained by the fact that the Greedy approach minimizes the number of DC selected in the first stage, minimizing as well the number of connections between DCs and thus decreasing the network utilization.

Parameter	Values
H^S CPU values	[100, 200, 400]
H^S Memory values	[200, 400, 800]
H^S Storage values	[10000, 20000, 40000]
L^S Bandwidth	100 Gbps
H^V CPU values	[1, 2, 4, 8]
H^V Memory values	[2, 4, 8, 16]
H^V Storage values	[20, 40, 80, 160]
L^V Bandwidth	(0.1:1) Gbps

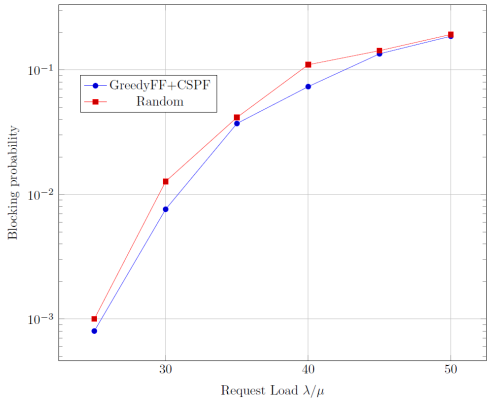


Figure 55: Blocking probability for the VM graph allocation request

8.4.4 Scenario 1, provisioning times with COP and Hierarchical SDN

In the experimental validation, we have introduced COP agents on top of SDN controllers in order to translate the received COP commands to SDN controllers NBI. In the network scenario where two geographically distributed DCs are interconnected through the WSON. We consider the on-demand deployment of two VMs in the cloud (one on each DC location) and the E2E connectivity provisioning across the proposed scenario. The network orchestration is performed using COP: a bidirectional CALL_SERVICE is requested to provide E2E connectivity to the previously deployed VMs, firstly requesting the creation of a virtual link in the upper layer topology (L2) which is translated internally by the VNTM module into two unidirectional L0 CALL_SERVICEs sent to the AS-PCE through the Provisioning Manager. They trigger, in the ASPCE, the creation of the corresponding GMPLS connections Label Switched Paths (LSPs). Afterwards the provisioning of the E2E service in the upper

layer is requested to the SDN controllers, by two new unidirectional CALL_SERVICESs to each domain. We can observe the request for Virtual Machine (VM) creation from VIMAP towards the Cloud Controller (which is running on the same server). The creation time for a single VM is of 15 seconds, which include the necessary time to boot up the VM. Secondly, we can observe the Call requests and the multi-domain call service set-up delay is of O(30) including all the steps (see Figure 56).

Time	Source	Destination	Info
REF	VIMAP-ABNO	VIMAP-ABNO	POST /create_vm HTTP/1.1 (application/json)
16.178267	VIMAP-ABNO	VIMAP-ABNO	HTTP/1.1 200 OK (text/html)
16.181848	VIMAP-ABNO	VIMAP-ABNO	POST /create_vm HTTP/1.1 (application/json)
30.914099	VIMAP-ABNO	VIMAP-ABNO	HTTP/1.1 200 OK (text/html)
REF	VIMAP-ABNO	VIMAP-ABNO	POST /restconf/config/calls/call/1 HTTP/1.1 (ap
0.123129	VIMAP-ABNO	SDN-CTL-1	POST /restconf/config/calls/call/00001 HTTP/1.1
0.287926	SDN-CTL-1	VIMAP-ABNO	HTTP/1.1 200 OK (application/json)
0.329396	VIMAP-ABNO	AS-PCE	POST /restconf/config/calls/call/00002 HTTP/1.1
1.439163	AS-PCE	VIMAP-ABNO	HTTP/1.1 200 OK (application/json)
1.493375	VIMAP-ABNO	SDN-CTL-2	POST /restconf/config/calls/call/00003 HTTP/1.1
1.527963	SDN-CTL-2	VIMAP-ABNO	HTTP/1.1 200 OK (application/json)
1.628074	VIMAP-ABNO	SDN-CTL-2	POST /restconf/config/calls/call/00004 HTTP/1.1
1.660056	SDN-CTL-2	VIMAP-ABNO	HTTP/1.1 200 OK (application/json)
1.699451	VIMAP-ABNO	AS-PCE	POST /restconf/config/calls/call/00005 HTTP/1.1
2.308023	AS-PCE	VIMAP-ABNO	HTTP/1.1 200 OK (application/json)
2.358390	VIMAP-ABNO	SDN-CTL-1	POST /restconf/config/calls/call/00006 HTTP/1.1
2.502622	SDN-CTL-1	VIMAP-ABNO	HTTP/1.1 200 OK (application/json)
2.519650	VIMAP-ABNO	VIMAP-ABNO	HTTP/1.1 200 OK (application/json)

Figure 56: Wireshark capture of the exchanges between VIMaP and ABNO

8.4.5 Scenario 2: Remote P-Component

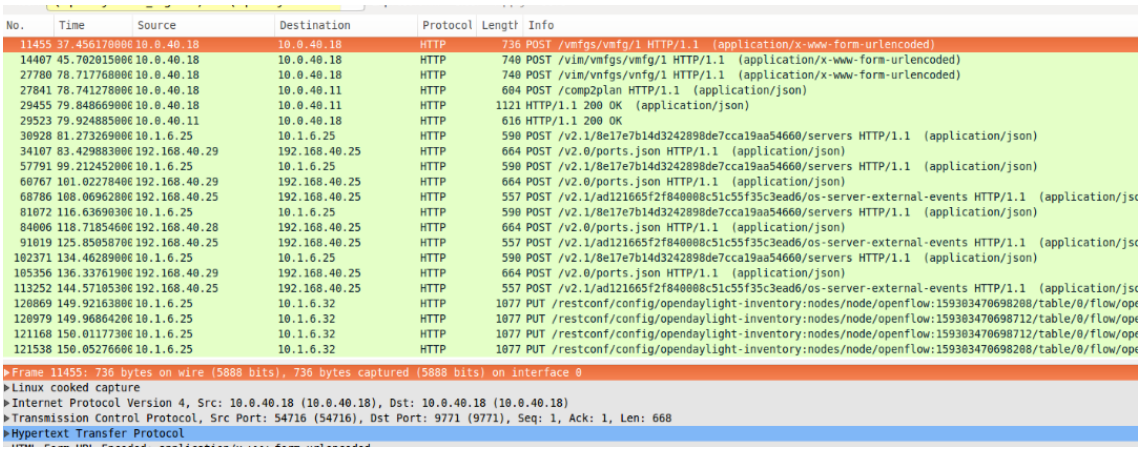


Figure 57: Wireshark capture of the exchanges between VIMaP application and the Cloud controller(10.1.6.25) and the SDN Controler (10.1.6.32)

The second test aims at validating the P-Interface. For this, a second scenario has been set up and the interface between the P-component and the VIM-component has been implemented. For the former, the VIM and P components are connected using an OpenVPN connection, and the interface is consumed as follows (see Figure 57):

- A request can be done to the VIMaP application, (NBI) including the characterization of VM and their interconnection. Each VM has a flavour, as per the following table:

Flavor	CPU	RAM (GB)	*Disk (GB)*
Tiny	1	0.5	1
Small	1	2	20
Medium	2	4	40
Large	4	8	80
Xlarge	8	16	160

- A request to the Planner component is carried out by the VIMaP core using a REST call, which skeleton is show below:

```
{

   "id" : "1",

   "vms" : [{ "name" : "vm1", "flavor" : "tiny" },…  ],

   "net_allocation":[{"edgeId"    :    "link1",    "source"    :
"vm1","target" : "vm3"},…],

}
```

- The P-Component can retrieve the topology from the VIM using a REST interface, using the URL http://vimhost:9771/vim/getAugmentedTopology

- The P-Component replies with the computed placement and allocation
- The VIM core component provisions the VM and paths.

The overall service takes O(100)s [150 in the presented experiment], and the actual placement computation 1.22 seconds (including VPN latency). From the point of view of TCP / HTTP, control plane interface characterization, a computation takes an HTTP exchange, with 604 and 616 bytes, respectively, in less than 2 seconds.

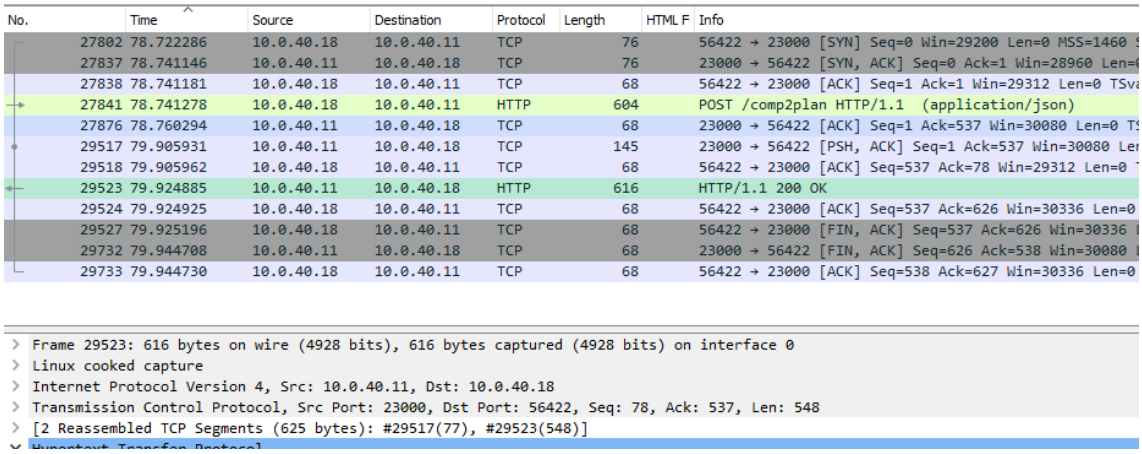


Figure 58: Wireshark Capture highlighting the main HTTP exchange between VIM.-component and P-component (detailed TCP exchange)

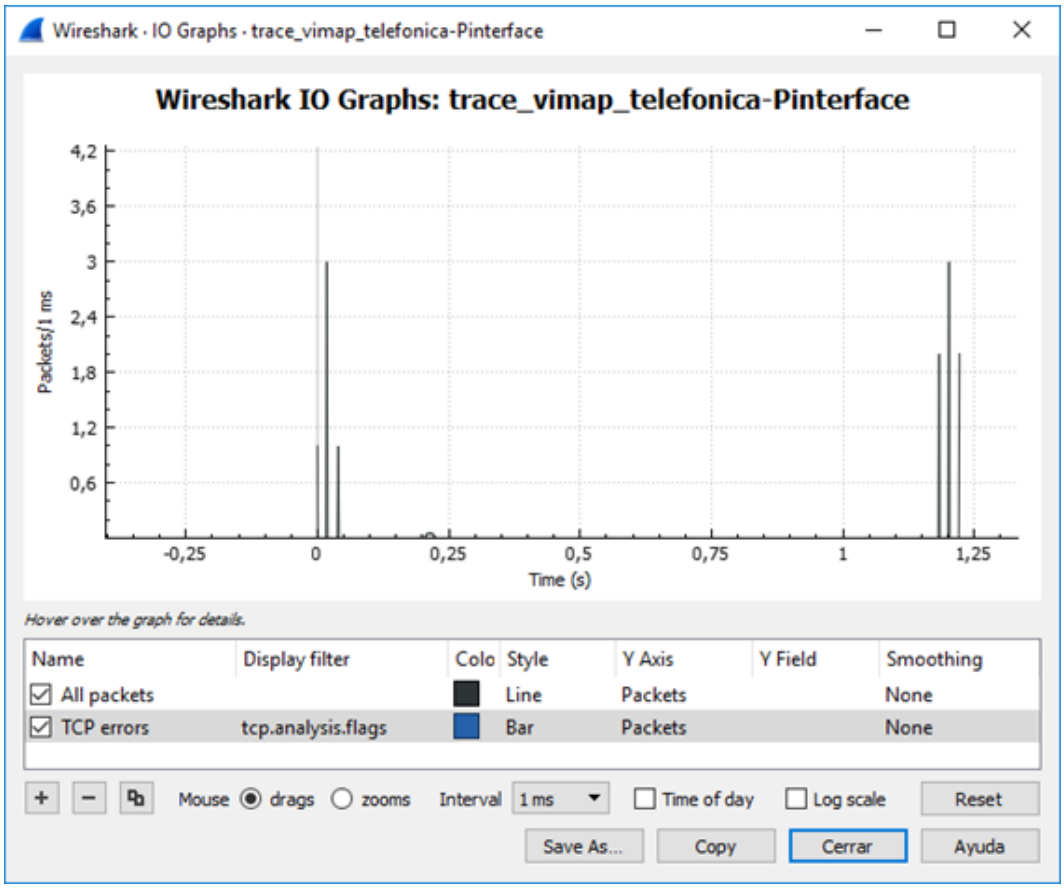


Figure 59: I/O plot of the HTTP based placement request exchange

8.4.6 Scenario 2: Algorithm scalability analysis

From the point of view of the P-component, the scalability figures that we consider are:

- Max number of supported nodes: 1000
- Max number of demands considered among those nodes for path calculation: 10000
- Max number of links (/connections) among those nodes: 8000

8.5 Conclusion

Within 5G-Crosshaul, we have introduced the need of the VIMaP application from a many-fold perspective: first and foremost, to orchestrate network and cloud resources in a heterogeneous environment characterized by having compute nodes interconnected by transport networks that are designed to support fronthaul and backhaul traffic and, on the other hand, involve multiple technological layers justifying deployments of SDN controllers in hierarchical arrangements. We have defined and experimentally demonstrated the novel Virtual Infrastructure Manager and Planner (VIMaP) component for the resource allocation and planning, of Virtual Machine Graphs (VMGs). We have formally modeled the problem and included a baseline heuristic

solution to evaluate the proposed architecture. We have demonstrated the feasibility of the approach with an implementation and proof-of-concept experimental deployment.

From the point of view of 5G-Crosshaul Project KPI and objective *Obj.2. Specify the XCI NBI interface* (As part of the XCI, VIMaP NBI specification is related to Objective 2.) *nable the introduction/provisioning of new Crosshaul services in the order of magnitude of hours.*

- The considered service is the **automated provisioning of a graph of Virtual Machines**, and their interconnection, as received in a request from the operator by the VIMaP NBI. The service is deployed when the VMs are instantiated and flows are provisioned ensuring conenctivity. And the VIMaP orchestrates the service and resources from an underlying SDN and cloud controller with optional third party placement computation (P-component)
- The measured performance allows us to state that by means of SDN/NFV automation and the Crosshaul XCI, VIMaP services are provisioned in orders of magnitude of minutes, thus meeting this indicator.
- The mesurements have been obtained by means of experimental assessment processing the traces captured by Wireshark software taking as refrence key evens such as service provisioning trigger from the involved functional components

9 Mobility Management Application (MMA) for Traffic Offloading

The Mobility Management Application (MMA) is one of the Over-The-Top (OTT) applications of the 5G-Crosshaul system. The main goal of the MMA is the reduction of the traffic to/from the core network by selective offload of traffic as near as possible to the RAN reducing the cost of the core network and improving the experience of the user reducing the latency and delay in end-to-end communications. Therefore, the MMA is in charge of the user detection and mobility management in the 5G-Crosshaul domain to optimize the traffic offload for media distribution. Then, the MMA requires some information to provide this service, the tenant's constraints, detailing the SLAs, PoC and XPUs available and the CDN nodes placement for the traffic offloading.

The MMA procedures can be divided in two differentiated stages: the user detection and the mobility.

First, the MMA is in charge of detecting the presence of new users in the 5G-Crosshaul domain. This is possible through REST-based core and RAN interfaces. Once the MMA collects all the available and useful information of the new user it notifies the CDNMA and TVBA, after this notification the MMA receives the CDN node assigned to the user

from the CDNMA and makes use of the RMA API to provide the best paths between the different elements of the network.

Second, the MMA mobility solution is based on Distributed Mobility Management (DMM), avoiding the bottlenecks and scalability problems of the centralized MIPv6 and PMIPv6 legacy solutions. DMM improves the scalability by flattening the network, distributing mobility anchors by placing multiple ones closer to the user. In the following sections, we analyze two DMM approaches that can be used for the MMA implementation: a DMM PMIPv6-based solution and an SDN-based solution.

9.1 Consolidated design

Figure 60 depicts the high level design, MMA is divided in three separated modules: the user detection, the mobility procedures and the application notification. The user detection procedures are performed by the User Monitoring Manager, the mobility procedures by the Mobility Manager and the notifications to external applications by the Notification Manager.

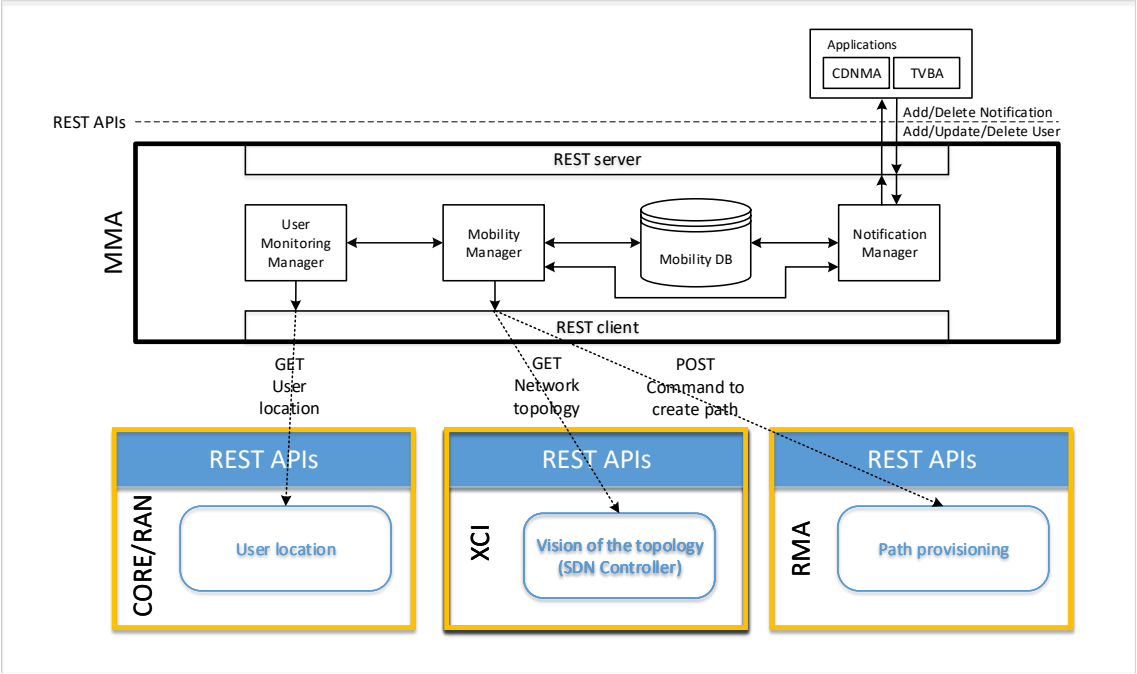


Figure 60: MMA high level design

The User Monitoring Manager relies on the core and RAN APIs based on REST that will return the user location.

The Notification Manager will be in charge of notifying to the subscribed applications the presence of new users and their mobility inside the 5G-Crosshaul domain. This notification procedures are based on a REST API.

Both prodecures can be described by two algorithms. We can differenciate two separeted situations, when a user enters the 5G-Crosshaul domain (Algorithm 2), and when a user moves and a handover occurs (Algorithm 3).

The Mobility Manager is based on DMM. We have analyzed two DMM approaches: a PMIPv6-based and an SDN-based as implementation choice.

- PMIPv6-based DMM solution: the key entity in this solution is the Distributed Mobility Management Gateway (DMM-GW). The DMM-GW extends the PMIPv6 Mobile Access Gateway (MAG) functions incorporating most of the functionality of the PMIPv6 Local Mobility Anchor (LMA). Hence, a DMM-GW provides connectivity to IP based services, e.g., Internet, and has the capability of assigning and anchoring IPv6 prefixes. A unique IPv6 prefix pool belongs to each DMM-GW, from which a prefix is assigned to every Mobile Node (MN) attached to the DMM-GW's access links. In this way, a DMM-GW acts as a plain access router to forward packets to and from the Internet. Moreover, it is provided with mobility anchoring functions, that is, a DMM-GW is able to maintain the uplink and downlink forwarding for the IP flows that an MN started while attached to that DMM-GW, even after the MN has moved to a new DMM-GW. An external node, referred to as Control Mobility Database (CMD), is used to store the location of the MNs in the domain (i.e., the bindings).
- SDN-based DMM solution: SDN is focused on decoupling the data plane and the control plane, so the design of the DMM-GW is based on plain forwading nodes, bearing no mobility functionality, and hence delegating the whole intelligence to the network controller. The application adopt an event-driven communication paradigm in order to be as modular and reactive as possible, being this aspect fundamental in mobility solutions. Indeed, an event-driven communication can be used to fire triggers upon certain events, like the mobility support being activated by an MN handover. Mobility support is achieved by installing OpenFlow rules at the Egress Routers and DMM-GWs. This SDN-based solution envisions a shared data plane and a compliant control plane, exploiting OpenFlow as signaling protocol.

Algorithm 1 New user connection

```
1: if packet-in received then
2:     if llc/snap header then
3:         Get user mac
4:         Get AP IP
5:         Get Network Topology
6:         Decide GW/PoC
7:         if CDNMA subscribed then
8:             Notify
9:             Get CDN Node
10:        Create paths
11:        Update DB
12:        return
13:    else if DHCP header then
14:        if User in DB then
15:            Select previous IP
16:        else
17:            Select new IP
18:        if DHCP discovery then
19:            Send DHCP Offer
20:        else if DHCP request then
21:            Send DHCP ACK
```

Algorithm 2: New user connection

Algorithm 2 Handover

```
1: if Handover then
2:     Determine the target BS
3:     Find the proactive path
4:     if CDNMA subscribed then
5:         Notify
6:         Get CDN Node
7:     Create paths
8:     Update DB
9:     return
```

Algorithm 3: Handover

9.2 Implementation

The main MMA internal component are shown in Figure 60 and summarized in Table 22.

Table 22: MMA application components

Component	Description
REST Client	Entity in charge of interacting with the XCI components via REST APIs.
REST Server	Entity in charge of interacting with external applications (CDNMA, TVBA) via REST APIs.

User Monitoring Manager	Entity in charge of detecting the presence of new users and their mobility inside the 5G-Crosshaul domain.
Mobility Manager	DMM implementation in charge of managing the mobility procedures inside the 5G-Crosshaul domain.
Notification Manager	Entity in charge of managing the application subscription and the notification procedures.
Mobility DB	Internal Mysql db to store mobility and notification information.

The MMA implementation leverages on a set of interconnected OpenFlow Ethernet switches with a WLAN interface working as PoA. The forwarding will be based on a direct modification of flow tables at the 5G-Crosshaul Packet Forwarding Element (XPFE) using standard OpenFlow rules. The 5G-Crosshaul mobility will be based on a flat IPv6 network, on which traffic is forwarded to the nearest Point of Connection (PoC) to the Internet. To make it possible the MMA will handle the association of the Gateways (GW) to the users in addition to the Neighbor Discovery mechanisms. The assignment of the PoC to the GW will be made based on heuristic and proximity.

Depending on how the RMA executes the allocation of paths the mobility can be handled in different manners:

- Based on VLANs from PoCs to XPUs through the network, handling the user-VLAN mapping.
- Based on IP hop by hop.
- Based on MAC hop by hop.

The GWs and the CDN nodes will be instantiated at the 5G-Crosshaul Processing Units (XPUs). The number of GWs and CDN nodes will depend on the contractual agreements between tenants, but typically the GW and the CDN node will be instantiated in each XPU. The notification to the CDNMA will be made through the MMA REST API.

In the case of a fixed user, like in Figure 61, the GW is necessary because the PoC switch may not be SDN enabled and the IP stack may not work.

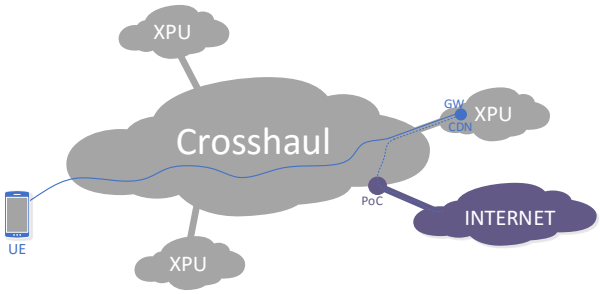


Figure 61: Fixed user

There may be the possibility of having the CDN Node in a different XPU of the PoC, like in Figure 62.

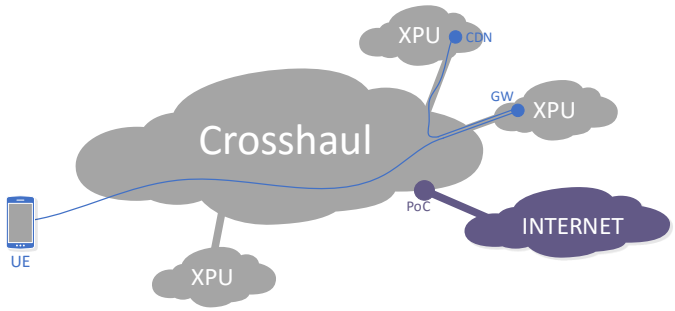


Figure 62: CDN node not located in the PoC XPU

The traffic of the user goes through the GW to the PoC/CDN node. When the user enters in a 5G-Crosshaul domain and attaches to a GW it is assigned a prefix; in a mobility scenario there will be several prefixes (one for each GW). The MMA can handle the old prefix in two different ways:

1) All the traffic addressed to the previous prefix/GW will be tunneled to the new prefix/GW (Figure 63).

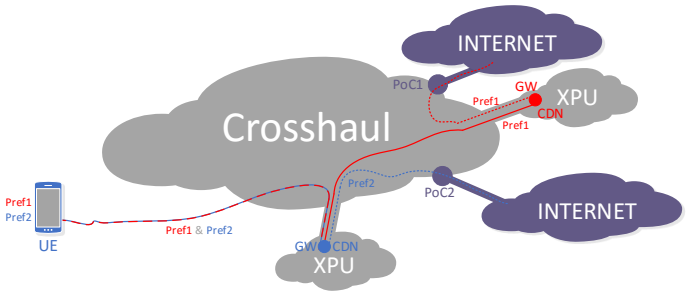


Figure 63: Mobile user scenario with GW tunnel

2) The path to the previous GW is maintained, so the new and the previous GW work independently with their respective paths. (Figure 64)

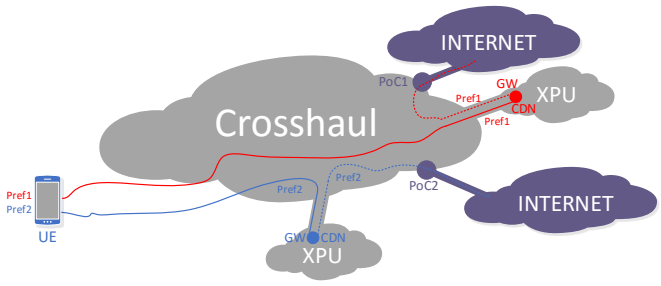


Figure 64: Mobility scenario with previous path maintenance

In both mobility scenarios, if a change of CDN node is necessary, the MMA will notify the CDNMA through the MMA REST API to instantiate a new CDN node nearer to the new GW.

The above functionality can be easily implemented on the Mobility Manager through an SDN application. We will hence focus on the functionality required by other applications, the detection of users at the RAN level and the notification prodecures: the User Monitoring Manager and the Notification Manager.

The User Monitoring Manager have been implemented in Ryu, which provides software components, a well-defined API and clear documentation for developing SDN applications. The user detection is implemented in the packet-in handler of the Ryu application, when a MN connects to the 5G-Crosshaul PoA (a WLAN interface) it generates a LLC packet, which generates a packet-in to the controller. Ryu Packet Library (ryu.lib.packet) supports the LLC header allowing a faster and easy implementation of the parsing routine of the packet.

The ofp_event.EventOFPPacketIn event allows our packet-in handler routine to identify this LLC packet and obtain the MAC address of the new MN connected to the 5G-Crosshaul domain. The MN will configure an IP address using DHCP and the PoA will notify the controller the IP address exchanged in the DHCP messages.

Now the MMA knows the L2 and L3 addresses of the MN, it stores this information in the Mobility Database and proceeds to the notification routine. MMA notifies to the multimedia applications (CDNMA and TVBA) the presence of a new MN in the domain with a REST API.

The Notification Manager have been implemented using the Web server function corresponding to WSGI implemented in Ryu. The first step is the subscription procedure, the receiver of the notification sends an HTTP POST message with the URL where the REST API of the listening receiver is encoded in JSON format. The second step is the notification procedure, where the MMA sends the L2 and L3 addresses of the PoA and the MN to the different subscribers in an HTTP POST message.

In the particular case of the CDNMA notification, the CDNMA responds with the CDN Node assigned to the MN, the MMA stores this information in the Mobility Database and uses the API offered by the RMA to allocate the optimal path from the user to the CDN.

9.3 KPIs

Table 23: list of KPIs addressed by MMA

Application	***MMA for Traffic Offloading***		
List of related project KPIs	***Obj.6. Design scalable algorithms for efficient 5G-Crosshaul resource orchestration*** *(5GPP KPI) Support 10 times increased node densities* *(5GPP KPI) Increase of total 5G-Crosshaul network throughput by > 20%*		
Measurement of project KPIs as well as application specific ones	**KPI metrics (unit)**	**Description**	**Way of measurement**
	Throughput increase (%)	Percentage of increase of the total 5G-Crosshaul network throughput	Measure the overhead in DMM compared to legacy PMIPv6-based solutions.
	UE density (Terminal/Km^2)	Number of UEs supported per geographical area unit	Measure the packet delivery cost in DMM compared to legacy PMIPv6-based solutions
List of the State-of-the-Art approach	State of the art approach 1: *PMIPv6* Proxy Mobile IPv6 is a network-based mobility management protocol. Network-based mobility management enables IP mobility for a host without its participation on the mobility signaling. The mobility entities of the network are responsible of tracking the movement and initiate the mobility signaling. Figure 65 depicts a typical PMIPv6 scenario, there are some well-known issues with PMIPv6, starting with the scalability, a very high density of mobile nodes in PMIPv6 is susceptible of different bottlenecks in the Local Mobility Anchor (LMA) due to the centralized architecture, In PMIPv6 all the traffic destined to a Mobile Node (MN) will be sent to the topological anchor point of the mobile node's home network (LMA), it manages the binding state of the mobile nodes and sends all the traffic through a tunnel to the Mobile Access Gateway (MAG) where the MN is anchored. The traffic originated in the MN follows the same principles going from the MAG to the LMA and then to the destination. This architecture proposed in PMIPv6 has well known scalability issues due to the centralization, all the traffic goes through the LMA which is susceptible of bottlenecks and introduces an overhead due to the IP-		

in-IP tunnel from the MAG to the LMA.

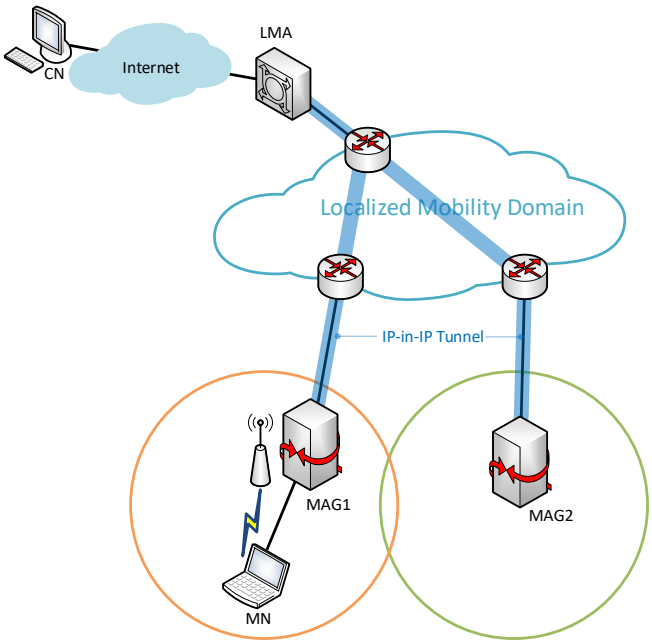


Figure 65: PMIPv6 scenario

9.4 Validation and Evaluation

This section provides the MMA validation and evaluation results. We have analyzed the results of the User Monitoring Manager, the Notification Manager and the Mobility Manager implementations.

9.4.1 Evaluation Environment and Scenario

We have used different scenarios for the User Monitoring Manager and for the Mobility Manager evaluation and validation.

First, the scenario used for the user detection tests is showed in Figure 66. The objective of this scenario is the validation of the user detection and the notification procedure.

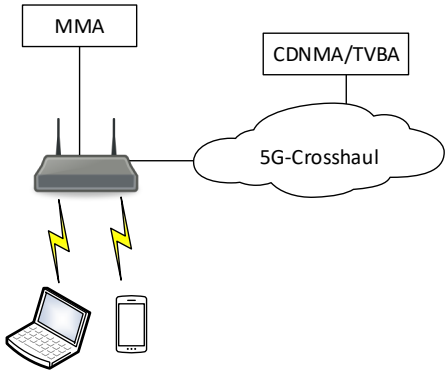


Figure 66: User detection scenario

Then, we have measured different metrics using the PMIPv6-based DMM approach

(Figure 67) and the SDN-based DMM approach (Figure 68).

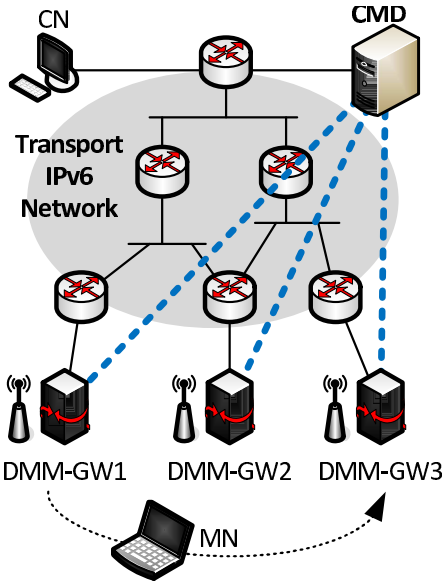


Figure 67: PMIPv6-based scenario

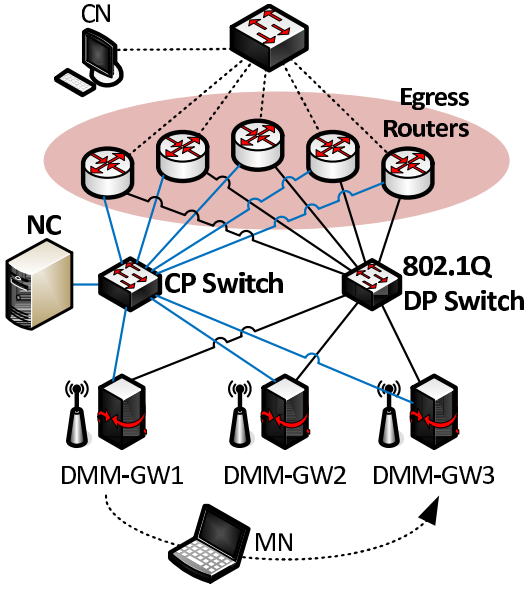


Figure 68: SDN-based scenario

9.4.2 Test Cases for functional validation

The main objective of these test cases is to use the scenarios showed in the previous subsection to validate the correct operation of the MMA. We can separate the tests in two groups. First the test related to User Monitoring Manager and the Notification Manager, and second the tests related to the Mobility Manager. Tests from 1 to 3 validates the user detection and tests 4 and 5 evaluates the DMM implementation. The following tables describe the different test cases:

Table 24: MMA Offloading case test cards

Test Card #	UC3M_MMA_001	Execution Status	Passed
Test Name	User detection		
Objectives	The MMA is able to detect the new users in the 5G-Crosshaul domain and get the necessary information.		
Test Card #	UC3M_MMA_002	Execution Status	Passed
Test Name	IP address configuration		
Objectives	The MMA is able to assign an IP address to the new users with DHCP		
Test Card #	UC3M_MMA_003	Execution Status	Passed
Test Name	REST API		

Objectives	The MMA is able to send and receive REST notifications

Table 25: MMA DMM test cards

Test Card #	UC3M_MMA_004	Execution Status	Passed
Test Name	Handover signaling		
Objectives	Measure the handover signaling cost in the PMIPv6-based and in the SDN-based DMM scenarios.		
Test Card #	UC3M_MMA_005	Execution Status	Passed
Test Name	Flow recovery		
Objectives	Measure the flow recovery time in the PMIPv6-based and in the SDN-based DMM scenarios.		

9.4.3 Evaluation Results

The evaluation of the User Monitoring Manager and the Notification Manager does not provide any statistical or empirical result, with the test performed we only validate the correct operation of the each application module.

Figure 69 shows the command line output of the User Monitoring Manager. The MMA obtains the IP/MAC of the AP and the user, and then notifies through the Notification Manager the CDNMA and TVBA, the CDN IP is selected by the CDNMA and sent in the notification reply.

```
DEVICES
| AP IP:      10.5.1.16  | AP MAC: 04:f0:21:06:1d:17  | User IP:     192.168.1.5  | User MAC: 00:25:d3:44:8c:5b  | CDN IP:      10.5.1.230  |
```

Figure 69: MMA User Monitoring Manager validation

We have evaluated the Notification Manager module with the CDNMA and TVBA. The notification prodecure can be decomposed in three iterations:

1. The subscription: Figure 70 shows the wireshark capture with the HTTP messages exchange focusing on the subscription. The JSON object in the body includes the url provided by the CDNMA where the MMA will notify with the presence of new users.
2. The notification: Figure 71 shows the HTTP post message sent from the MMA to the CDNMA with all the information of the user and AP.

3. The CDN node assignment: Figure 72 shows the HTTP 200 OK sent by the CDNMA with the CDN Node assigned in the JSON object.

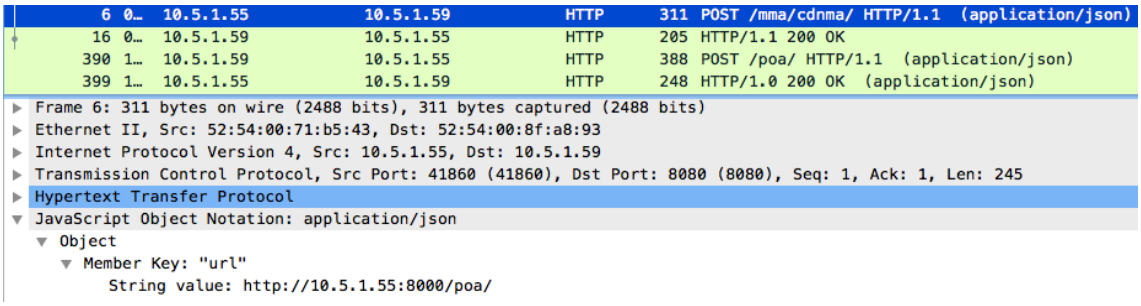


Figure 70: Notification Manager validation: subscription

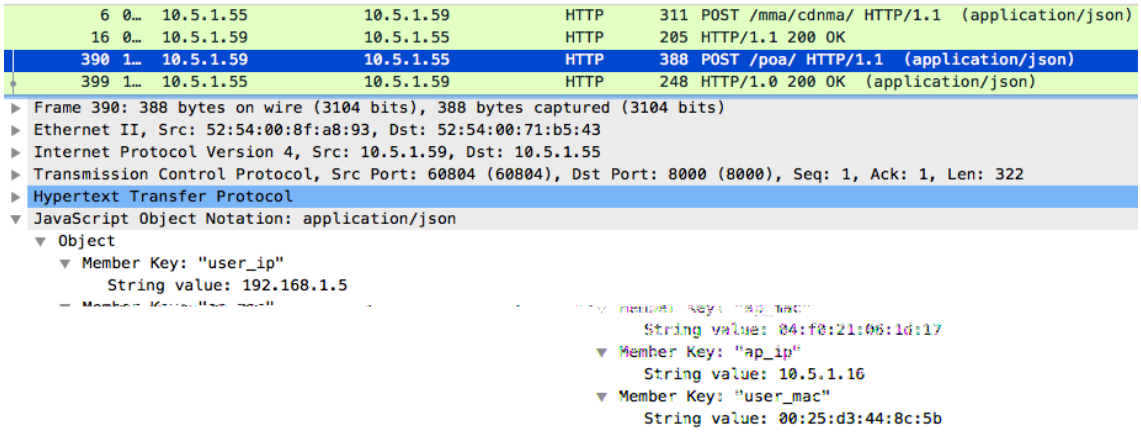


Figure 71: Notification Manager validation: notification

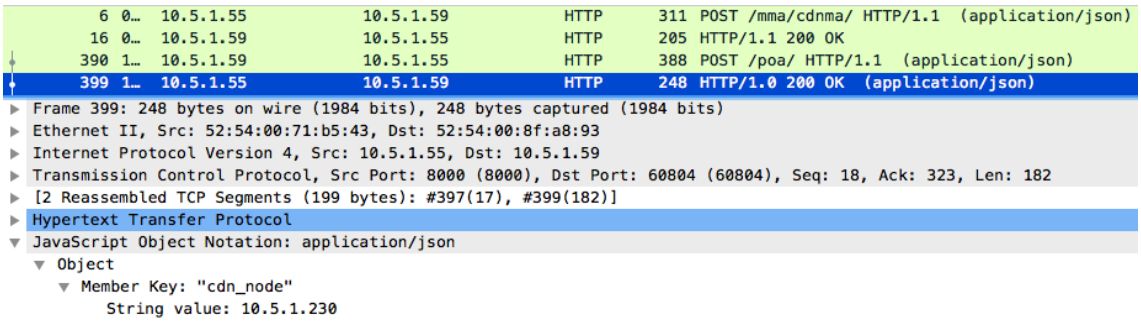


Figure 72: Notification Manager validation: CDN node assignment

In case of the Mobility Manager evaluation, we have obtained different results [56] that are shown in the following lines.

First, we have measured the handover signaling cost for both DMM approaches: Figure 73 and Figure 74 show the handover signaling cost of the SDN-based DMM and the PMIPv6 solutions respectively, *k* represents the number of Egress Routers, λ represents the mean of the exponential interval that an old DMM-GW remains active and μ represents the mean of the exponential time that a MN expends attached to a DMM-GW

before a handover. We observe a performance degradation on the PMIPv6-based solution for large values of the ratio λ/μ, a scenario with high mobility and long-lived IP flows. In order to cope with this limitation, the deployment of such solution should jointly consider the coverage area and the level of mobility of MNs. The DMM-GW's coverage area should not be too small in order to reduce the number of active DMM-GWs. Nevertheless, this solution is more suitable when handling scenarios with low mobility, i.e., for high values of μ, or short lived flows, i.e., for low λ.

The SDN-based solution behaves in a more predictable way, as it only depends on the value k and it is independent of the traffic pattern of MNs. Operators have the profiles of each MN, therefore the value k can be also adapted on an MN basis. As a result, the network can be managed in a smarter way and a higher network's efficiency can be achieved by spreading the MNs on multiple Egress Routers.

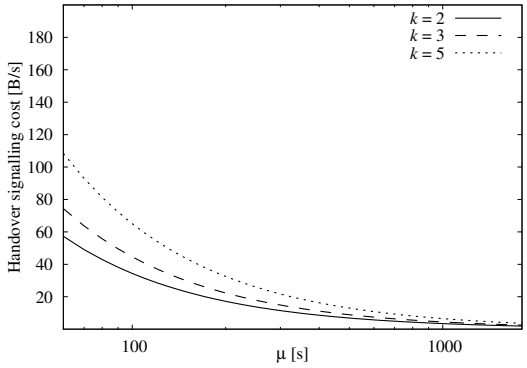


Figure 73: Handover signalling cost in the SDN-based DMM solution

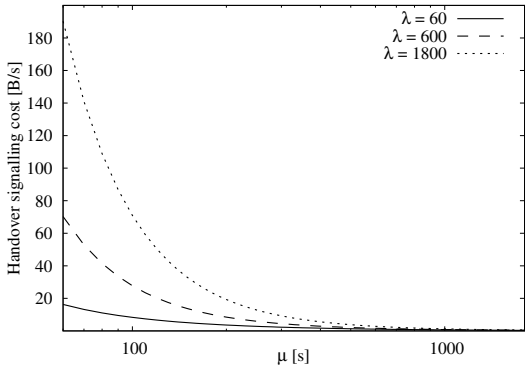


Figure 74: Handover signalling cost in the PMIPv6-based DMM solution

Secondly, we have also measured the flow recovery time in both scenarios.

Figure 75 and Figure 76 show the results for the PMIPv6-based and SDN-based approach respectively. The Layer-2 handover does not depend on the mobility protocol. The table reports the same value for all setups because the measured difference was negligible in our testing system. Regarding the Layer-3 configuration, it can be observed that the two protocols behave similarly on average, showing a lower variance in the SDN case due to some system-level optimizations applied to the node acting as network controller. The IP flow recovery time exhibits the largest gain in favour of the SDN approach. This is due to the use of tunnelling by the PMIPv6- based solution, which was observed to introduce, on average, around 15ms of additional delay with respect to the Layer-3 configuration.

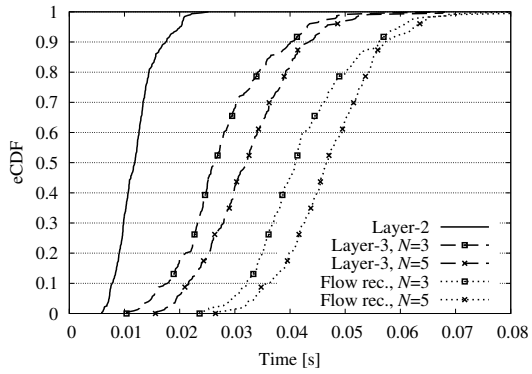


Figure 75: Flow recovery time in the PMIPv6-based DMM solution

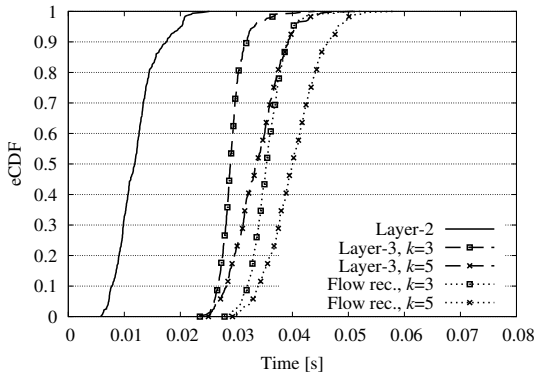


Figure 76: Flow recovery time in the SDN-based DMM solution

Nowadays data sessions tend to be short, most of them are initiated and terminated in the same region as the case of instant messages or CDNs which brings content closer to the users. In a scenario where $\lambda<\mu$ the overhead reduction of a DMM solution compared to a legacy PMIPv6 solution is clear. Figure 77 shows the percentage of overhead introduced by legacy PMIPv6 depending on the packet size in the scenario where $\lambda<\mu$. The dashed line in the figure shows the KPI threshold of 20%.

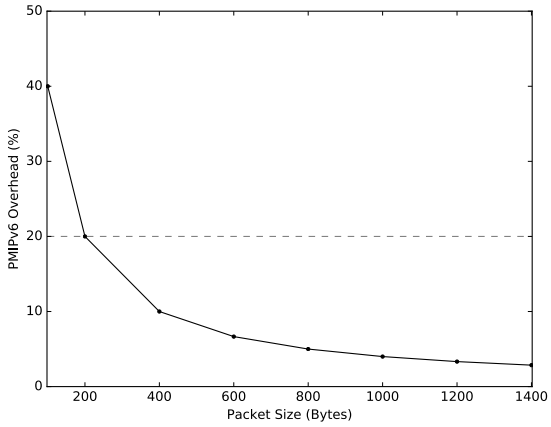


Figure 77: PMIPv6 overhead depending on packet size with $\lambda<\mu$

A DMM solution allows the delay reduction between endpoints bypassing the core network decreasing the CAPEX and OPEX reducing the use of powerful, expensive network nodes (like P-GW) and links. As shown in Figure 77, in a short flow scenario DMM allows a throughput increment, reducing the overhead and the signaling between the MAG and the LMA and avoiding the IP-in-IP tunneling overhead; the throughput increase varies depending on the packet size, while with low packet sizes the throughput increase is higher. The DMM solution also increase the scalability of the network avoiding the bottleneck of the LMA distributing the gateways across the network.

We have based our scalability analysis on [57]. Considering that the transfer of packets at rate λ without encapsulation has a delivery cost of 1 unit, transfer packets through a

tunnel incurs a penalty $\tau > 1$ and transmitting through a wireless link introduces an extra cost $\omega > 1$.

In a PMIPv6 solution a packet transfer between a CN and a MN can be divided into three segments: i) segment between CN and LMA, ii) the tunnel between LMA and MAG and iii) the wireless segment between the MAG and the MN, so the packet delivery cost of the PMIPv6 scenario is the following:

$$C_{PD}^{PMIP} = \lambda(C_{CN-LMA} + \tau C_{LMA-MAG} + \omega C_{MAG-MN})$$

In the DMM case, the path followed by a packet depends on the prefix, if the MN is connected to the anchor the packet is not encapsulated, and if the MN is not connected to the anchor the packet is encapsulated to the previous anchor, then, the packet delivery cost is:

$$C_{PD}^{DMM} = \lambda\left(C_{CN-DMM_{GW}} + \omega C_{DMM_{GW}-MN}\right) + \lambda_{pref}(N_{pref} - 1)(N_{pref}\,\tau C_{SDMM_{GW}-DDMM_{GW}})$$

Focusing only on the packet delivery cost of the packets within the mobility domain, leaving apart the cost related to reach the domain and the packet delivery on the last wireless hop, we assume: $C_{CN-LMA} = C_{CN-DMM_{GW}} = 0$ and $C_{MAG-MN} = C_{DMM_{GW}-MN} = 0$. In order to compare the DMM solution against the legacy PMIPv6 we present in Figure 78 the ratio between the packet delivery cost of both approaches:

$$\frac{C_{PD}^{DMM}}{C_{PD}^{PMIP}} = (N_{pref} - 1)\frac{C_{SDMM_{GW}-DDMM_{GW}}}{C_{LMA-MAG}}$$

The different curves represent the ratio of the distance in terms of delay between DMM-GW and the distance between LMA and MAG.

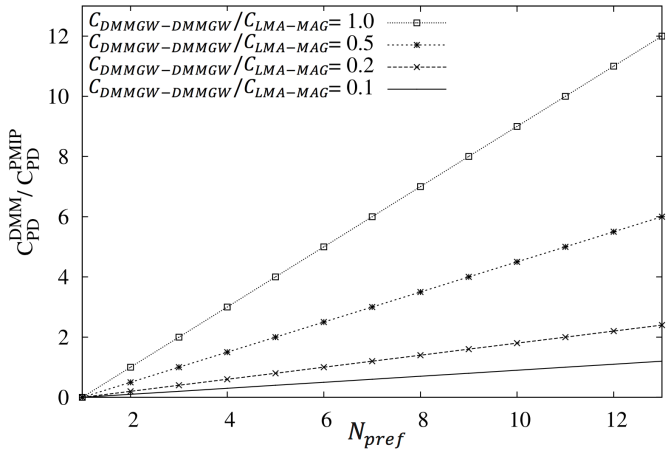


Figure 78: Packet delivery cost DMM vs. PMIPv6

The benefits obtained from DMM depends on the ratio $\frac{C_{DMM_{GW}-DMM_{GW}}}{C_{LMA-MAG}}$, in large operator networks with high distance between LMA and MAG a DMM solution reduce significantly the cost of the network. The DMM lower packet delivery cost allows more traffic, maximizing the usage of the infrastructure and increasing the node density.

9.4.4 Summary

As a summary, the tests related to the User Monitoring Manager and the Notification Manager are focused on the validation of the correct operation of the application. The user detection and notification procedure work as expected, we have validated the correct operation of the application.

In the case of the Mobility Manager we have shown that DMM introduces a lot of benefits compared to PMIPv6, avoiding bottlenecks and the single point of failure of the LMA. As we showed in the handover signaling cost graphics, DMM suffers a performance degradation with long-lived IP flows. The results obtained from the experiments show that the SDN-based implementations have a better performance in handover signaling cost and flow recovery than the PMIPv6-based. The solution can be implemented easily and provides a lot of flexibility due to the softwarization of the SDN paradigm. In short flow scenarios DMM allows a higher throughput reducing the overhead and in large deployments DMM decreases the packet delivery cost allowing more traffic increasing the node density.

10 Mobility Management Application (MMA) for High-Speed Train Scenario

For high-speed train scenario adopting two-hop architecture, i.e. passengers use on-board point-of-access points which are then backhauled by ground-to-train communications, the outbound gateway(s) providing ground-to-train moving backhaul perform handover frequently to maintain the availability and quality of moving backhaul. Current LTE systems adopt break-and-make handover, which inevitably causes interruption and throughput degradation during handover.

Conventional handover is composed of four stages:

1. **HO Measurement**: in addition to current serving eNB, UE also measures the signal quality of neighbor eNBs and reports to its serving eNB.
2. **HO Preparation**: once the signal quality of some neighbor eNB reaches some criteria compared to serving eNB, serving eNB starts negotiation with the neighbor eNB (called target eNB) to reserve/allocate resource for upcoming handover.
3. **HO Execution**: serving eNB commands UE to disconnet and which target eNB to connect immediately.
4. **HO Completion**: after the UE completes attachment, the target eNB notifies MME to switch the path it sends packet through.

One issue of the conventional handover procedure is related to packet redirection during handover. Before path switch command is issued in HO completion stage, MME has no idea about the handover at all. Therefore, each and every packet for UE is sent to original serving eNB first, who then re-routes the packets to the target eNB, and it prolongs the round-trip-time of packets with negative effect on the throughput. Since the handover procedure is strictly controlled by LTE handover control signals, this issue is seldom addressed because it involves the orchestration among eNBs (serving and target) as well as MME and the serving GW.

With Crosshaul, MMA for High-Speed Train (HST) is proposed to address the aforementioned issue in a transparent manner. The main idea behind MMA for HST is to act as a proxy between eNBs and MME and send the HO control signal on behalf of them. By doing so, the HO completion stage can be performed in parallel with HO execution stage, and the path switch doesn't need to wait until HO execution is done.

10.1 Consolidated design

As depicted in Figure 79, MMA for HST runs on top of XCI. Also, it is composed of:

1) HO detection

- XFE forwards specific packets to the controller for inspection and analysis.
- It can be performed all the time or in a context-aware basis

2) HO execution

- RAN and CN proxies (as an MMA function) communicate with CN and RAN on behalf of the real ones, respectively.
- Different HO stages, which are assumed to happen in order, can now happen simultaneously.
- MMA handles the interaction among RAN, CN and the proxies to assure the causal consistency.

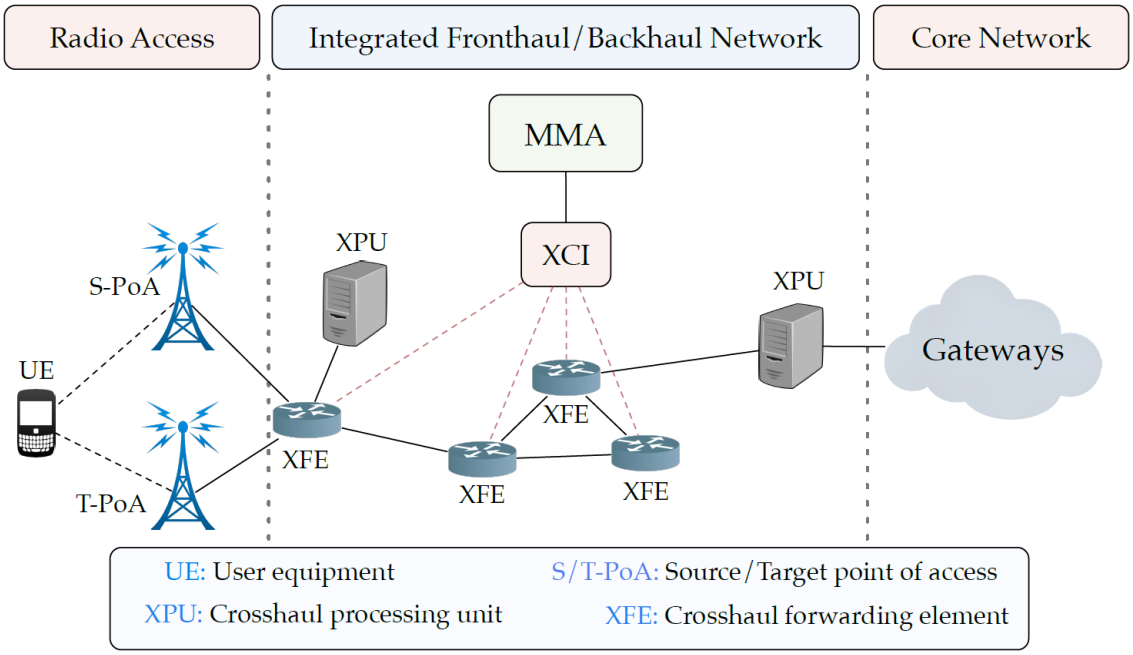


Figure 79: MMA interaction with 5G-Crosshaul for HST use case

To execute the MMA in 5G-Crosshaul platform, we can detect the HO decision by inspection the X2AP protocol with the help of XCI inspected the X2AP packets forwarded by XFE. If the HO is detected, this will trigger MMA to start HO execution module. Besides, the path switching request will be inspected similarly and then provide the new path outcome from execution module. This can be summarized in MMA Algorithm as depicted in Algorithm 4.

Algorithm 1: MMA Algorithm

Input: X2AP information input XCI
Output: HO decision detection
1 Information parsing;
2 **for** *each XFE duplicate and forward X2AP information for XCI* **do**
3 IF Find New eNB UE X2AP ID, Old eNB UE X2AP ;
4 MMA ← Start HO Excution and Completion;

Input: S1AP information input XCI
Output: Path switching detection
5 Information parsing;
6 **for** *each XFE duplicate and forward S1AP information for XCI* **do**
7 IF Find eNB to UE S1AP ID ;
8 MMA ← provide the pathswitching Immediately;

Algorithm 4: the MMA procedure

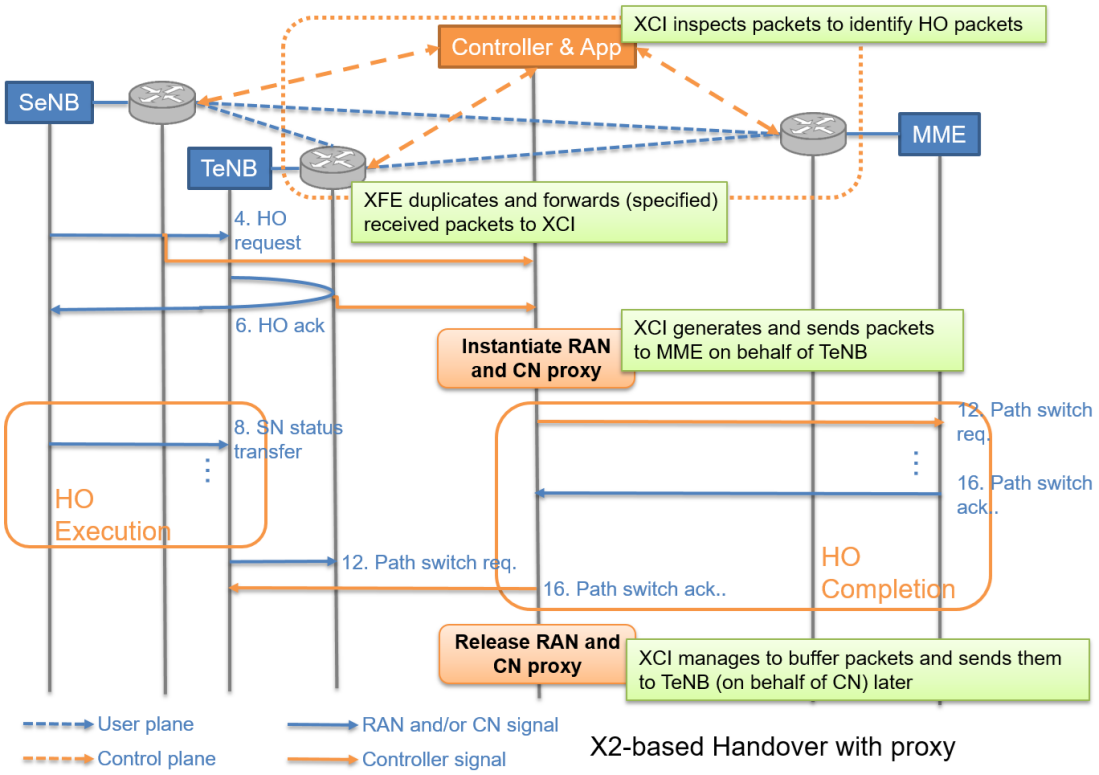


Figure 80: XCI and XFE role in MMA Algorithm

10.2 Implementation

The scenario considers a high-speed train, and the UE in Figure 83 which hands over between base stations is actually the outbound gateway (such as CPE). It is installed on board to provide moving backhaul for on-board point-of-access points.

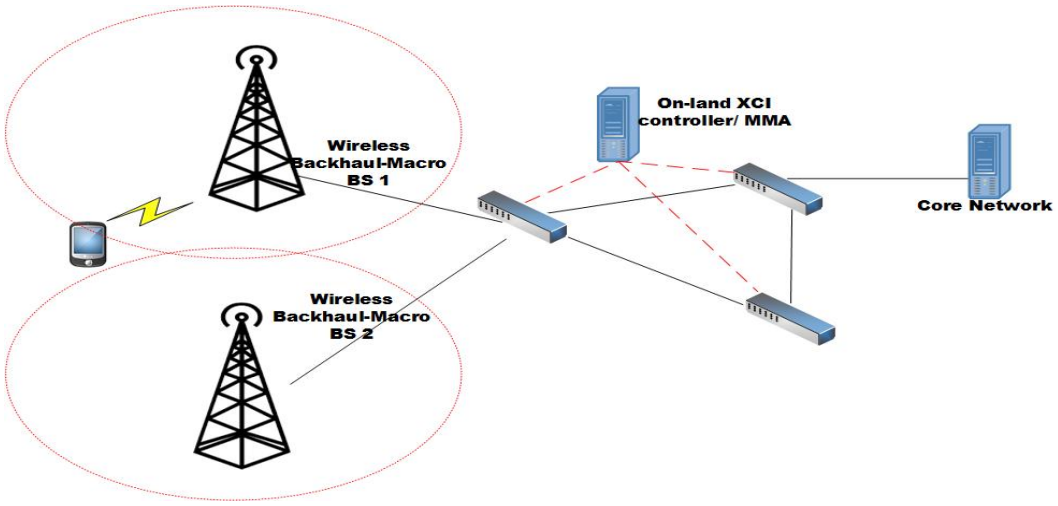


Figure 81: MMA for high-speed train scenario

10.2.1 High-level Design

The high-level design of the MMA implementation is detailed in Figure 82, which explains the different components and external APIs used.

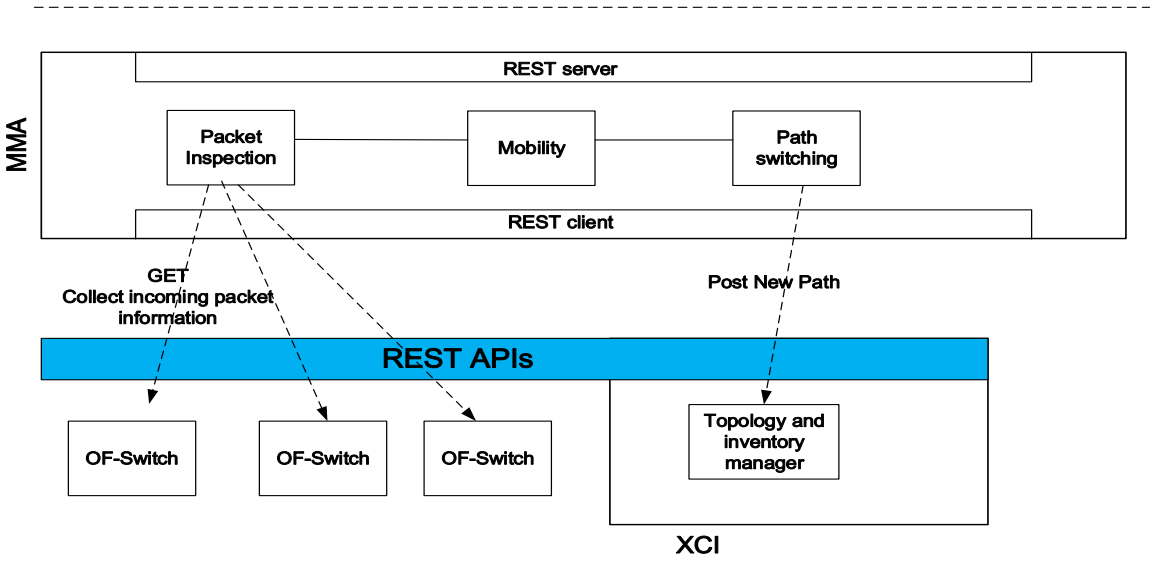


Figure 82: MMA High-level software design

10.2.2 Implementation Details

The main internal components are described in the table below.

Table 26: MMA application components

Component	Description
REST Client	Entity in charge of interacting with the XCI components via REST APIs.
Packet inspection	Entity in charge of inspecting and tagging the packet w.r.t to different HO messages
Mobility	Entity in charge to detect the HO request and trigger path switiching in advance.
Path switching	Request new path from core network
Topology and inventory manager	Entity in charge of storing the allocated topology and resources of tenants. Information can be recalled via Web based GUI.

10.3 KPIs

Table 27: KPIs addressed by the MMA

<table>
<tr><td>List of related project KPIs</td><td colspan="3">Obj.6. Design scalable algorithms for efficient 5G-Crosshaul resource orchestration
(5GPP KPI) Increase of total 5G-Crosshaul network throughput by > 20%</td></tr>
<tr><td rowspan="3">Measurement of project KPIs as well as application specific ones</td><td>KPI metrics (unit)</td><td>Description</td><td>Way of measurement</td></tr>
<tr><td>HO interruption time reduction (ms)</td><td>Reduce time (ms) required for HO process</td><td>Experiment (traditional HO) and analysis</td></tr>
<tr><td>Throughput increase (%)</td><td>Percentage of increase of the total 5G-Crosshaul network throughput during handover due to MMA for HST</td><td>Analysis and/or simulation</td></tr>
</table>

List of the State-of-the-Are approach **(used to compare with proposed solutions)**	LTE Legacy Handover mechanism: In the legacy LTE, the handover process passes through 3 stages defined as HO preparation, HO execution and HO completion. In this application we execute HO preparation and completion at the same time which will improve HO processing time. Moreover as a result, the throughput increases during HO (X2 HO type) around 5% and save delay time for redirected packets during HO in average of 6.5 ms. State of the art approach: The state of the art [58] [59] [60] [61] did not consider the reduction of HO time using the transport network or redirect packets during the HO. While our proposed MMA did consider both elements.

10.4 Validation and Evaluation

Simulation is carried out using MATLAB illustrated in. With the help of MMA connected to the switches close to eNB and core Network, MMA can detect the handover by inspecting the packets at switch 1 and establish new path for traffic coming from core network by configuring (OpenFlow) rules in switch 1 and 2.

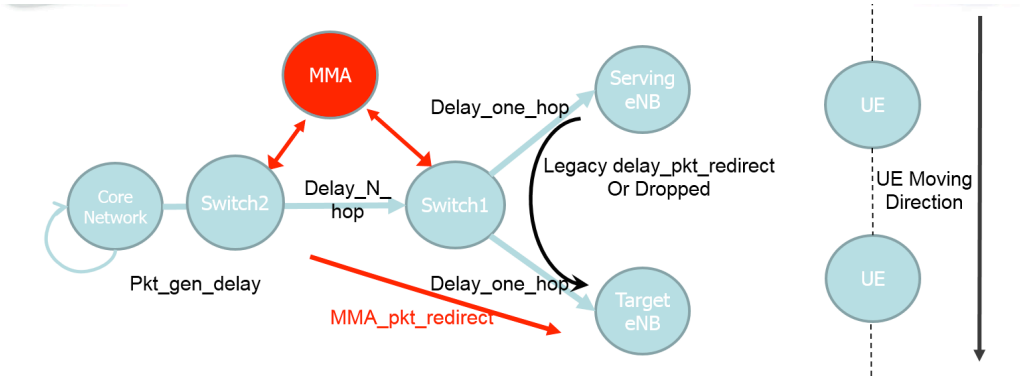


Figure 83: MMA simulation environment

Moreover, the simulation is built considering the legacy LTE environment, where the simulation setup has the following inputs (Table 28).

Table 28: Parameters used in LTE environment

Parameters	Assumption
Cell Layout	Hexagonal grid – 7 sites, 3sectors per eNodeB
Carrier frequency	2 GHz
Number of physical resource block (PRB)	50
Number of subcarrier per PRB	12
System Bandwidth	10 MHz , 180 kHz per PRB
eNodeB Tx Power	46 dBm
Number of UE per sector	10
Traffic Type	Full buffer
Handover Margin	1 dB
L3 sampling interval	200 TTI
L3 filter coefficient	4
Averaging window (N_{av})	5 and 6
UE direction	Range [0°,360°]
UE speed	300 km/h
UE noise figure	7 dB
UE position	Uniform distribution
Packet Scheduler	Proportional fair
Path loss	$128.1 + 37.6\log_{10}$ (R in km) dB
Shadow Fading	Standard deviation = 8 dB Correlation mean = 0 Correlation between eNodeB = 0.5
Fast fading	Winner channel Model

Figure 84 represents the comparison of throughput improvement with and without MMA. The x-axis represents the number of redirected packets during the handover process while the y-axis represents the throughput percent improvement during the handover. In both cases the throughput will decrease, but since MMA reduces the HO process time, it will benefit of 5% throughput gain using 10% of the time. We thus conclude that throughput improvement is achieved.

Figure 85 represents the comparison of average delay per redirected packets during HO with and without MMA. The x-axis represents the number of redirected packets during the handover process while the y-axis represents the average delay per redirected packets. In MMA case, the redirect packet time approaches to zero, since it is redirected in advance from the switch close to core network. Without MMA, the redirect packets

will suffer from delay due to redirect time needed to resend the packets from serving eNB to the target eNB during handover process.in average the redirected delay per packets is approximately 7ms.

Figure 86 represents the comparison of the handover processing time improvement with and Without MMA. The x-axis represents the number of redirected packets during the handover process while the y-axis represents the HO processing time. In MMA case, the average delay is reduced by 20ms by initiate the HO completion process in advance compare with the legacy HO process time in High-speed train.

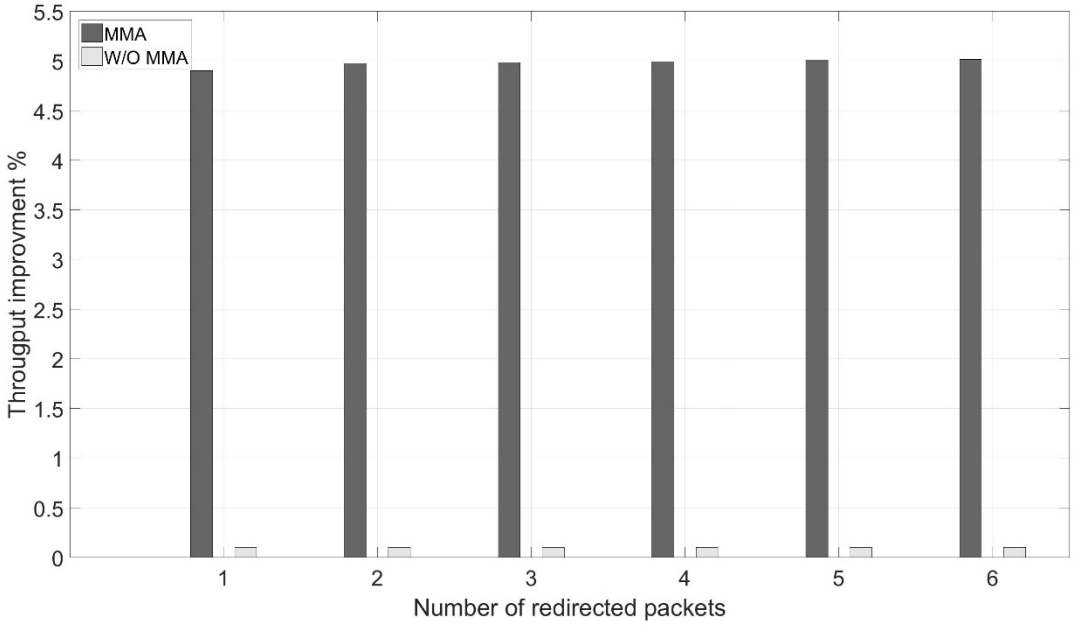


Figure 84: Comparison of Throughput improvement with MMA and W/O MMA

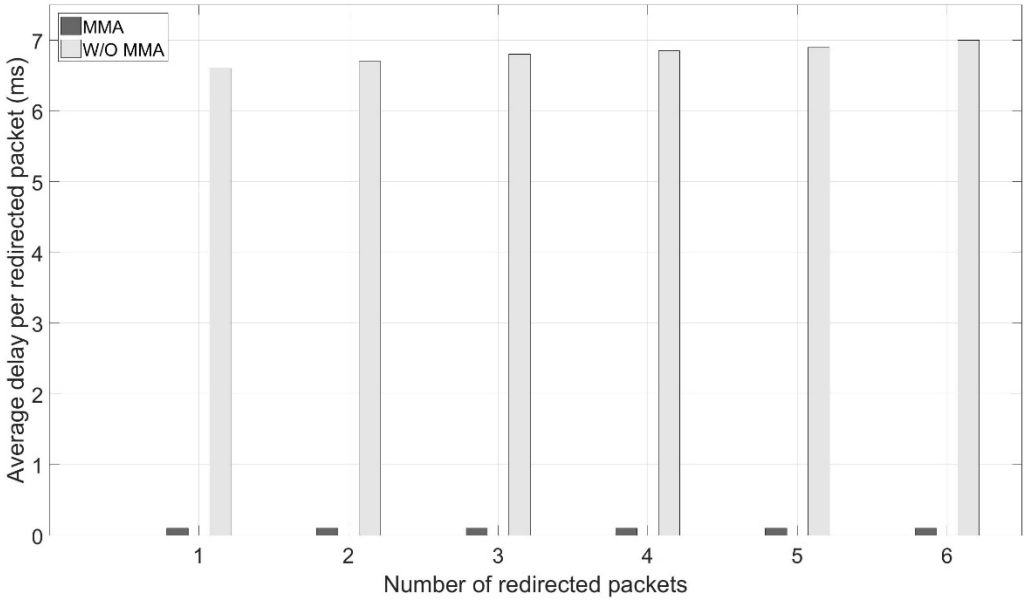


Figure 85: Comparison of Average delay per redirected packets during HO with MMA and W/O MMA

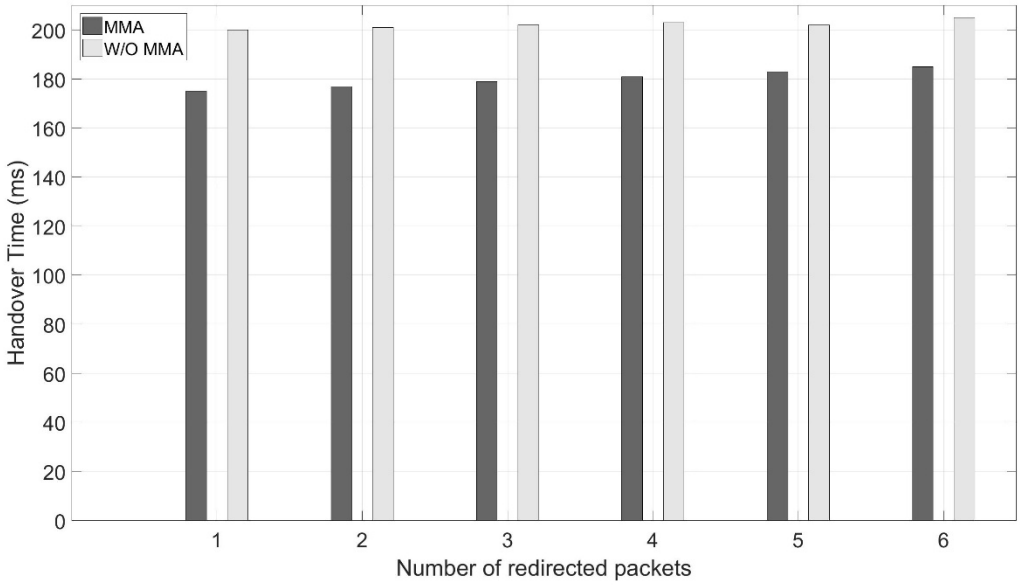


Figure 86: Comparison of HO processing time with MMA and W/O MMA

11 Multi-Tenancy Application (MTA) for High-Speed Train Scenario

MTA for High-Speed Train (HST) scenario consists of two parts: On-board MTA and On-land MTA. On-board MTA is implemented in HST and is composed of Radio Access Network (RAN) for users (i.e., User Equipment (UE)) and outbound gateways (i.e., CPEs) for connection to outdoor wireless backhaul network (i.e., Macro Base Stations (MBSs). On-land MTA is implemented in a transport network which is shared among different network operators. Traffic of different network operators transmitted from HST can be forwarded to their respective Core Networks (CNs) according to the configuration/pre-configuration by On-land MTA applied to the shared transport network.

11.1 Consolidated design

11.1.1 Methodology of On-board MTA in HST

On-board MTA is responsible for the radio access among the UEs from different network operators. Each UE in HST can connect to the Base Stations (BSs) installed in HST. On-board MTA has two modules, namely Packet Inspection Module (PIM) and Load Balancing Module (LBM).

PIM can extract the information of network operator from a packet sent by UE and know which network operator is related to this packet. Once the network operator of this packet has been determined, the On-board MTA generates a packet header which is dedicated to this network operator. After finishing the process, this packet is transmitted into the transport network using an appropriate outbound gateway decided by LBM in HST.

LBM in HST can select an available/appropriate outbound gateway with sufficient residual bandwidth to transmit the traffic from UEs of different network operators. When LBM receives a packet from PIM, it starts by checking the residual bandwidth of which outbound gateway can support to transmit by using Eq. (1)

$$B_i^{Outbound} - B_i^{Accumu.} \geq S^{PKT}, i \in [1,2,\dots,m_1], \quad (1)$$

where $B_i^{Outbound}$ is the bandwidth provided by outbound gateway i, $B_i^{Accumu.}$ is the accumulative bandwidth of outbound gateway i, S^{PKT} is size of the UE packet, m_1 is the number of outbound gateways in HST. On-board MTA is more related to RAN sharing in HST. It handles the radio resource of Point of Accesses (PoAs) in HST and configures operator-level information in the sharing radio resource in HST. After transmitted from HST, the UE traffic from different operators will be injected into the

transport network to be forwarded to their respective core networks. ON-land MTA will take care of the traffic transmission from UEs in HST for different operators.

11.1.2 Methodology of On-land MTA in HST

On-land MTA is responsible to distribute and maintain slices in the shared transport network. As shown in Figure 87, an infrastructure provides a Transport Network (TN) to connect RAN and CNs. To fulfill the tenants who want to use the TN, in parallel On-land MTA is responsible for handling the requirements of the tenants, e.g., the requirements from Mobile Virtual Network Operators (MVNOs). After receiving the requests from tenants, On-land MTA will process the information provided by shared tenants, generates the corresponding information such as Network Slice Identifier (NSID), Quality of Service Identifier (QoSID), and update the related tables which will be located internally and only be accessed by On-land MTA.

On-land MTA will provide a service template to the tenant/network operator who triggers a sharing request to TN. A new network operator takes part in the shared TN. The XCI (i.e., SDN controller) will follow the methodology in Figure 88. In particular, the new operator will interact with On-land MTA using the open interface and send a network sharing request in TN. On-land MTA will ask whether there is enough resource to satisfy the requirements of this network operator (via a service template provided by On-land MTA) and decide whether it will assign a resource slice to this network operator.

Each tenant can dynamically send a request to On-land MTA to update its information in the corresponding table generated by On-land MTA. Furthermore, each tenant is isolated from each other and only communicates with On-land MTA. The isolation can allow the tenant only to see the network resource allocated by On-land MTA without knowing the status of other tenants. The tenants can focus on the allocated network resource and freely allocate the available bandwidth to the services it provides.

Figure 89 is an example to show what On-land MTA information will be generated and stored for the shared tenant. As depicted, it can be seen that after processing the information sent by the requested tenant (e.g., VNO in Figure 88), if the requirement of this tenant can be fulfilled by currently available resource, On-land MTA will allocate a Tenant Identifier (TenantID), Network Slice Identifier (NSID) and Quality of Service Identifier (QoSID) to this tenant and keep the information internally.

The table generated by On-land MTA follows four operators described in the below:

- Create the database in the XCI (i.e., SDN switch) in shared transport network where the QoS criteria are meant to be applied.

- Define the queues according to the information received from the tenant. On-land MTA will allocate a TenantID to the tenant and also allocate an NSID to the requested tenant to distinguish each service that is defined by the tenant.
- Mark the packets with the corresponding NSID in Ingress SDN switch.
- Match the packets by the SDN switch except for the Ingress switch according to the flow table rules.

After completing the above operations, the requested tenant can start to use the allocated resource to transmit the traffic. It can be seen that On-land MTA can make multi-tenants individually to obtain the network resource without knowing the situation of physical infrastructures.

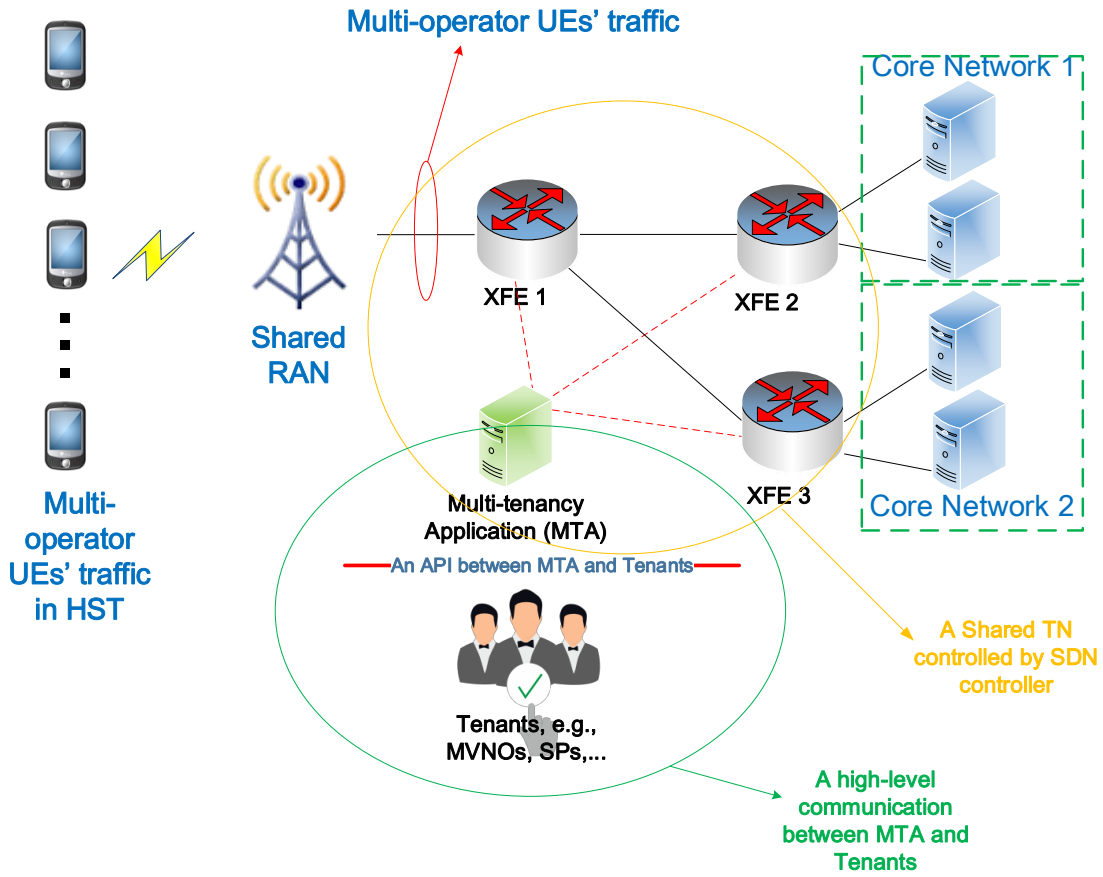


Figure 87: A simple example of two network operators sharing the transport network

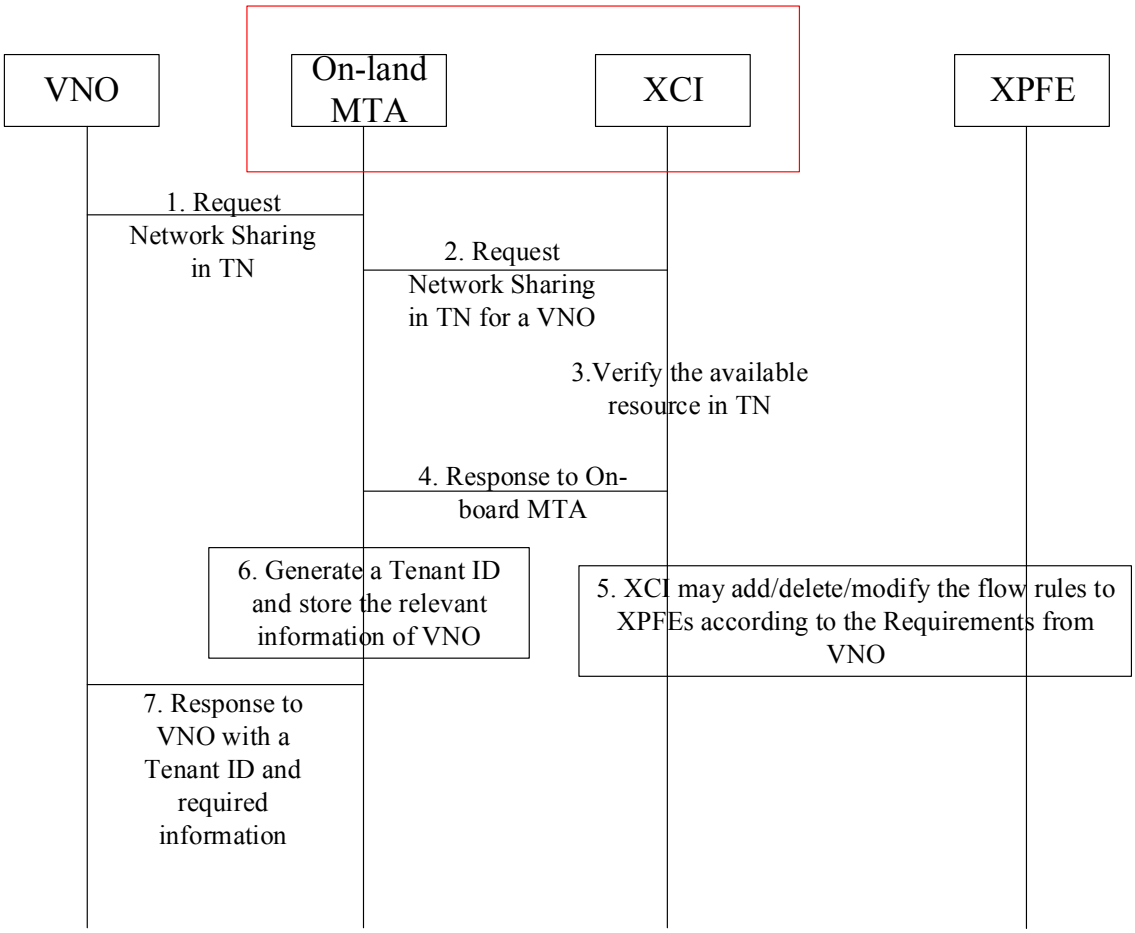


Figure 88: Methodology for a VNO to request Network Sharing in TN

NSID	Name	IP	Port	QoSID	Min Bandwidth (Kbps)	Max Bandwidth (Kbps)
1	abc	10.0.0.5	5002	1	500000	1000000
2	xyz	10.0.0.6	5002	1	200000	1000000
3	abc	10.0.0.5	5003	2	400000	1000000
4	xyz	10.0.0.6	5003	2	500000	1000000

Figure 89: An example of related table consisting of the corresponding information generated by On-land MTA.

11.2 Implementation

The MTA application is developed as a Python application implemented and operating on top of 5G-Crosshaul XCI services, utilizing the XCI services by REST APIs. It implements the following main functionalities:

- Onboard controller inspects the packet to analyse the information of the network operator and tags the packet w.r.t network operator. It also performs the load balancing to distribute the traffic load to make efficient use of the bandwidth of outbound gateways on-board.
- On Land controller manages the shared network w.r.t tagged packets and re-routes the packet to their appropriate core network. It also provides a web based GUI to the network operators allowing them to maintain and manage their own network topology in the shared infrastructure, where tenants can also initiate multiple services utilizing their allocated topology and resources.

11.2.1 High-level Design

The high-level design of the MTA implementation is detailed in Figure 90, which explains the different components and external APIs used.

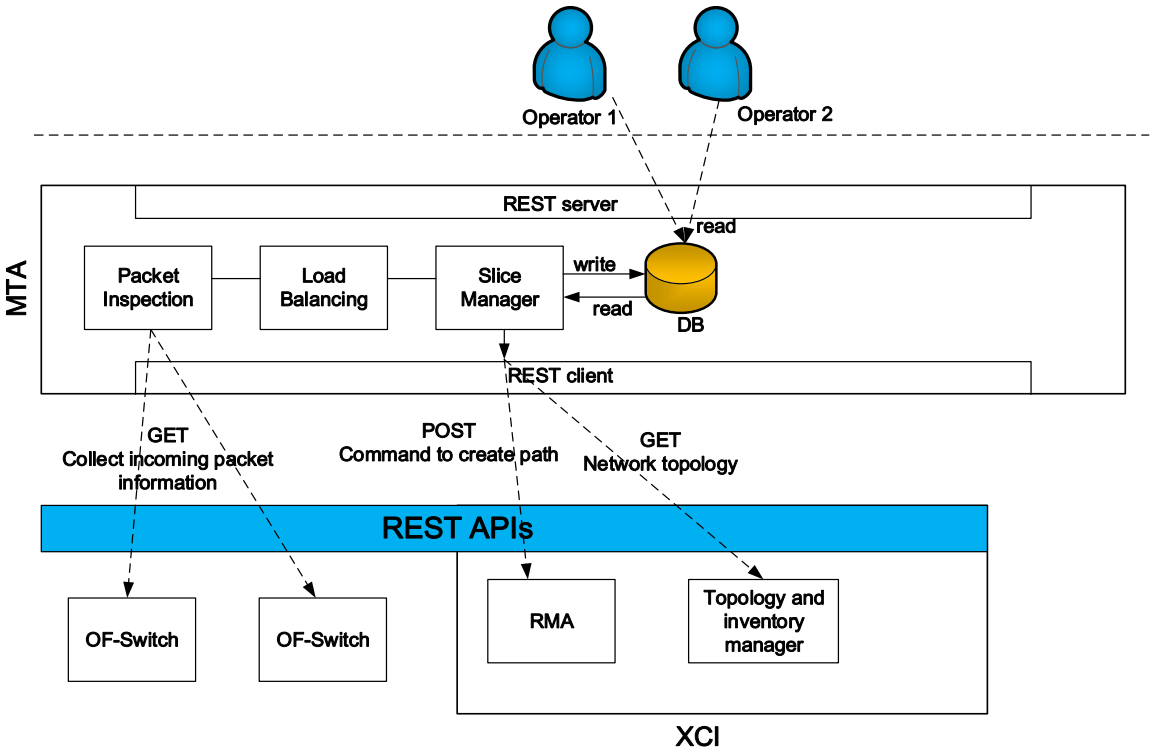


Figure 90: MTA High-level software design

11.2.2 Implementation Details

The main internal components are described in the table below.

Table 29: MTA application components

Component	Description
REST Client	Entity in charge of interacting with the XCI components via REST APIs.
Packet inspection	Entity in charge of inspecting and tagging the packet w.r.t to different network operators
Load Balancing	Entity in charge to distribute traffic load on-board in a balancing manner to make efficient use of the bandwidth of outbound gateways on-board.
Slice Manager	Slice manager manages the slice allocation and deallocation of network operators sharing the shared transport network and update slice information with on-board MTA (if any updated information is required.)
DB	Internal Mysql db to store tenant related information
RMA	Entity in charge to provide the route path within the allocated slice of the tenant
Topology and inventory manager	Entity in charge of storing the allocated topology and resources of tenants. Information can be recalled via Web based GUI.

11.3 KPIs

Table 30: List of KPIs addressed by MTA

<table>
<tr><td>Application</td><td colspan="3">MTA</td></tr>
<tr><td>List of related project KPIs</td><td colspan="3">Obj.5. Increase cost-effectiveness of transport technologies for ultra-dense access networks
(5GPP KPI) Reduce Total Cost of Ownership (TCO) by 30% by improved optical transmission and sharing mobile and fixed access equipment
Obj.6: Design scalable algorithms for efficient Xhaul resource orchestration

(5GPP KPI) Reducing the network management Operational Expenditure (OPEX) by 10%</td></tr>
<tr><td rowspan="2">Measurement of project KPIs as well as application specific ones</td><td>KPI metrics (unit)</td><td>Description</td><td>Way of measurement</td></tr>
<tr><td>Service Response Time (seconds)</td><td>The SRT is defined as the time difference between the time point in which On-land MTA has accepted the sharing request from the tenant and the time point in which On-land MTA has completed the generation of the corresponding information such as NSID and QoSID and so forth and response to the tenant with the completion of the</td><td>To obtain the time required to generate a table which is showed in Figure 92.</td></tr>
</table>

		operations. When the tenant receives the response from On-land MTA, it can start its service immediately.	
	OPEX saving (%)	Percentage of OPEX savings in the transport segment based on chosen OPEX models	Relatively compare the impact to cost reduction when SDN-based and traditional multi-tenancy mechanism applies to shared transport network.
	CAPEX saving (%)	Percentage of CAPEX savings in the transport segment based on chosen CAPEX models	
List of the State-of-the-Art approach **(used to compare with proposed solutions)**	State of the art approach 1: Static Sharing Each operator in this on land sharing environment will be allocated a fixed portion of bandwidth from each sharing equipment in transport network. If required bandwidth from the operator is larger than the fixed bandwidth which is allocated to this operator, the static sharing module can drop the packet of this operator. Static sharing bandwidth without considering the status of the required bandwidth from all operators such that it may fully utilize the bandwidth.		
Cost Models	CAPEX model OPEX model	MTA use two cost models to evaluate the feasibility of network sharing in terms of CAPEX and OPEX. **Cost Model 1** SDN influences Capex [62] [63] [64] (1) Cost savings that can be reached by using simpler network devices (2) Cost of extra components such as OpenFlow controllers, line cards and transceivers (3) The ratio of the number of switches that an OpenFlow controller can manager (4) The possibility to better align network capacity with actual demand SDN influence Opex [62] [63] [64] (1) Continuous cost of infrastructure: cost for power and cooling energy is reduced because SDN allows for better traffic steering which reduces the number of network devices and their power consumption. Energy cost will be even low for the sharing scenario but a high utilization of network equipment (2) Maintenance cost: software management for network equipment is reduced to a minimum of one compared to traditional network which has vendor-specific software for different devices	

		(3) Repair cost: sharing the equipment will reduce the costs for repair because each operator can take responsibility for a part of the network (4) Human resource cost: SDN reduces the amount of manual configuration require in the network (5) First time installation of network equipment cost: SDN creates a has robust testing abilities ahead of rollout and reduced the number of devices that need to be updated **Cost Model 2** Other than that cost model presented in WP1 (Refer IR 1.3) is also used to evaluate the feasibility of MTA in the high-speed train scenario.

11.4 Validation and Evaluation

Figure 87 shows a simple example of the network scenario where On-land MTA is implemented together with SDN controller (i.e., XCI). In this example, an infrastructure provides a TN to connect RAN and Core Networks (CNs). To fulfill the tenants who want to use the TN, in parallel On-land MTA is responsible for handling the requirements of the tenants, e.g., the requirements from MVNOs. After receiving the requests from tenants, On-land will process the information provided by shared tenants, generates the corresponding information such as Network Slice Identifier (NSID), Quality of Service Identifier (QoSID), and update the related tables.

In the environment settings (shown in Table 31), it is assumed that N tenants and M services as a pair of (N, M) will be requested simultaneously to On-land MTA to share the network resources. According to the operations described in section 11.1.2 of On-land MTA in HST, SRT will be impacted by the number of tenants which may request more than one service.

Each simulation is executed for 10 times to obtain the minimum, maximum and average time by using Ryu controller as the XCI (i.e., SDN controller) and Mininet to generate each XPFE (i.e., SDN switch) which is controlled by Ryu controller together with On-land MTA. The following subsection is to explain the simulation results in terms of SRT and the pair of (N, M).

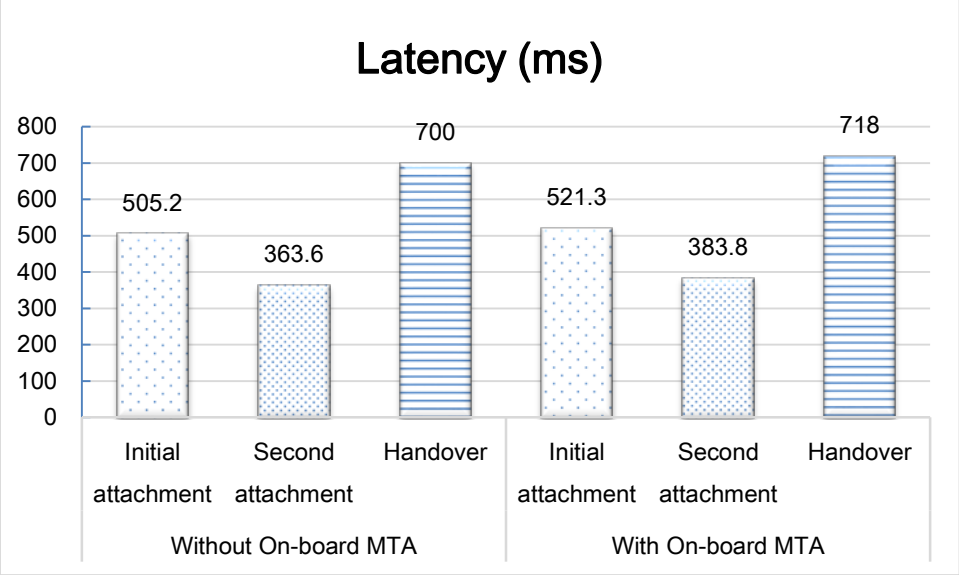
Table 31: Used Parameters

Used Symbol	Description
N	number of tenants sharing the transport network
M	number of services requested by tenants

Service Response Time (SRT)	the time difference between the time point in which On-land MTA has finished the operations mentioned in On-land MTA

11.4.1 On-board MTA

Figure 91 shows the emulation results of latency and throughput with/without On-board MTA installed in HST. Figure 91(a) shows the results that the latency of UE attachment and handover procedures. For the attachment procedure, it can be seen that On-board MTA middleware will not have a big impact on increasing the processing time. The time difference among first attachment, second attachment and handover procedures with/without On-board MAT will not be larger than 18 ms. It means On-board MTA is almost transparent between UE and the network operators. The processing time in On-board MTA will be a small ratio of the overall end to end latency. Figure 91(b) shows similar results that can be seen in term of throughput. The iPerf (https://iperf.fr) utility is used to verify the throughputs for Uplink (UL) and Downlink (DL). The ideal throughput of the testbed was 36 Mbps for uplink and 110 Mbps for downlink, while the average without sharing proxy was 27.1 Mbps for uplink and 91.25 Mbps for downlink. With the same configuration, the average of On-board MTA architecture was 26.3 Mbps for uplink and 90.7 Mbps for downlink, or 97% of the results without On-board MTA. The impact of On-board MTA on the throughput can therefore be ignored (3%). The difference was caused by the packet header overhead of IPsec tunnels created by the On-board MTA, and determined by its packet forwarding capability.



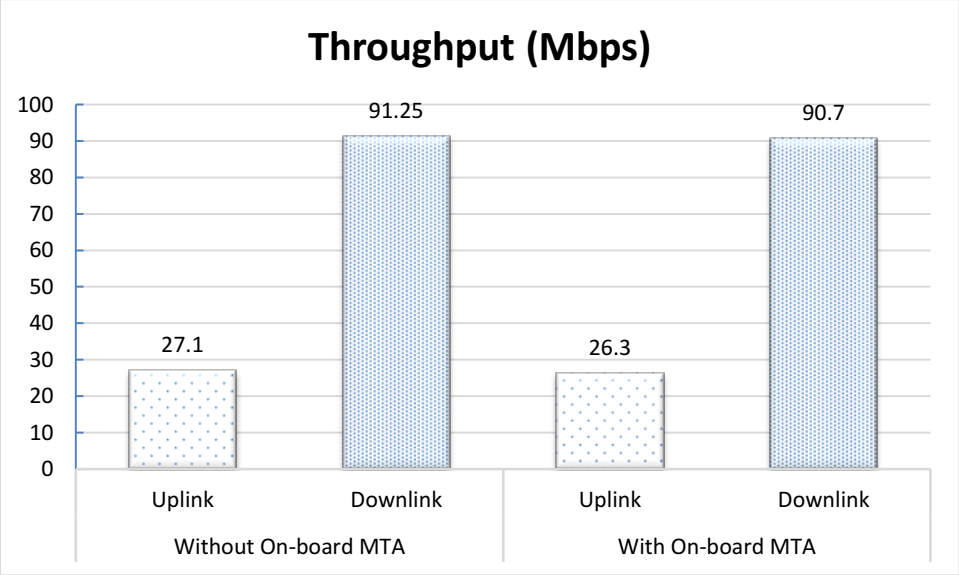


Figure 91: Emulation results of the latency for attachment (top) and the throughput (bottom)

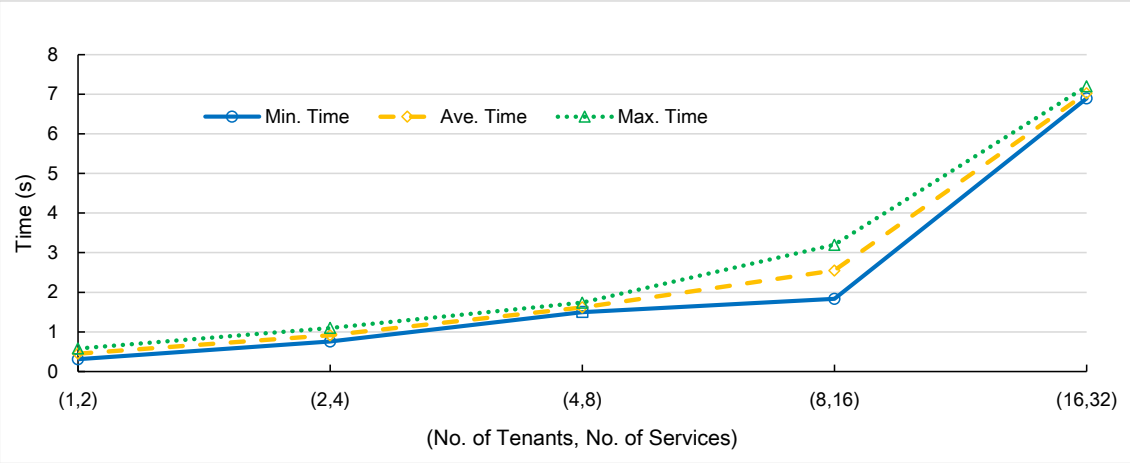
11.4.2 On-land MTA

Figure 92 shows the simulation results of SRT. In Figure 92(a) and Figure 92(b), it can be seen that even if the total number of tenants and services are increased to 512 tenants and 1024 services, the required time to response to all tenants simultaneously for their services takes around 240 seconds (i.e., 4 minutes). Also, when observing the value of SRT, it can also be found that the value of SRT time will increase up to 2 folds when doubling the number of tenants and services. The reason behind the increment in SRT is that when a tenant is added to share the network resource it may request more than one service to On-land TA. As mentioned in 2.10.2.2 Methodology of On-land MTA in HST, each tenant needs to be allocated TenantID, NSID and QoSID according to the requirements. For SRT, the information of each service from each tenant shall be processed by the On-land MTA operations. Therefore, it is reasonable that SRT will increase as the number of tenants and services increase. The overhead of On-land MTA may increase exponentially due to processing the corresponding information, i.e., TenantID, NSID and QoSID. From the simulation results, it may have a trade off between the On-land MTA and the tenants to keep the overheads under an acceptable range.

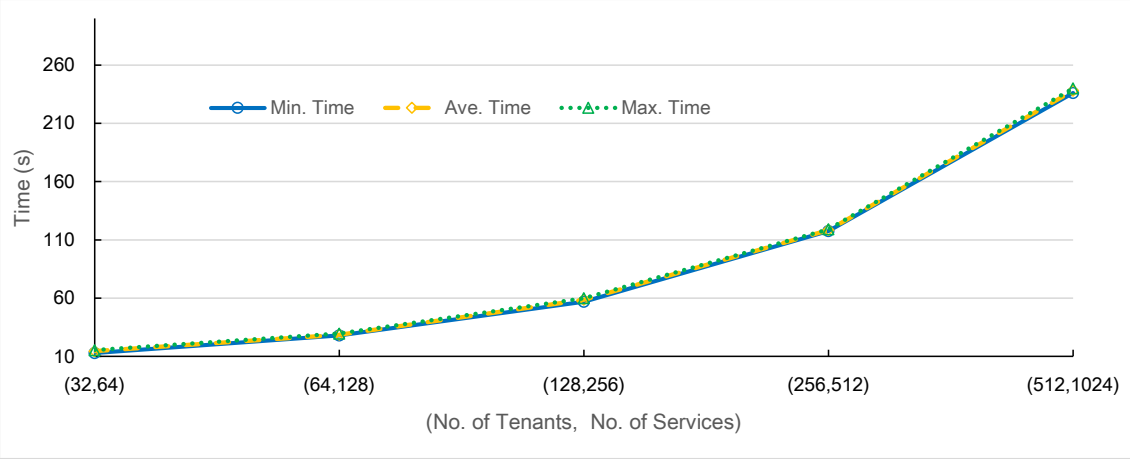
Figure 93 shows the results of reduction for CAPEX and OPEX by comparing legacy network and Crosshaul network using the entire IP traffic. The results are calculated by using the cost model 2 described in Section 11.3 KPIs. From the results, it can be seen that in regardless of the sharing percentage setting from 0.1 to 0.9, the cost of CAPEX by using Crosshaul as network can reduce cost and the results meet the CAPEX KPI mentioned in the table in Section 11.3. However, for the OPEX, when sharing percentage is set from 0.1 to 0.5, it has similar OPEX cost. When the sharing percentage

is set equal or larger to 0.6, the OPEX cost will increase since the network needs more effort to manage the different sharing tenants (i.e. network operators).

11.4.3 CAPEX and OPEX for MTA



(a)



(b)

Figure 92: Service Response Time (SRT) (a) from (1,2) to (16, 32) (b) From (32, 64) to (512, 1024).

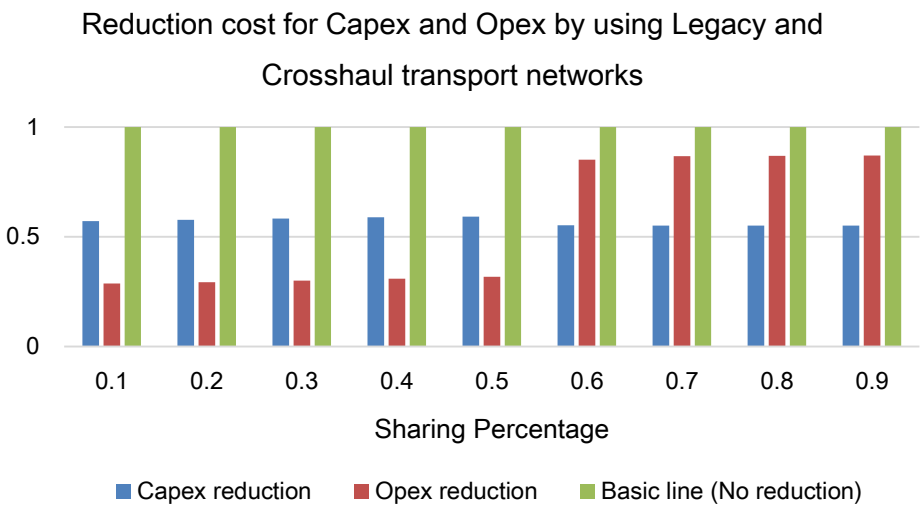


Figure 93: Reduction for CAPEX and OPEX by using the CAPEX an OPEX of Legacy Transport Network as Baseline

12 Content Delivery Network Management Application (CDNMA)

The Content Delivery Network Management Application (CDNMA) is an application related to the distribution of media content that uses the services and APIs offered by the XCI NBI, and other applications like the RMA and MMA, to deploy and manage a vCDN service in which the CDN functions (CDN nodes –origin and replica servers- and load balancer) are dynamically allocated across the 5G-Crosshaul network.

A typical content delivery network is an infrastructure composed of replica servers located at the edge of the network to which the end-users are connected. In this infrastructure, the content based on the user request is obtained from the origin server and a user is served with the content from the closest replicated server. In this sense, the end users are communicated with a replicated CDN server close to them and receive the content from that server.

Based on this approach and taking advantage of the 5G-Crosshaul infrastructure, the application follows two main workflows, shown in Figure 94. As a first step, the CDNMA interact with the XCI NBI in order to deploy the vCDN infrastructure exploiting the virtualization capabilities and leveraging the NFV platforms, deploying

the content replica servers as VNFs instead of hardware appliances. At this point, it is important to mention that based on the multitenancy definition from the operator point of view, a vCDN infrastructure service considers essentially the "push mode" of resource awareness, meaning that, the CDN operator needs to previously know the network topology and location of compute nodes in order to decide where deploy the CDN nodes, so, the CDN nodes placement is decided by the CDN operator based on the network topology view and the location of the compute nodes obtained by the CDNMA from the SDN controller. In this stage, the CDNMA requests the vCDN infrastructure configuration (network service) to the NFVO, providing the Network Service Descriptor (NSD) composed of the VNFs (origin and replica server nodes) and the VNF-Forwarding Graph (VNF-FG). The NFVO together with the VNFM and the VIM manages the computing resources instantiating the CDN nodes on virtual machines deployed within servers located close to the end users. The content delivery infrastructure is implemented via networks of virtual content caches –replica servers- which are deployed on the XPUs connected to the XFEs across the 5G-Crosshaul network.

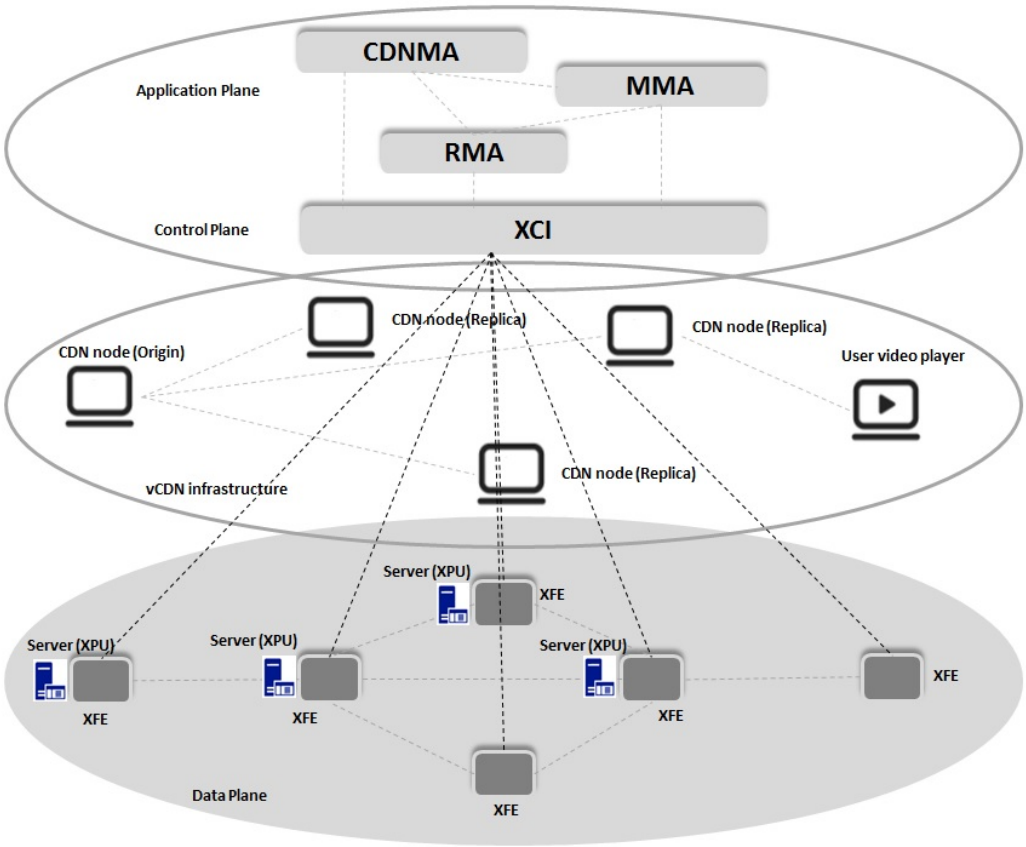


Figure 94: CDN Management Application

On the other hand, once the vCDN infrastructure is running the CDNMA interact with the MMA and RMA applications to manage the vCDN service, providing an efficient and flexible way of delivering content to the end users, especially Video on Demand

(VoD) and deal with the massive content requests from the users, controlling the load balancing over several CDN nodes strategically placed at various locations. The CDNMA receives the point of attachment of the user to the network from the MMA application and based on this information and the monitoring of the different CDN nodes and the policies defined by the CDN operator, decides the best CDN node to serve the user request. The RMA through its own algorithms and the XCI will provision and establish the route or path followed by the mobile users' traffic between the CDN node assigned and the point of attachment of the user to the network.

Here it is important to note the different inputs received by the application which consist of the monitoring information from the CDN nodes (status, load, number of connections), the information about the user location (network entry point) and the content delivery rules defined by the CDN operator, to control the load balancing between the different replica servers.

The CDN management application will improve the content delivery maximizing bandwidth and improving accessibility through the CDN infrastructure according the user demands.

12.1 Consolidated design

The CDNMA is a Web application composed of two basic algorithms that allow the CDN service management and implementation over the 5G-Crosshaul network. The first algorithm is in charge of the CDN infrastructure instantiation, and the second one is oriented towards the control and management of the service during the lifetime.

For the deployment of the CDN nodes on virtualized server instances we consider:

- a Network Topology NT = (S, H, L), where S is the set of switching nodes, H is the set of hosting (computing) nodes and L is the set of links;
- a CDN graph CDNG = (NT, CT, COC, NL), where NT is the network topology, CT is the CDN node type (origin or replica), COC are the CDN operator criteria for the deployment of the CDN nodes (location, capacity, redundancy...) and NL are the links between the CDN nodes;
- a vCDN infrastructure vCDNI = (CDNG, NSD), where CDNG is the CDN graph and NSD is the network service descriptor sent to the XCI.

```
Algorithm 1 vCDN Infrastructure deployment()
1:  NT Network Topology ( )
2:      Get network topology from XCI (switching nodes, computing nodes, links)
3:
4:  CDNG CDN graph ( )
5:      Network Topology ()
6:      CDN Operator Criteria for deployment
7:              (Select XPU for deployment based on the criteria: compute node, location, capacity, …)
8:      CDN Node type
9:              (Select where to deploy the origin and replica server)
10:     Logical interconnection between CDN Nodes
11:             (VNF-Forwading graph)
12:
13:  vCDNI vCDN infrastructure ( )
14:     Send complete CDN graph description to XCI
15:             (Network Service Descriptor)
```

Algorithm 5: vCDN Infrastructure Deployment

For the control and management of the vCDN service, we take into account the monitoring information received from the CDN nodes, the point of attachment of the user to the network and the content delivery rules provided by the CDN operator. The CDN Management CDNM = (CM, PU, CDR), where CM is the CDN nodes monitoring information (status, CPU load, memory load, number of connected users), PU is the point of attachment of the user, and CDR are the content delivery rules (thresholds, constraints, priorities…). Based on all of this information the CDN node assignments are performed to serve the user requests.

```
Algorithm 2 CDN service control and management ( )
1:  CDNM CDN Management ( )
2:      Get CDN nodes monitoring information
3:              (Receive status –up, -down, CPU load, memory load and number of connected users)
4:      Get Point of Attachment of the user to the network
5:              (Receive the MAC and IP address from the PoA and the MAC and IP addresses of
6:               the users from the MMA)
7:      Check Content Delivery Rules defined by the CDN operator
8:              (Evaluate the received monitoring information with the CDN rules)
9:              (Get the CDN Node to be assigned for the streaming)
10:     Send the CDN node assigment
11:             (Send the assigned CDN node to the Video Player for load balancing between
12:              replica servers)
13:             (Send the CDN node assignment to the RMA)
```

Algorithm 6: CDN Service and Control Management

12.2 Implementation

The CDNMA is a Web application implemented in Python and Django Framework for Python and developed from scratch that operates on top of the 5G-Crosshaul XCI services and interacts with the XCI NBI and with other applications, like MMA and

RMA. The CDNMA provides to the operator (e.g., CDN operator) a graphic interface running on a REST API for management and operation actions. This application generates the content delivery rules that are used by the Video Player application deployed on the origin server for the load balancing of the user requests.

The CDNMA implements the following main functionalities:

- Request the vCDN infrastructure instantiation to XCI NBI, providing a complete descriptor file with detailed information about the CDN elements and how they have to be configured and connected.
- Management and storage of specific monitoring information got from the CDN nodes.
- Management and treatment of the information exchanged with the MMA and RMA applications.
- Management and update of the content delivery rules over the CDN nodes placed in the network (CDN node assignments), based on the monitoring information received and the logic defined by the CDN operator.

12.2.1 High-level design

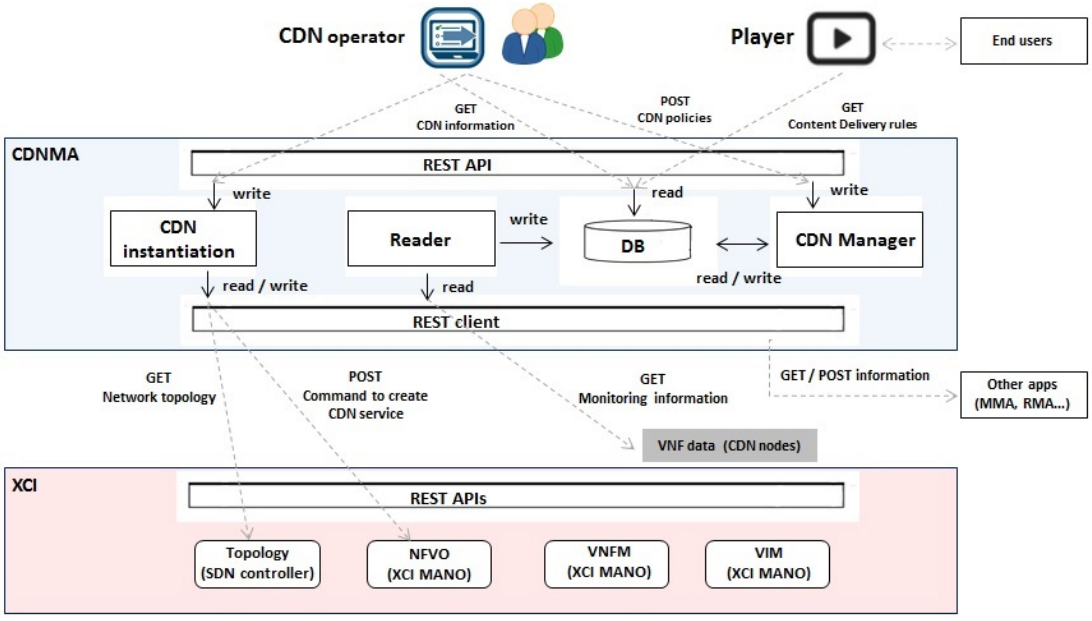


Figure 95: CDNMA high-level software design

12.2.2 Implementation Description

The main internal components showed in Figure 95 are described in the table below.

Table 32: CDNMA application components

Component	Description
REST Client	The entity in charge of interacting with the XCI components via REST

	APIs receives the monitoring information from the CDN nodes and interchanges specific information with other applications.
CDN instantiation	The entity in charge of the vCDN infrastructure instantiation. This entity through the REST APIs requests the network topology from the SDN controller, receives the CDN policies from the CDN operator regarding where to deploy the CDN nodes and sends the network service descriptor with the vCDN service to the NFVO.
Reader	The entity in charge of getting and processing the CDN nodes monitoring information.
DB	Internal no-SQL DB to store the monitoring information and the content delivery rules.
CDN Manager	The entity in charge of deciding the CDN node assignments, based on the CDN policies provided by the CDN operator and the monitoring information stored in the DB.

The CDNMA application uses the services and APIs offered by other applications and the XCI NBI, SDN controller and NFV MANO, in particular the NFVO (Orchestrator), to manage the CDN infrastructure configuration and the content delivery rules on the 5G-Crosshaul network. The interaction with the services offered by the other architectural components is the following:

- SDN controller services: get the network topology, retrieve a list of all known nodes, hosts and links from the topology manager.
    - Get nodes and all the links getting out/in from/to the node
        - XFEs and port connections
    - Get all link connections between nodes including their properties
        - Links (connection points, Bandwidth)
    - Get hosts and the node connections
        - XPUs and port connections

- NFVO services: instantiation and termination of the CDN service. The NFVO receives a Network Service Descriptor (NSD) from the CDNMA application to instantiate the CDN service. The Network Service Descriptor (NSD) contains the descriptors which describe components that are part of the CDN service.
    - CDN Node VNF Descriptor (VNFD)
    - CDN VNF-FG Descriptor (VNF-FGD)

- MMA services: retrieving information about the Point of Attachment (PoA) of the user. The MMA through a REST API sends a json file with the MAC of the point of attachment and the MAC and IP of the user to the CDNMA.

- RMA services: for requesting the computation and provisioning of the routes or paths followed by the mobile users' traffic between the selected nodes. The CDNMA through a REST API sends the identification of the two points (CDN node, PoA) to the RMA.

- CDN nodes monitoring information: the CDNMA application receives specific monitoring information from the different CDN node VNFs placed in the network.
    - Status (up, down)
    - CPU load (%)
    - Memory load (%)
    - Number of connected users (n)

12.2.3 Implementation Details

The CDNMA application follows a modular approach where each module contains everything necessary to execute the desired functionality and fulfill the requirements. The application interacts with other modules through well-defined interfaces.

The application implementation is divided in different blocks or modules which cover the requirements established for the CDNMA. This is the status of the work developed:

- **CDN nodes images (VNFs).** CDN nodes images have been created and will be deployed on virtualized server instances. There are two kinds of CDN nodes: the origin and replica server nodes. The origin server is based on an Ubuntu server and includes the storage folder for the videos and the Web video player application that provides the user service. The replica server is based on Wowza Streaming Engine media server software and stores the different videos cached from the origin server.
- **CDNMA authentication and welcome pages.** The CDNMA requires users authentication to access to the CDN service management for a specific tenant. The application has been deployed to allow multitenancy with one CDN service per tenant.

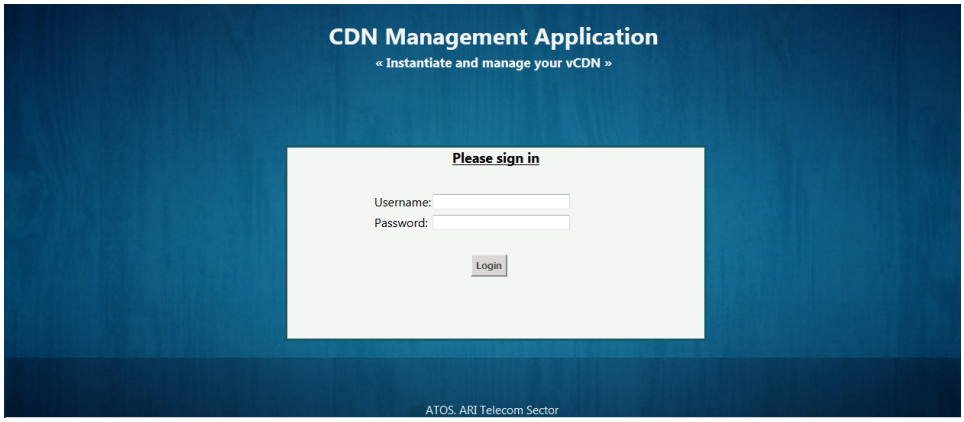


Figure 96: CDNMA Login page

- **CDN instantiation module.** It has been developed and tested exploring the feasibility of implementing a vCDN infrastructure in an automatic way. This module is composed of:

- a graphical user interface (GUI) through which the CDN operator can configure the CDN service (network service descriptor) (Figure 97),
- an interface to interact with the NFVO sending the network service descriptor file,
- an interface to get the network topology from the SDN controller,
- a GUI to display the network topology (Figure 98).

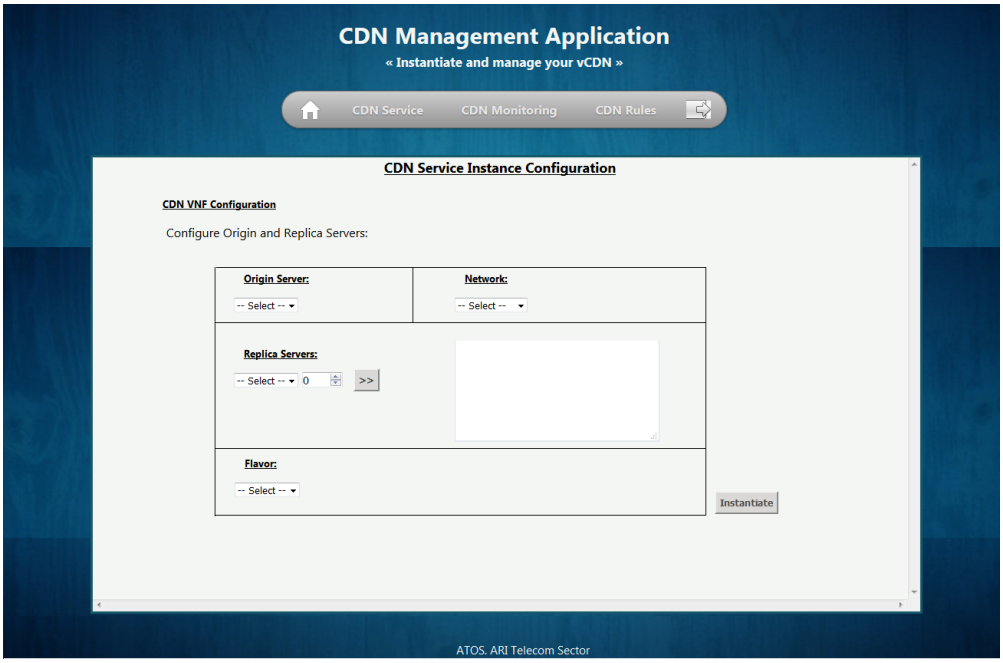


Figure 97: CDN instantiation module (CDN service configuration)

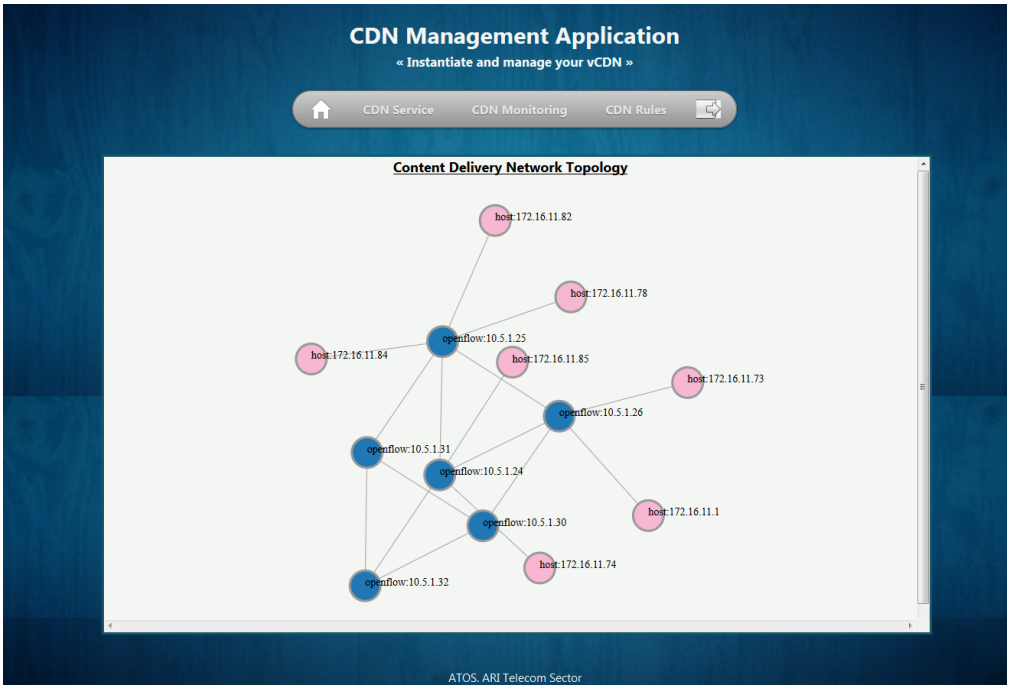


Figure 98: CDN instantiation module (Network topology)

- **CDN Management module.** It has been developed for the control and management of the vCDN service. This module is composed of:
  - an interface to get the CDN nodes monitoring information and save it in the database,
  - an interface to get the point of attachment and MAC and IP of the user from the MMA,
  - an interface to send to the Video Player the CDN node assigments (the replica server that will deliver the content to the user),
  - an interface to send the CDN node assigments to the RMA,
  - a GUI to make the monitoring information available for the CDN operator (Figure 99),
  - a GUI to manage the (un)registration to the MMA service and to display the information sent by the MMA about the users' points of attachments,
  - and a GUI to manage the content delivery rules defined by the CDN operator.

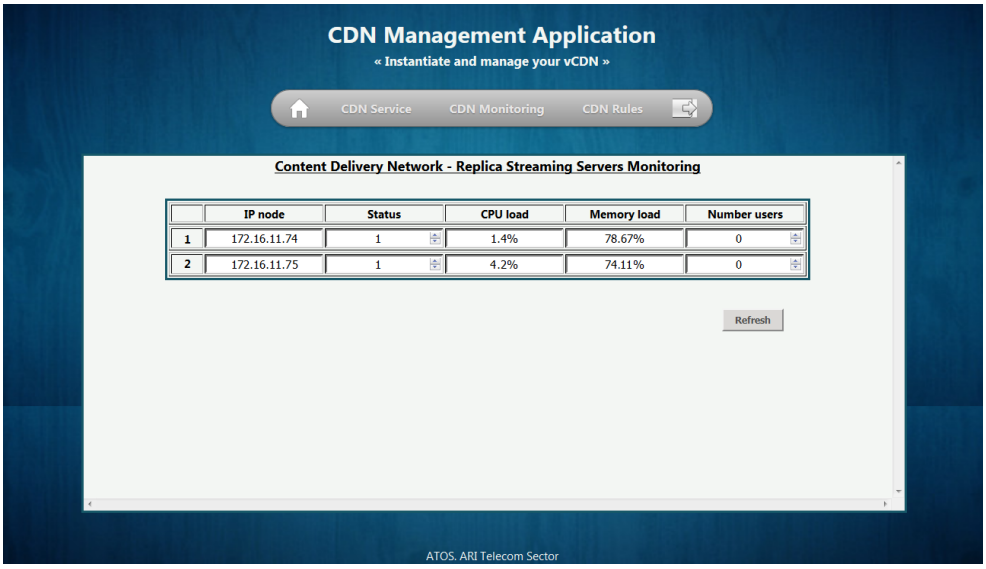


Figure 99: CDN management module (Monitoring information)

12.3 KPIs

Table 33: list of KPIs addressed by CDNMA

Application	***CDN Management Application (CDNMA)***		
List of related project KPIs	***Obj.2: Specify the XCI's northbound (NBI) and southbound (SBI) interfaces*** *(5GPP KPI) Enable the introduction/provisioning of new 5G-Crosshaul services in the order of magnitude of hours* ***Obj.8: Xhaul key concept validation and proof of concept*** *(5GPP KPI) Orchestration of 5G-Crosshaul resources based on traffic load variations*		
Measurement of project KPIs as well as application specific ones	**KPI metrics (unit)**	**Description**	**Way of measurement**
	Provisioning of vCDN infrastructure (minutes)	The time required to deploy a vCDN infrastructure: deployment of VMs and configuration of the monitoring process for the replica servers.	This measurement is taken from experiments in the 5TONIC testbed. Measures are collected by application and proprietary NFVO algorithm logs.
	Service delivery (seconds)	The time that CDNMA requires to provide the video to the user (since MMA new request notification is received until RMA response to path creation is registered)	The CDNMA application will take in consideration the number of connected users, memory load and CPU load per replica server to assign the end user requests to one or another replica server. Measures are collected by the application logs which includes interactions with MMA and RMA
	Bandwidth capacity (Mbps)	Maximum reliable transmission rate that a path can provide	This can be shown sending the same video from two different replica servers, one located nearer the UE than the other one. Another option could be to show the difference between
	Latency (ms)	Network latency is the time taken by data to transmit from one designated point (5G	

		Point of Attachments…) to another (Core, MEC server…)	request a cached video or a not cached video.
	Orchestration of resources (minutes)	Orchestration of resources based on replica servers monitoring information analysis.	This measurement is taken from experiments in the 5TONIC testbed. Measures are collected by application and proprietary NFVO algorithm logs.
List of the State-of-the-Art approach **(used to compare with proposed solutions)**	State of the art approach 1: Content Delivery Network Built on NFV and SDN Current Content Delivery Networks (CDN) can leverage SDN and NFV platform to replicate the content over several replica servers deployed as VNFs instead of hardware appliances [65]. CDNMA covers this main feature and besides takes advantage of 5G-Crosshaul infrastructure where replica servers are deployed on the XPUs connected to the XFEs located across the 5G-Crosshaul network. In addition, CDNMA interacts with the MMA and RMA applications to manage the vCDN service, providing an efficient and flexible way of delivering content to the end users.		

12.4 Validation and Evaluation

12.4.1 Evaluation Environment

The CDNMA application scenario is focused on the media distribution in the 5G-Crosshaul network aiming to reduce the global resource consumption in the network and improving the quality of service (QoS) for the end users. Figure 100 presents an overview of such a scenario.

A complete 5G-Crosshaul network is deployed by a Physical Infrastructure Provider. The CDNMA and XCI will manage the deployment of a virtual infrastructure composed of CDN nodes, acting as origin and replica servers, which are virtualized network functions instantiated within the XPUs. Once the CDN infrastructure is deployed, the end users connected to the network through different points of attachment (RUs, Wi-Fi access points) will request content from a Web page located in the origin server. The users' requests are managed by the CDNMA that, together with the XCI and the MMA

and RMA applications, will determine the most suitable replica server following the rules defined by the CDN operator.

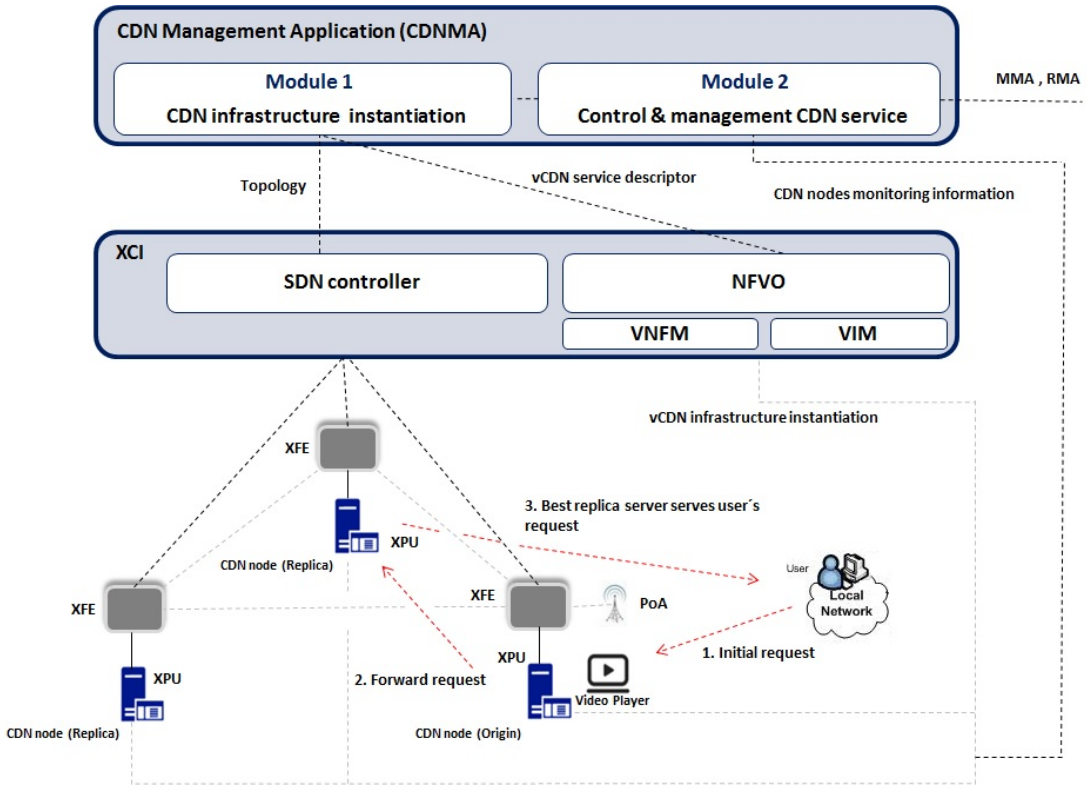


Figure 100: CDNMA scenario description

The validation environment where the CDNMA application will be tested is the 5TONIC-Madrid testbed.

This environment (Figure 101) allows testing the CDNMA application together with the main elements defined in the 5G-Crosshaul architecture. In the application plane the CDNMA will interact with the MMA and RMA applications. The XCI, comprised of an SDN controller and a specific NFV architecture, will interact with the application plane through its NBI. OpenDaylight (ODL) (Beryllium-SR3 release) will be the SDN controller used to manage the OpenFlow elements in the data plane. Openstack, Mitaka version, is used as VIM, and is in charge of controlling the virtual infrastructure and a proprietary development is the NFVO and VNF Manager

One main requirement for the CDN Application is to deploy the origin server and the different replica servers in specific hosts within the 5G-Xhaul network infrastructure.

Initially, TeNOR was chosen as the NFVO and VNFM for the orchestration of the CDN nodes and the VNFs lifecycle.

However, TeNOR was found to be limited and not able to manage the host information sent by the CDNMA in order to be forwarded to the VIMaP (OpenStack) to provision the origin and replica servers in the requested hosts.

Due to this limitation of TeNOR and in order to fulfill the project requirements, Atos has developed a proprietary algorithm which acts as NFVO and VFNM so that the CDN operator can provide the VIMaP with the host where each VNF should be deployed.

The data plane is composed of six switches (XFEs) based on Lagopus software, which forms the data packet plane used to exchange traffic between all the infrastructure components under an Ethernet-based domain, and three servers (XPUs) where it will be deployed the VMs with the CDN nodes. The connection between the user terminal (laptop) and the network entry point will be done through a Wi-Fi connection.

More details about this evaluation environment and the final demo implemented can be found in [37]

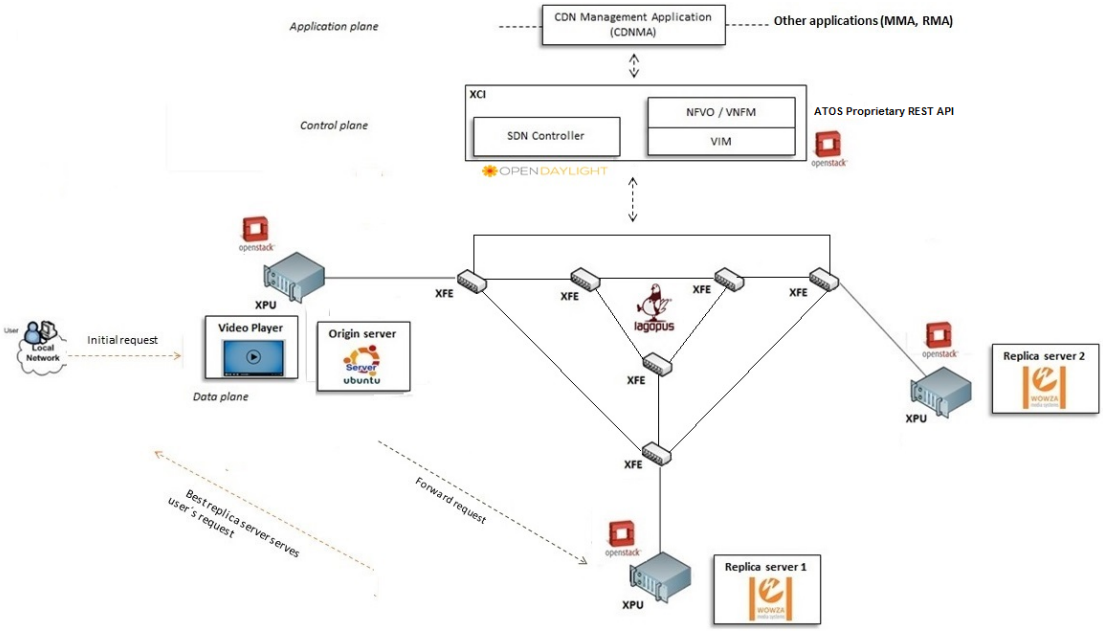


Figure 101: 5TONIC-Madrid: Final CDNMA evaluation environment

12.4.2 Test Cases

The main objective of the planned tests is to prove that the CDNMA application is able to instantiate upon client (CDN operator) request, a vCDN infrastructure based on the CDN operator criteria and the network infrastructure available. It will also be tested that it is able to control and manage the CDN service through the load balancing between the different replica servers. To that end, the main aspects that the tests aim are the following:

- Processing of CDN operator request. The application will show towards the operator a graphic interface running on a REST API for management and operation actions.
- Management of the network topology information obtained from the SDN controller through a REST API.
- Request the vCDN infrastructure instantiation to NFVO. Providing a complete descriptor file with detailed information about the CDN elements and how they have to be configured.
- Management and storage of specific monitoring information obtained from the CDN nodes.
- Management and treatment of the information exchanged with the MMA and RMA applications.
- Management and update of the CDN node assignments based on the monitoring information received and the logic defined by the CDN operator.
- Performance of the Web player application following the CDN node assignments. Video player that serves the requested video from the assigned server.
- For the details on the test cards, see [66, 67].

Table 34: test cases for CDNMA validation

Test Card 1	ATOS_CDNMA_APP_01	Execution Status	Passed
Test Name	vCDN instantiation		
Objectives	Get the network topology available from the SDN controller. Receive the CDN policies from the CDN operator regarding where to deploy the CDN nodes. Provide a complete network service descriptor file for the vCDN infrastructure instantiation to the NFVO.		
Test Card 2	ATOS_CDNMA_APP_02	Execution Status	Passed
Test Name	Management of monitoring information		
Objectives	Get the specific monitoring information from the CDN nodes. Get specific information (location) about the users from the MMA. Store the information in the database.		
Test Card 3	ATOS_CDNMA_APP_03	Execution Status	Passed
Test Name	CDN Management		

Objectives	Analyze the monitoring information. Check the CDN distribution rules from the CDN operator. Provide the CDN node assignments.		
Test Card 4	ATOS_CDNMA_APP_04	Execution Status	Ongoing
Test Name	Video visualization		
Objectives	Request the path computation to the RMA. Serve the video from the CDN node assigned.		

12.4.3 Evaluation Results

The first set of results is the one related with vCDN infrastructure deployment in the 5G-Crossshaul network. Figure 102 shows the time that took a vCDN to be deployed depending on the number of replica servers that are initially set up. In this scenario, four measurements were taken for vCDN with one, two, three and four replica servers each of them.

The measurements reflect the time from two perspectives:

- CDNMA perspective, which covers the whole instantiation process
- Proprietary NFVO perspective which interacts with the VIM (Openstack), which is in charge of the VMs deployment and configuration within the 5G-Crosshaul network.

The results show that a vCDN can be instantiated and ready to provide the CDN service in a magnitude of minutes. Furthermore, the difference, with regards to time, between deploying a vCDN with a specific number of replica servers and a vCDN with more replica nodes is a linear time increase:

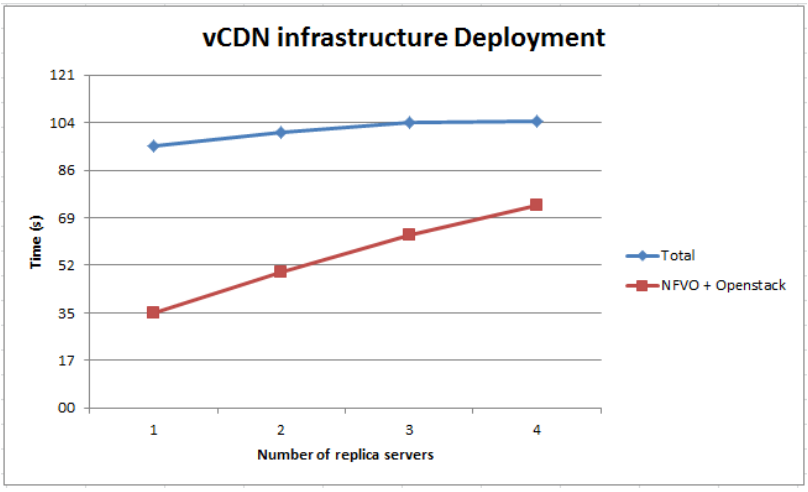


Figure 102: vCDN Infrastructure deployment time

The second set of results are the ones regarding the QoS experienced by the end user (latency, throughput) when he requests a content.

The latter set of results is in the process of being collected and will be documented and evaluated in D5.2.

12.4.4 Summary

The testing performed shows that the CDNMA application leverages from the service offered by a software defined network to deploy a CDN service in an order of magnitude of minutes.

Regarding the QoS, the first results show that the CDNMA application, thanks to the 5G-Crosshaul network and other applications on top of its NBI, like the RMA and MMA, is able to dinamically reconfigure the network path based on network status information and/or (re)-allocate end users and instantiate replica servers in order to improve the quality perceived by the end user, measured through parameters like latency or throughput.

13 TV Broadcasting Application (TVBA)

Based on the principles of IP multicast [68] [69] [70] [71], the TV Broadcast Application (TVBA) aims at providing a solution for TV broadcasting and multicasting services making use of the 5G-Crosshaul architecture, running as an OTT service. This is the case in which service providers share the 5G-Crosshaul network to reach their remote customers so that different tenants operate in an agnostic manner over the underlying infrastructure which they do not require to control directly.

More specifically, a TV broadcasting/multicasting service, typically offered from a premium content of a live source (e.g., a football match), will provide the signal coded at the desired format and bit rate (e.g., image resolution, scan format, etc.) which will be injected into the 5G-Crosshaul network. The TVBA will deploy media transmission, live-video broadcast, over the 5G-Crosshaul infrastructure with focus on minimizing both the cost and the spectrum consumption of the next generation TV while ensuring a fixed QoS and QoE.

The TVBA benefits from XCI services (i.e., SDN Controller and MANO) to deploy media transmission over the 5G-Crosshaul infrastructure. It communicates with the XCI manager (typically through a REST API) and provides the information needed for the service establishment and operation.

As part of its functionality, it provides a set of functions for content media management such as headend injection, routing by means of the selection of the most suitable physical or virtual nodes for very low-latency delivery, and media quality monitoring near the user in terms of QoS and QoE.

In order to select the optimal path from the headend to the users, the TVBA makes use of other 5G-Crosshaul applications, particularly the Resource Management Application (RMA) and the Mobility Management Application (MMA). For the quality monitoring it deploys Quality Probes (TVBAQPs) in the closest XPUs to the users. TVBAQPs are Virtual Network Functions (VNFs) managed by the NFV Orchestrator (NFVO), part of the XCI MANO, that analyze the media received through the same user's multicast group and send the results to the TVBA in terms of QoS (packet loss, media reception) and QoE (freezing frames, color alteration).

13.1 Consolidated design

TVBA algorithms are designed to control the QoS and QoE of the received media near the user. Always managed by the operator through a Web GUI, the media content is injected from the TVBA Headend into the 5G-Crosshaul network and subscribed users are provisioned to receive the service, establishing the path throught the Software-Developed Network by means of the RMA. TVBA Self-Healing system will request a new path to the RMA or new broadcast conditions to the TVBA Headend whenever a QoS or QoE alarm is triggered by the TVBAQP.

Algorithm 7 shows the TVBA algorithms:

Algorithm 1 TVBA Core

```
1: Begin
2:     GET Network Topology from SDN Controller
3:     POST Command to inject media from TVBA Headend into 5G-
   Crosshaul network (data plane)
4:     POST Policies (Criteria defined by the Operator)
5:     for all node do
6:         Check suitability for deployment (geographical coverage, QoS, ...)
7:     end for
8:     if Requested service provisioning for User then
9:         GET User location and mobility from MMA
10:        POST Command to RMA to establish a network path for the User
11:        Write Path to Database to keep track of status and maintain
   broadcast-tree
12:        if Path Established then
13:            GET Collect information about available resources (tenant level)
14:            POST Command to deploy VNFs (TVBAQPs)
15:            POST Command to start TVBAQPs to monitor QoS & QoE
   → [TVBA Monitoring]
16:        else
17:            ERROR Broadcast service deployment not possible
18:        end if
19:    end if
20:    Release Resources
21: End
```

Algorithm 2 TVBA Monitoring

```
1: Begin
2:     while User subscribed do
3:         GET Collect Network Analytics (QoS) from SDN Controller
4:         GET Collect QoS  QoE Analytics from TVBAQP ← [TVBA Core]
5:         if QoS/QoE alarm then
6:             if Path issue then
7:                 POST Command for RMA with constrains to establish a new
   network path for the User
8:             else if Source issue then
9:                 POST Command for TVBA Headend to change broadcast
   conditions
10:            else
11:                WARN Warn Operator
12:            end if
13:        end if
14:    end while
15: End
```

Algorithm 7: TVBA procedures

13.2 Implementation

The TV Broadcast Application (TVBA) is an OTT service whose purpose is to provide TV broadcasting and multicasting services making use of the 5G-Crosshaul architecture and maintaining acceptable QoS and QoE at user's reception. To achieve this objective the TVBA uses the following resources:

- 5G-Crosshaul MANO (XCI) to manage networking, storage and compute resources and to manage the deployment of TVBAQP, VNFs which monitor media quality near the users in terms of QoS and QoE.
  - VIM based on OpenStack.
  - NFVO and VNFM based on OpenBaton.
- 5G-Crosshaul SDN Controller (XCI) based on OpenDaylight.
- 5G-Crosshaul Data Plane (XPFEs and XPUs) to broadcast media and deploy TVBAQPs.
- TVBA Headend: it is the source of media itself and it broadcasts content injecting it to the network as directed by the TVBA.

13.2.1 High-level design

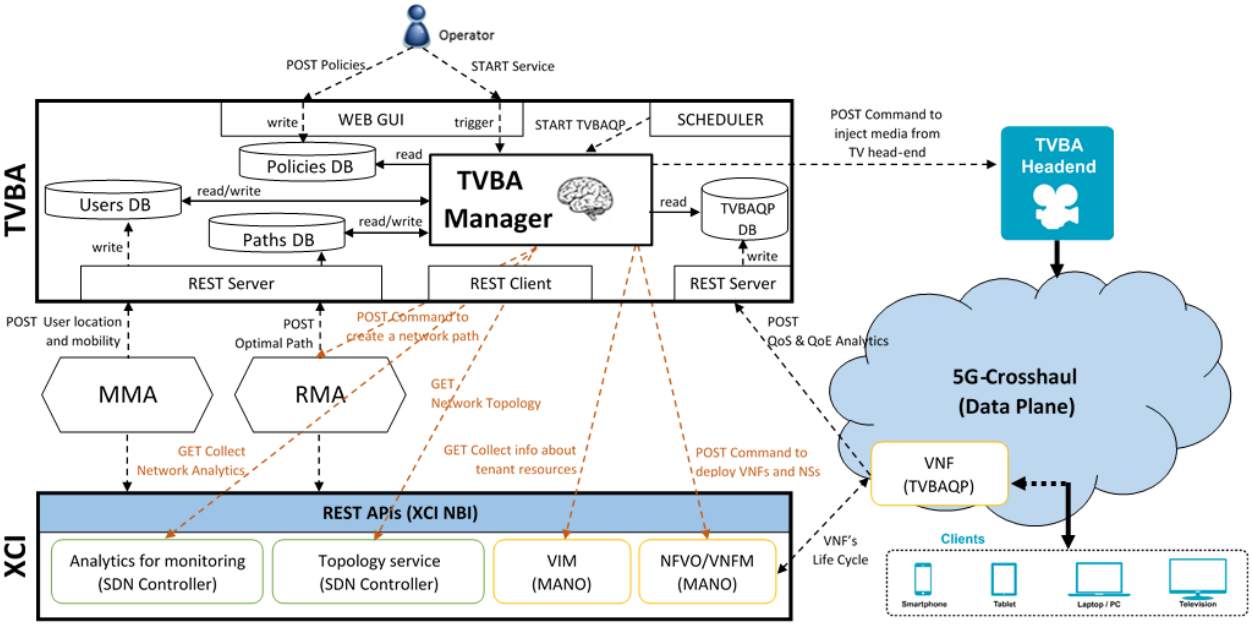


Figure 103: TVBA high-level software design

13.2.2 Implementation Details

In this section we present the TVBA architecture including the description of its main building blocks in Table 35. The application has been designed to provide full control over the Broadcast Network using REST APIs.

Initially the network topology is fetched from the SDN Controller by the TVBA Manager and the Operator interacts with it through a Web GUI (HTML5) where all information needed will be displayed (Figure 104).

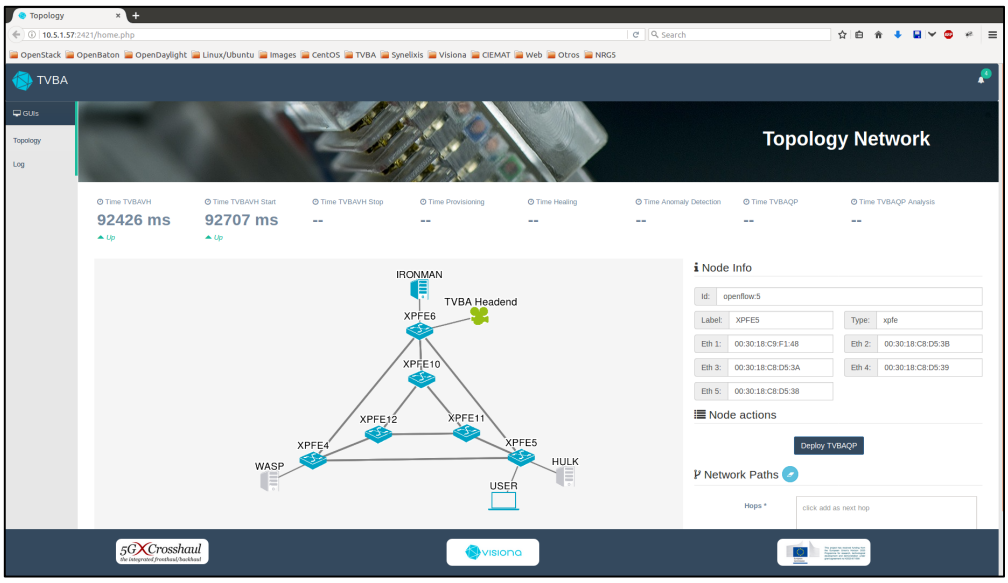


Figure 104: TVBA Web GUI

The TVBA Headend is shown as part of that network topology so both the Operator and the TVBA Manager (for automated requests) are able to interact with it and start/stop the service, as well as, modify broadcasting parameters as content quality, resolution, codec, etc.

When a User is announced by the MMA to the TVBA through a REST API, it can be added to the TVBA network so that the Operator is able to interact with the User and to request the provision of a broadcast service for it. In that case, the TVBA Manager makes a request to the RMA to generate a network path from the TVBA Headend to the User based on the QoS constrains defined by the Operator. This path is generated following an IP multicast structure (broadcast-tree) so that the User is able to receive the content through a multicast group.

Once the service is provisioned the TVBA Manager calls MANO's API to reserve resources (by means of the VIM), launch a TVBA Quality Probe (TVBAQP) and monitor its life-cycle (NFVO). The TVBAQP follows the QoS and QoE policies defined by the Operator and it's deployed as close as possible to the User, at least in a XPU attached to the same XPFE, subscribing to the same multicast group, that is, getting the same content. Further improvements on this line aligned with the RAN (Radio Access Network) structure could include mirroring of the data and network QoS analysis beyond the RAN-Crosshaul split.

The instantiated TVBAQP analyzes the quality of the content received and send the results to the TVBA through a REST API. Based on the quality results from the TVBAQP and on QoS Analysis from Network Alarms provided by the SDN Controller, the TVBA Manager decides if the QoS and QoE received by the User are acceptable. If they are not, the TVBA Manager does either request a new path to the RMA with new

constrains or change the broadcasted quality at the TVBA Headend. Further improvements on this line could provide the possibility to modify quality at TVBAQP level, that is, different users may get different quality of content, getting an optimized QoE based on network resources and User's device.

Instantiation of TVBAQPs (and following analysis and decision-taking by the TVBA Manager) can be scheduled in order to maintain a fixed QoE for the Users or requested on-demand by the Operator to check the QoE.

Table 35: TVBA application components

Component	Description
TVBA Manager	The core of the TVBA. It's the entity in charge of taking decisions (change path, change quality) and communicating with the XCI, other applications and the Operator.
Databases	Where all monitoring and exchange information is stored: QoS, QoE and Broadcasting Policies; Registered Users; Available Paths between the TVBA Headend and the Users; Deployed TVBA Quality Probes (TVBAQPs); and analysis results coming from TVBAQPs.
TVBAQP	Stands for TVBA Quality Probe. It is a VNF in charge of analysing the media received through the same multicast group than the user and sends the results to the TVBA in terms of QoS (packet loss, media reception) and QoE (freezing frames, color alteration).
TVBA Headend	Entity in charge of the injection of the media content to the network. It can be any source of video providing the adecuated API to be connected to the TVBA.
REST Client	Entity in charge of sending REST messages to RESTful entities such as RMA and SDN Controller (OpenDaylight).
REST Server	Entity in charge of providing a RESTful interface to other entities such as MMA, RMA and TVBAQP.
Web GUI	Interface for the operator with handful functions to manage the application and check information and results presented in a user-friendly way.
Scheduler	Entity in charge of starting periodically the already instantiated TVBAQPs in order to monitor the QoS and QoE of the Users.

13.3 KPIs

Table 36: list of KPIs addressed by TVBA

Application	***TV Broadcast Application*** *The TV Broadcast Application (TVBA) aims at providing a solution for TV broadcasting & multicasting services using 5G-Crosshaul architecture, running as an OTT service.*

<table>
<tr><td>List of related project KPIs</td><td colspan="3">Obj.2: Specify the XCI's northbound (NBI) and southbound (SBI) interfaces

(5GPP KPI) Enable the introduction/provisioning of new 5G-Crosshaul services in the order of magnitude of hours

Obj.8: Xhaul key concept validation and proof of concept

(5GPP KPI) Self-healing mechanisms for unexpected 5G-Crosshaul link failures through alternative path routing in mesh topologies.</td></tr>
<tr><td rowspan="3">Measurement of project KPIs as well as application specific ones</td><td>KPI metrics (unit)</td><td>Description</td><td>Way of measurement</td></tr>
<tr><td>Provisioning of services (secs)</td><td>The time required to deploy a TVBA service what includes: deployment of VNF, establishment of a path and first QoS & QoE measures and results of the TVBAQP.</td><td>This measurement is performed in the 5TONIC testbed where the TVBA is integrated with real hardware.

The measurements are done capturing the messages sent between TVBA components when important events happen.</td></tr>
<tr><td>Self-healing</td><td>Capacity to relocate paths and resources to deal with network issues.</td><td>This measurement is performed in the 5TONIC testbed where the TVBA is integrated with real hardware.

An anomaly (such a bandwidth congestion or a link failure) is created to force the self-healing process to be triggered.</td></tr>
<tr><td>List of the State-of-the-Art approach

(used to compare with proposed solutions)</td><td colspan="3">1. State of the art approach 1: SDN (OpenDaylight)
Software Defined Networking (SDN) technology efficiently creates (or re-creates) a broadcast-tree that handles user demand while optimizing network resources. Deployment of network can be performed in a matter of minutes instead of days.

2. State of the art approach 2: NFV (OpenBaton)
Network Function Virtualization (NFV) technology allows dynamical deployment of software to analyse QoE and QoS over 5G-Crosshaul Processing Units (XPUs) in order to determine whether it is necessary to activate self-healing protocols. This deployment is done in servers controlled by OpenStack.</td></tr>
<tr><td></td><td colspan="3">Media Distribution: TV broadcasting & Multicasting
This use case is related to the distribution over 5G networks of media contents, especially video traffic, and TV broadcasting which are expected to be the dominant contributors to the mobile data traffic demand.</td></tr>
</table>

13.4 Validation and Evaluation

13.4.1 Scenarios Descriptions

The TVBA application has being evaluated over a topology based on a mesh of 6 XPFEs and 3 XPUs attached to different XPFEs. Such a scenario guarantee that multicast paths can be established, and QoS or QoE issues can be faced reconfiguring paths or changing quality in the source.

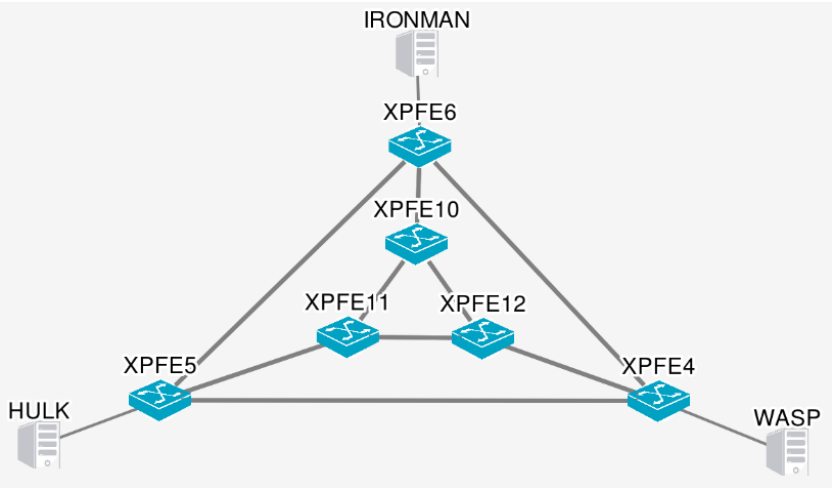


Figure 105: TVBA scenario for tests

As showed in Figure 105, different users could be connected to different XPFEs (or to the same) allowing a multicast path rather than unicast as well as a new paths could be computated in case there is any QoS or QoE problem detected at any of the users.

13.4.2 Evaluation Environment

Although part of the initial development has been done using several Virtual Machines in a local Hypervisor, the final evaluation environment has been moved to 5TONIC Testbed in Madrid. More concretelly the resources used from the testbed are the following:

- 3 XPUs, bare metal servers controlled by OpenStack (VIM)
- 6 XPFEs, switches using Lagopus.
- 3 Virtual Machines controlled by an Hypervisor in a server:
  - NFVO (OpenBaton) for orchestration.
  - SDN Controller (OpenDaylight) to manage the network.
  - TVBA, including REST Server, Web GUI and other TVBA components.

Regarding other components for the TVBA:

- TVBA Quality Probes are deployed as VNFs in the XPUs.

- The TVBA Headend, as source for testing purposes, has been partially virtualized and launched in one of the XPUs as a VNF.
- The RMA has been replaced by a light version where path computation is set manually in advance so optimal path is based on number of hops. This fits perfectly for TVBA testing purposes.
- The MMA has been replaced by a light version where users discovery is not necessary but it still notifies to the TVBA about new users manually subscribed to the network.

13.4.3 Test cases (according to the planned test-cases in IR3.3/4.3)

TVBA test cases were reflected in [3]. The main objectives of the planned tests to be carried out are the following: i) functional validation and experimental assessment of the TVBA SBI interface; ii) functional validation of the interaction with the cloud controller (OpenStack); iii) functional validation of the interaction with the SDN controller (OpenDaylight); iv) functional validation of the TV headend injection; and v) functional validation and experimental assessment of the TVBA-TVBAQP interface which controls the exchange of information between the Quality Probe (testing QoS and QoE) and the TVBA Control Manager (the hearth of the application).

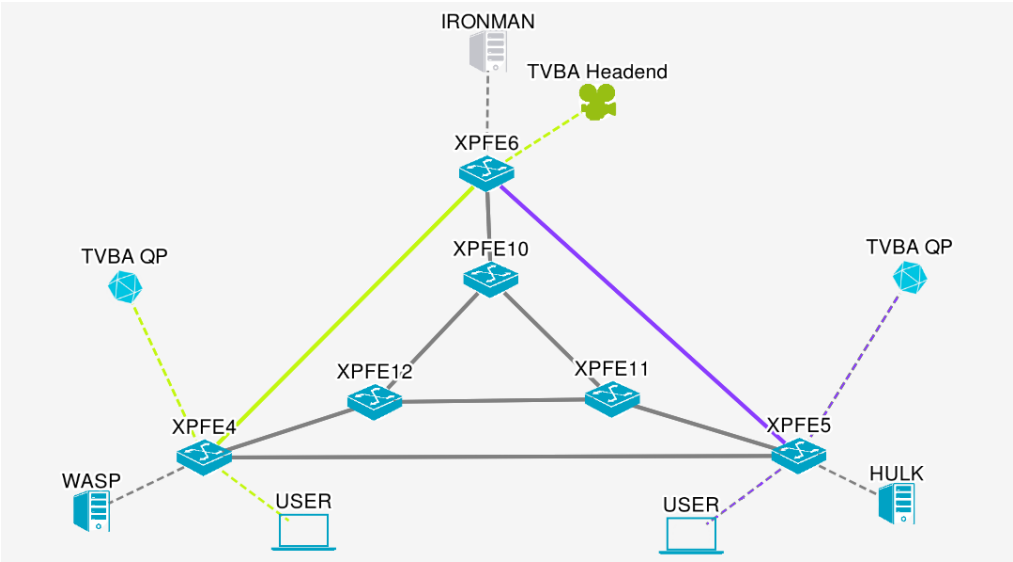


Figure 106: TVBA service provided for 2 users and TVBAQPs deployed

The main tests that are considered are the following, and in line with the services that are provided by the TVBA: i) test for TV Headend injection, ii) tests for service reconfiguration, iii) instantiation of Virtual Machines in remote locations and communication with them, iv) tests for topology detection and, v) test for end-to-end service visualization. For the details on test cards, see [3].

Table 37: tests for TVBA evaluation

Test Card 1	VIS_TVBA_APP_01	Execution Status	Passed

Test Name	Functional test of TV Headend		
Objectives	To test the functionality the TV Headend should provide as video source for the media distribution.		
Test Card 2	VIS_TVBA_APP_02	Execution Status	Passed
Test Name	Video service reconfiguration		
Objectives	To establish new parameters for the video service deployment (as future responsiveness to network status).		
Test Card 3	VIS_TVBA_APP_03	Execution Status	Passed
Test Name	End machine connectivity		
Objectives	To assert that the communication between the machines is working.		
Test Card 4	VIS_TVBA_APP_04	Execution Status	Passed
Test Name	Topology		
Objectives	To assert the topology is well setup.		
Test Card 6	VIS_TVBA_APP_05	Execution Status	Passed
Test Name	End to end service visualization		
Objectives	To assert the video transmission and reception is done end-to-end as expected.		

13.4.4 Evaluation Results

The effectiveness of TVBA to objectly detect quality issues at the User's attachment point placing VNFs to analyze the traffic and the choice of the appropriate self-healing mecanism when a quality issue is detected are evaluated. Specifically, we focus on investigating the contribution of TVBA to the following KPIs to meet the project objectives:

- Enable the introduction/provisioning of new 5G-Crosshaul services in the order of magnitude of hours.
- Self-healing mechanisms for unexpected 5G-Crosshaul link failures through alternative path routing in mesh topologies

To this end we evaluate the following average performance metrics obtained as a result of the experimental tests of the TVBA algorithms:

- Service provisioning time, defined as: $U_{dt} + QP_{dt} + QP_{st} + QP_{at}$. Where:

- U_{dt} is the *User's provisioning time*, that is, the time to establish the path from the TVBA Headend to the users.
- QP_{dt} is the *Quality Probe's deployment time*, that is, the time a TVBAQP (a VNF) needs to be instantiated. Tends to zero if the TVBAQP was already created in the same XPU to monitor other user.
- QP_{st} is the *Quality Probe's starting time*, that is, the time a TVBAQP needes to be started once it has been instantiated.
- QP_{at} is the *Quality Probe's analyzing time*, that is, the time a TVBAQP needs to analyze the media transmission looking for quality issues.

- Self-healing time, defined as: $R_t + QP_{st} + QP_{at} + TVBA_{dt} + S_t + QP_{st} + QP_{at}$. Where
  - R_t is the *Reaction time*, that is, the remaining time from the moment the issue happens to the next scheduled starting time of the Quality Probe.
  - QP_{st} is the *Quality Probe's starting time*, that is, the time a TVBAQP needes to be started once it has been instantiated.
  - QP_{at} is the *Quality Probe's analyzing time*, that is, the time a TVBAQP needs to analyze the media transmission looking for quality issues.
  - $TVBA_{dt}$ is the *TVBA's decision time*, that is, the time the TVBA needs to take a decision about how to solve the quality issue including all quality information collection.
  - S_t is the *Solution time*, that is, the time to apply the solution and it can be:
    - Service provisioning time, already defined.
    - TVBA Headend reconfiguration time, that is, the time the TVBA Headend spends changing its broadcasting parameter.
  - After Solution is applied, a new QP analysis is performed to guarantee the correction of the issue. If the issue is not corrected a new solution decision will be taken and the total time self-healing time will be measured.

Figure 107 shows the average of the previous metrics:

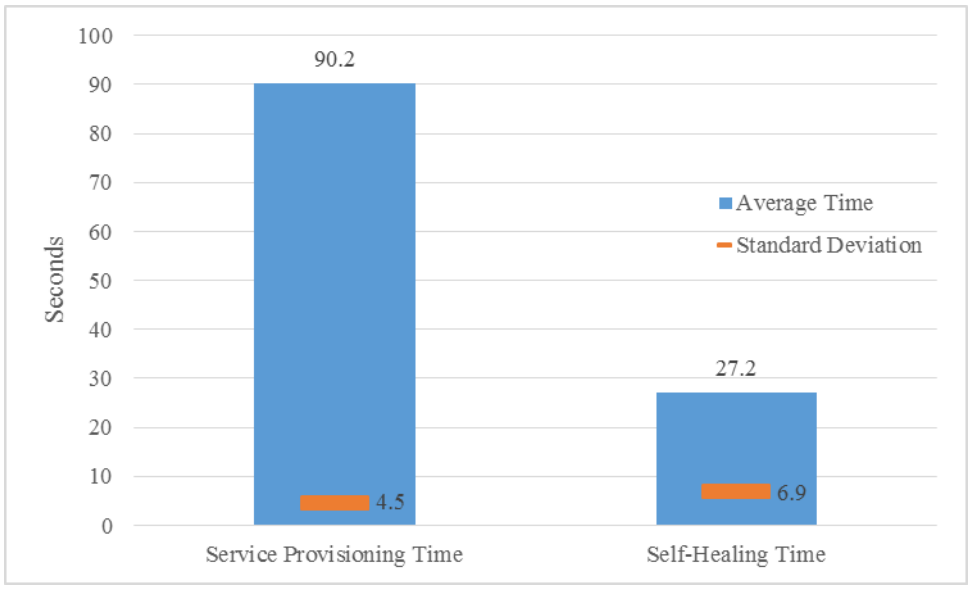


Figure 107: Average of metrics for KPIs

Figure 108 and Figure 109 show an average of each one of the components of the metrics so we can clearly conclude that QP_{dt}, QP_{at} and R_t are the most important ones.

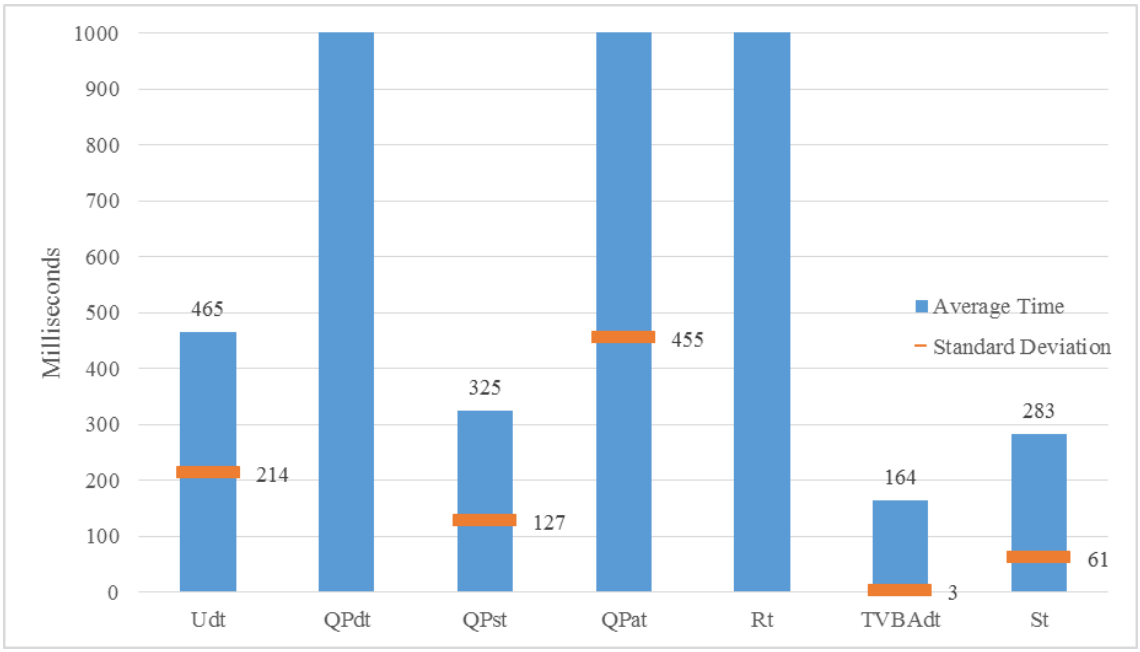


Figure 108: Average of each time involved in the metrics (zoom in of next figure)

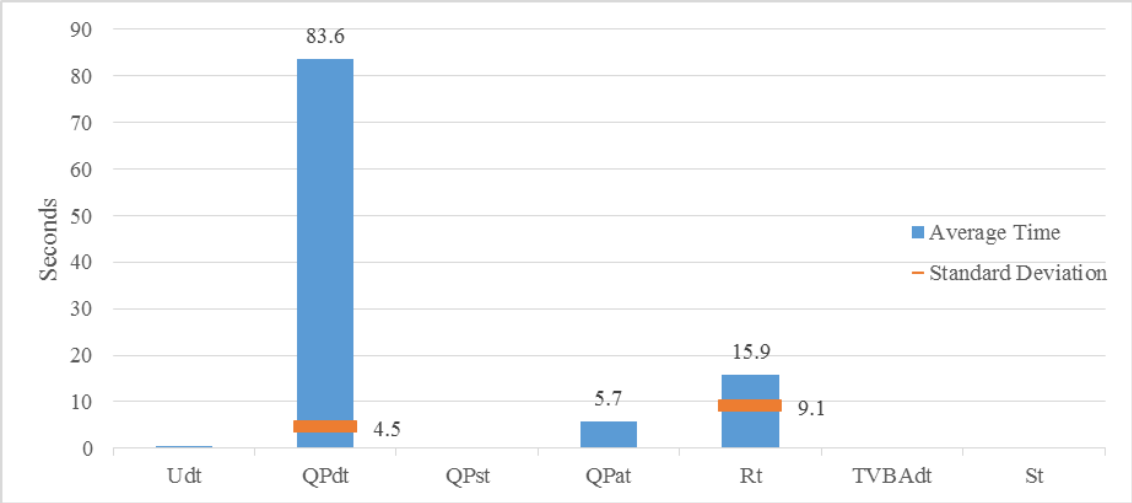


Figure 109: Average of the longest times involved in the metrics (zoom out of previous figure)

Analyzing each of them:

- QP_{dt} depends on the cloud image used as VNF Component for the TVBAQP and on the orchestration software installed by the NFVO in the VNF. A faster-booting image could reduce even more this time.
- QP_{at} depends on the buffering time of the TVBAQP and on the time to analyse each frame of the buffered video. Both times depend in turn on the number of frames to analyze. The more frames the more accurate the analysis but the more time to perform it. For TVBA testing 250 frames are analyzed as an acceptable trade-off, although it can be selected by the Operator at any time. Figure 110 shows the QP_{at} time depending on the number of frames to analyze.

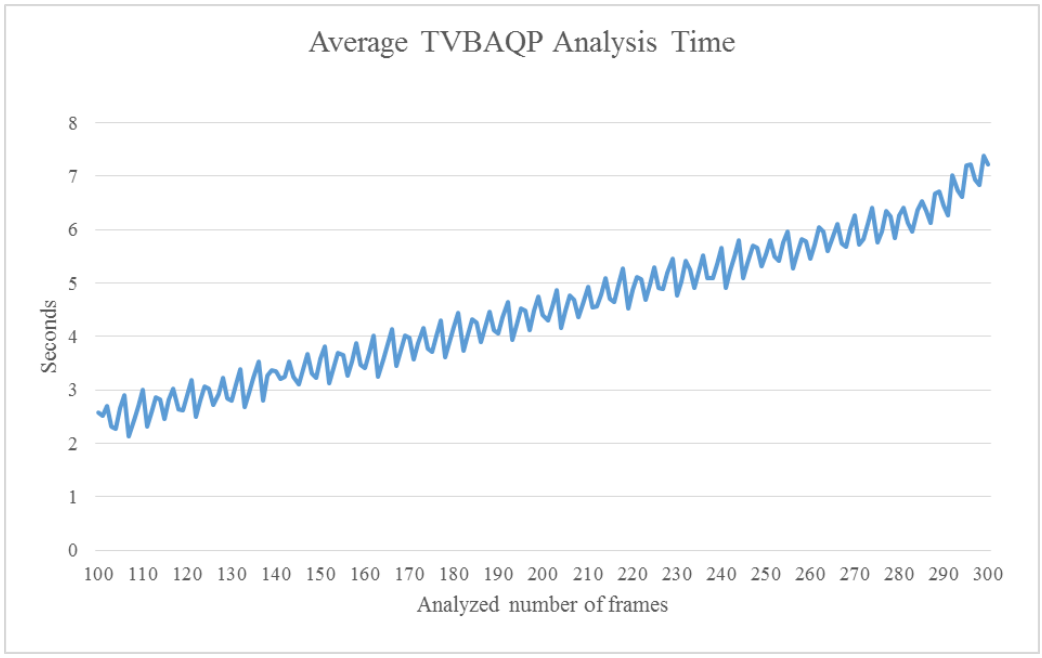


Figure 110: Average time spent by the TVBAQP to analyze frames of the received media

- R_t depends on the TVBAQP starting frequency established in the scheduler. Between TVBAQP starting events the traffic is not analyzed and therefore the

quality issues cannot be detected until the TVBAQP starts again. The time between starting events ($Sleep_t$, sleeping time) is defined in the scheduler by the Operator (for testing porposes 2 minutes are configured). The R_t will be always $QP_{at} < R_t < Sleep_t + QP_{at}$. On the other hand, the more the frequency (the less the sleeping time) the more resources are going to be used by the XPU hosting the TVBAQP. A trade-off could be achieved applying this constrain to the algorithms of Energy-Saving Applications.

13.4.5 Summary

Extensive experiments evidence that a succesful TV Broadcast Application service can be deployed following different optimization criteria and a 5G-Crosshaul software-defined network could intelligently make use of the most effective policy depending on the specific traffic conditions analized under quality-oriented VNFs (TVBAQPs).

The TVBAQP-performed analysis of QoE and QoS received by Users allows the TVBA to heal the system by means of two mecanisms: modifying the service path and reconfiguring the source of media. Both the provisioning of a multicast TV service in the order of magnitude of minutes and the self-healing feature are two of the benefits given by the TVBA working as an OTT service of the 5G-Crosshaul network.

14 Use of developed applications in demostration

The applications developed in WP4 have been evaluated in the four demonstrations in WP5:

- Demo 1: Energy management of Crosshaul
- Demo 2: Media distribution over the Crosshaul services
- Demo 3: Hierarchical multi-domain resource management of the Crosshaul
- Demo 4: Crosshaul fulfilment of the requirements of multiple RAN splits

Each Demo is composed of several individual experiments, as described in D5.2 [67] and reported in Table 38. The experiments where WP4 applications where tested are listed in Table 39.

Table 38: List of Experiments in D5.2

Number	Name
1	Power Consumption Monitoring for XPFE physical nodes
2	Power consumption monitoring for single network paths and tenants
3	Energy-oriented network resource management in RoF domains
4	Energy-oriented network resource management in XPFE domains
5	Energy-oriented virtual infrastructure management
6	Energy-oriented network resource management in XPFE domains for on-demand provisioning of connections dedicated to fronthaul and backhaul traffic
7	EMMA resource management over mmWave mesh
8	Virtual CDN service on the 5G-Crosshaul infrastructure
9	Multicast TV service provisioning
10	Path reconfiguration when QoS degradation
11	Distribution of live content through the vCDN infrastructure on the 5G-Crosshaul infrastructure
12	Assessment of the SDN-based control of the Optical Transport Network (optical domain)
13	Assessment of the SDN-based control and data plane for the mmWave/Wi-Fi mesh domain
14	End-to-end characterization. Network Orchestration across multiple heterogeneous domains: control and data plane characterization
15	Backhaul and Fronthaul services integration through an SDN WS-WDM-PON transport network and XPFEs + Radio-over-Fibre
16	Evaluation of mixed Digital/Analogue Radio-over-Fibre Implementation
17	Integration of XCSE, XPFE, mmWave and CPRI compression data plane solutions
18	Evaluation of integrated packet-based FH/BH combining wired XPFEs and hybrid mmWave/optical wireless links

Table 39: Use of applications in demos and experiments

Application	**Section**	**Demo/Experiment**
RMA for joint path computation/VNF placement	2	Demo 2: Experiments 8, 9, and 11
RMA for joint Routing C-RAN functional split	3	Demo 3: Experiment 14
EMMA for XPFE/XPU	4	Demo 1: Experiments 1, 2, 4, 5 and 6
EMMA for mmWave Mesh Networks	5	Demo 1: Experiments 7
EMMA for High Speed Train	6	Demo 1: Experiment 3
EMMA for Multitier Networks	7	None
VIMaP	8	None
MMA for Traffic Offloading	9	Demo2: Experiments 8, 9, 10 and 11
MMA for High Speed Train	10	None
MTA for High-Speed Train	11	None
CDNMA	12	Demo2: Experiment 8
TVBA	13	Demo 2: Experiments 9, 10 and 11

15 Requirements on XCI NBI, other Apps, and neighbouring interfaces

15.1 Architecture View

In 5G-Crosshaul the scope of operation of the XCI is limited to (physical/virtual networking/storage/computing) resources within the 5G-Crosshaul transport domain. However, given that a proper optimization of the data plane elements may require knowledge of the configuration and/or other information from the Core network and/or the Radio Access Network (RAN) domains, our system design, as shown in Figure 111, contemplates a Westbound interface (WBI) to communicate with the 5G Core MANO and an Eastbound interface (EBI) to interact with the 5G Access MANO [72].

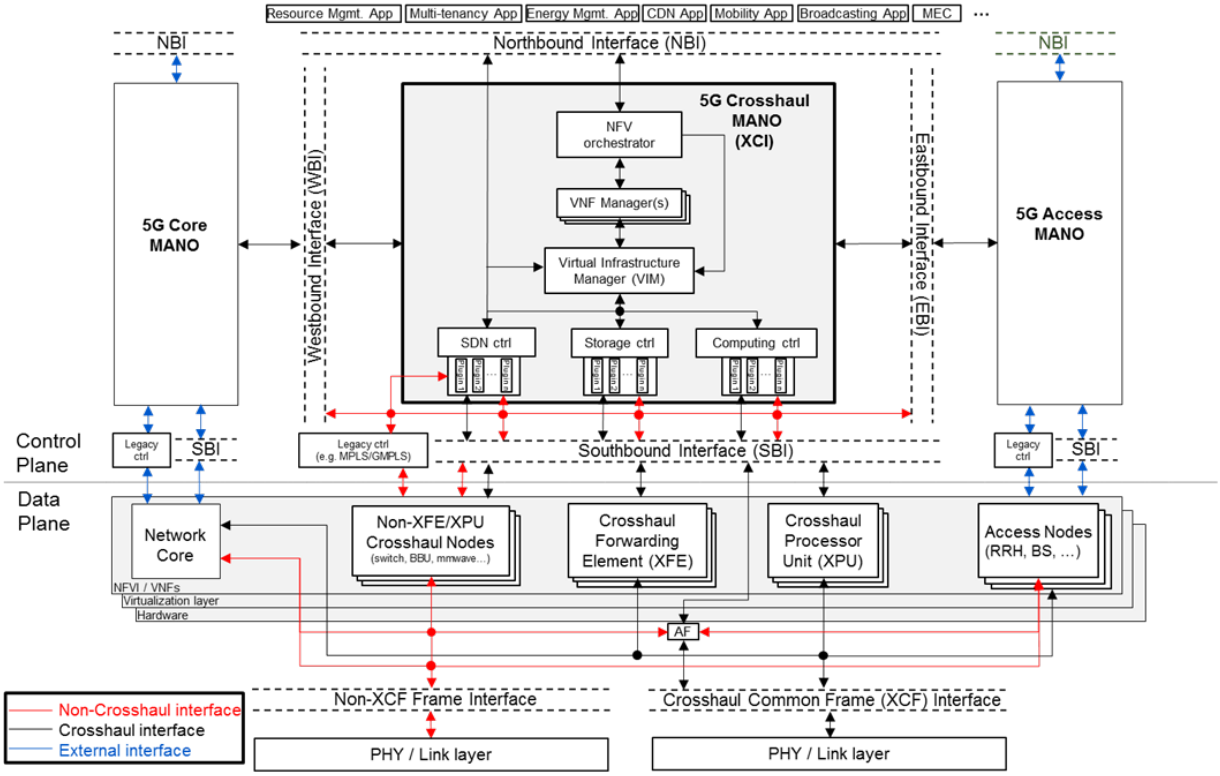


Figure 111: 5G-Crosshaul Architecture

In both 5G Core and Access MANO cases, different architectural approaches could be preferred. Assuming a same hierarchy level relationship between the 5G MANO systems for 5G-Crosshaul, core and access, the WBI and EBI interfaces are used to transfer a subset of monitoring information across domains enabling a selected subset of

the management and orchestration operations (abstracted level of operations and information available), peer-to-peer structure.

In the case of the 5G-Crosshaul MANO system being part of a hierarchical 5G MANO system spanning across 5G-Crosshaul and/or core and access, then the NBI interface can be used and detailed monitoring information and low-level management and orchestration operations are enabled, thanks to such a hierarchical structure.

In the following, we review both types of relationship: hierarchical and peer to peer relationship.

15.2 Hierarchical Structure

A global orchestration engine controls the MANO of each domain (RAN, transport, and Core) via northbound-southbound interfaces. Projects like 5G NORMA [73] define NBI/SBI to communicate with other network domains.

The objective of the 5G NORMA mobile network architecture [74] is to allow for integrating different technologies and enabling different use cases. Due to the partly conflicting requirements, it is necessary to use the right functionality at the right place and time within the network. In order to provide this flexibility, the network function virtualization (NFV) paradigm is adopted in the mobile access and core network domain, enabling mobile network functionality to be decomposed into smaller function blocks, which are flexibly instantiated.

The 5G NORMA functional control and data layer incorporates the novel concept of software-defined mobile network control (SDMC). The interfaces of those novel centralized SDMC-enabled control functions run as applications on top of the SDM coordinator (SDM-X) or SDM controller (SDM-C). The interfaces enable the SDMC apps to control the “legacy” distributed control functions as well as the distributed data layer functions.

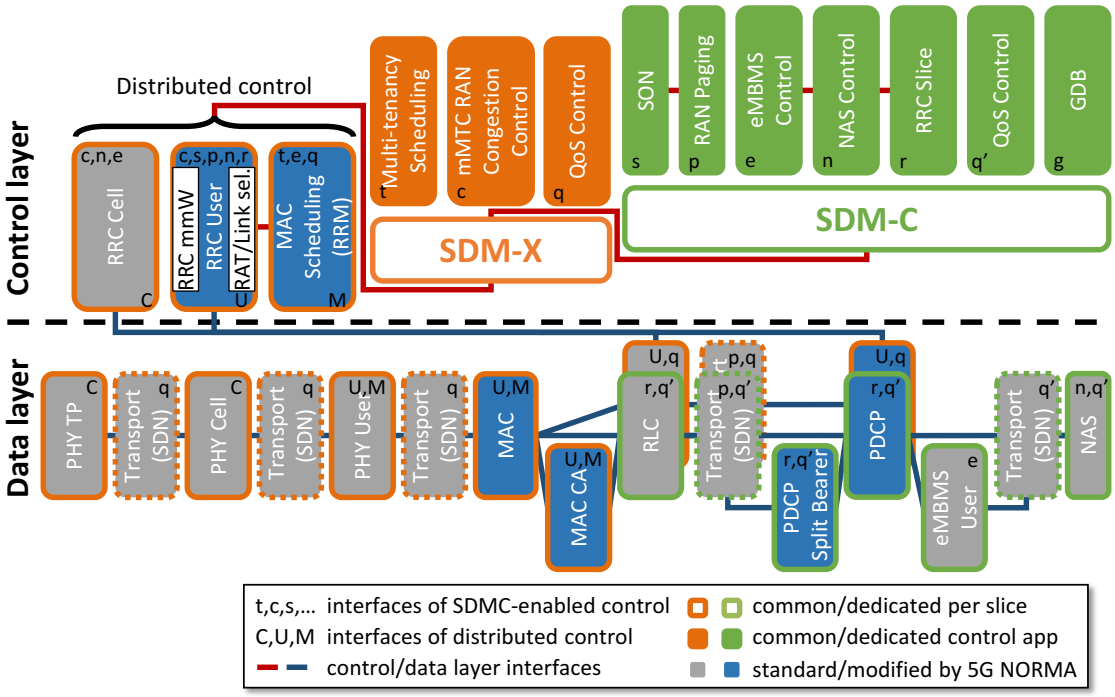


Figure 112: 5G NORMA control and data layer functional architecture

The control/data-layer architecture is depicted in Figure 112, here for the case of RAN slicing i.e., with a common MAC layer. Functions are classified whether they belong to the control or data layer. The control layer functions are further classified into i) distributed, ii) common and iii) dedicated control. Distributed control functions are implemented as VNFs throughout the network, while common and dedicated control functions employ the SDMC concept and run as applications on top of SDM-X and SDM-C, respectively.

The SDM-C and SDM-X configure the 5G network architecture including NFs and SDN transport elements via their southbound interface (SBI). An SBI provides an abstraction of the NF to the SDM-C/X, enabling direct representation of the NF behavior and requirements. The SDMC applications presented below shall require a specific set of information to operate. The SDM-C/X extracts such information from the distributed data and control layer NF via the SBI.

Figure 113 [75] depicts the northbound and southbound interfaces offered by SDM-C.

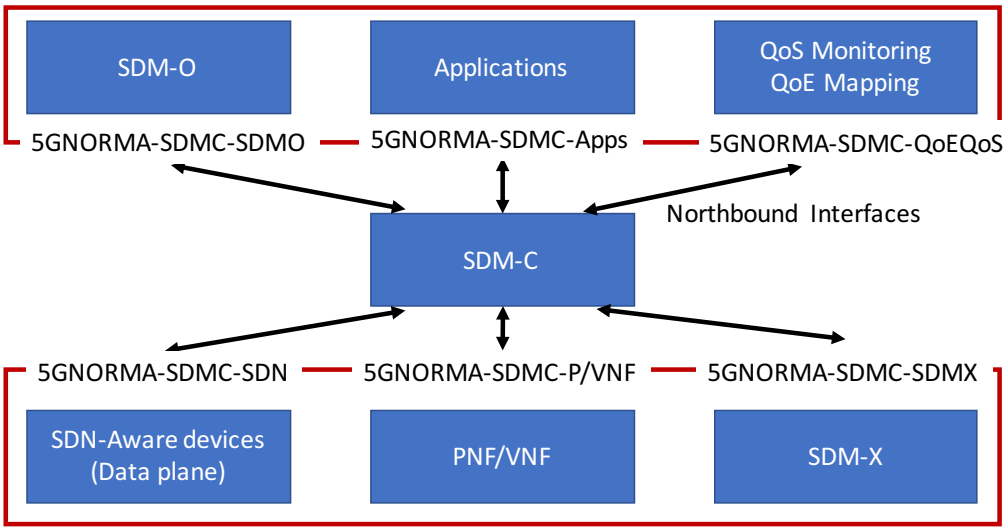


Figure 113: SDM-C interfaces [75]

15.2.1 RAN Slicing Options

Three options of RAN slicing are considered, and are illustrated in Figure 114.

1) The first option (Option 1) shows two network slices where each carries two different services. Each slice may be operated by a different mobile network operator (MNO). Furthermore, for each slice an individual RAN protocol stack is implemented down to the upper part of the physical layer (c.f. PHY-User and PHY-Cell). Only the lower part of the physical layer (c.f. PHY-TP) is shared across slices. The multiplexed access to PHY-TP is coordinated by the SDM-X which makes use of flexible and efficient radio resource management, such as in-resource and user-centric control, where different numerologies are supported within the same spectrum. One could think of Option 1 as implementing all user-specific functions such as forward error correction encoding, layer mapping and precoding in an individual fashion, while TP-specific functionality such as transmission of synchronization and cell-specific reference signals are shared.

2) Option 2 depicts again two network slices from two operators. Compared to the previous example, each slice uses an individual implementation of service-specific functionality such as PDCP, RLC, and slice-specific RRC. In addition, the tenant may implement a customized QoS scheduling to perform pre-scheduling. The access to the MAC layer is then controlled by the SDM-X where resource fairness across tenants and QoS guarantees corresponding to individual SLAs must be met. Furthermore, resource isolation must be provided to alleviate side-effects.

3) Option 3 illustrates the case of two operators using the same RAN as shared resource, i.e., the SDM-X is the interface between CN and RAN. In this

example, no customisation of radio resource management beyond SDM-X parameters and configuration would be possible.

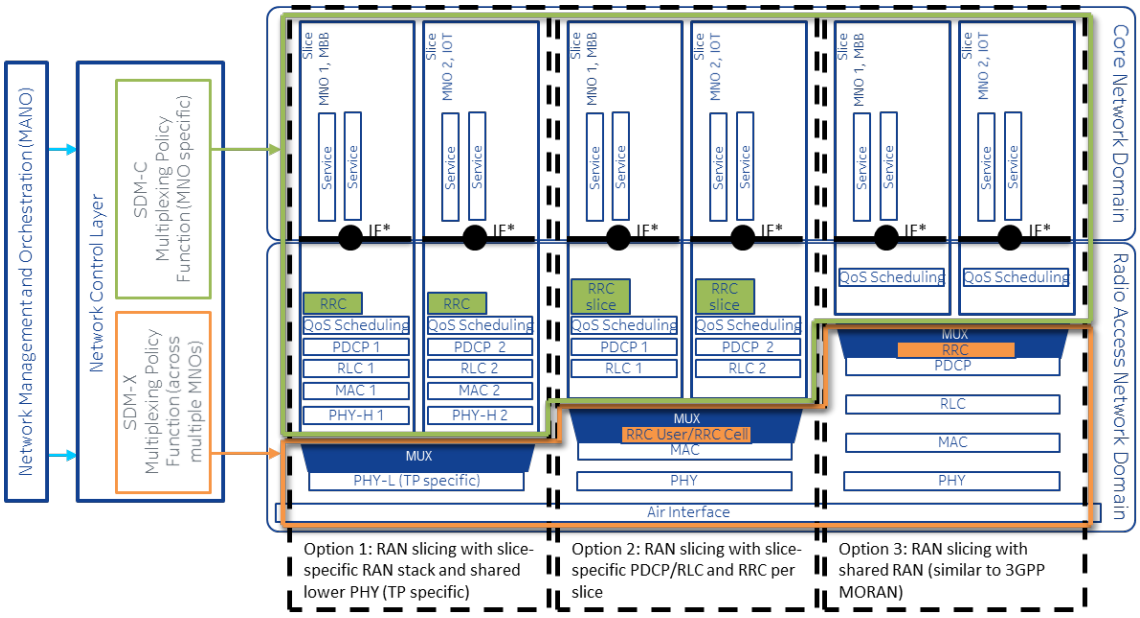


Figure 114: General architecture option including NW slicing and multi-connectivity

15.2.2 Integration of 5G-Crosshaul and 5G NORMA Architectures

5G NORMA and 5G-Crosshaul cover conjunctly the complete design of the operator network. Both network architectures design will be built on the network function virtualization and software defined networking paradigms and will support infrastructure sharing and multi-tenancy. While 5G NORMA is focused on the core and radio access network domains, 5G-Crosshaul controls and manages the transport domain, providing both architectures a complementary role and a complete design of the whole network, as shown in Figure 115.

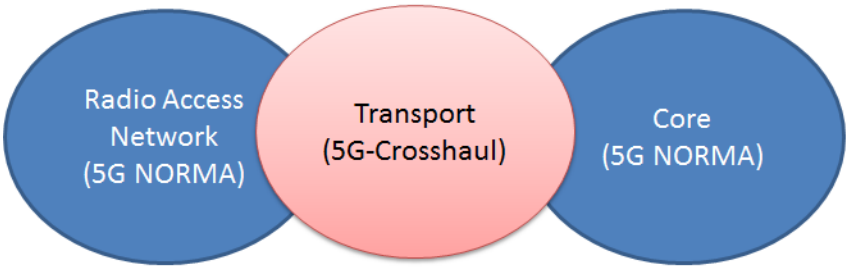


Figure 115: Complementary project roles

15.3 Peer-to-peer structure: 5G-Exchange as enabler of multi-domain Slicing and Network Sharing

In this type of relationshiop, an orchestrator of each domain, based on an evolution of the MANO architecture, can interact and exchange information via eastbound-

westbound interfaces. Projects like 5G-Exchange [76] define EBI/WBI to communicate with other administrative network domains.

5G-Exchange or 5GEx proposes an ecosystem for the trading of resources (with the slice as extreme case) in a multi-domain environment, as described in [77]. The initial high-level architecture framework of 5GEx, shown in Figure 116, identifies the main functional components and the interworking interfaces involved in multi-domain orchestration, where different providers participate each of them representing a distinct administrative domain.

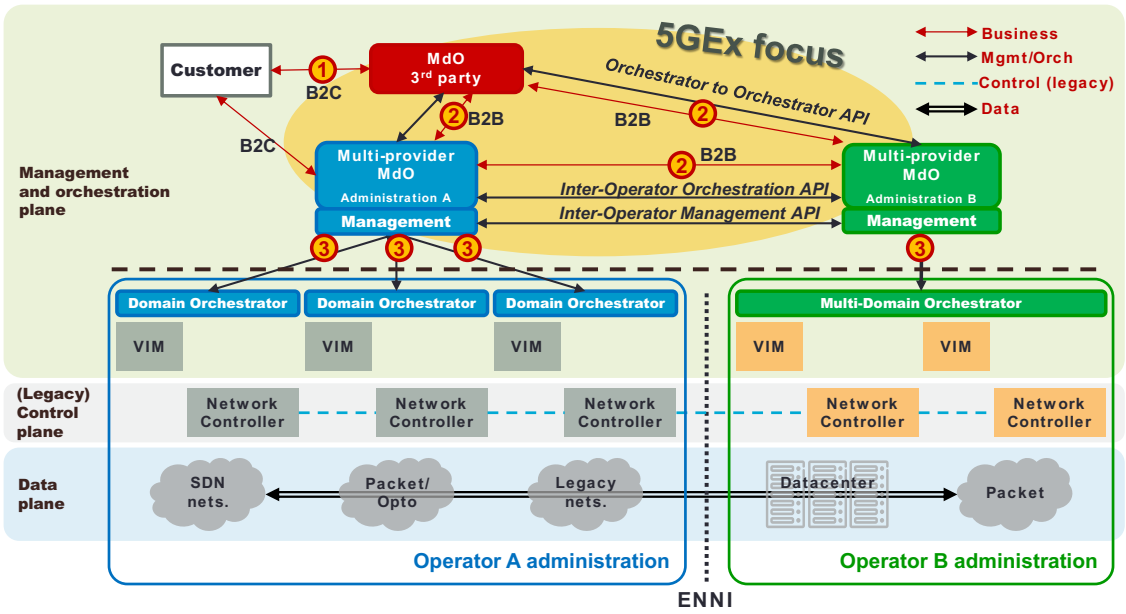


Figure 116: 5GEx reference architectural framework

The bottom part of the figure shows different Resource Domains, hosting the actual resources. The middle shows Domain Orchestrators, responsible of performing Virtualization Service Orchestration and/or Resource Orchestration exploiting the abstractions exposed by the lower Resource Domains.

The key 5GEx component is the Multi-domain Orchestrator (MdO), shown at the top of the figure. The MdO handles the orchestration of resources and services from different providers, coordinating resource and/or service orchestration at multi-domain or multi-operator level, orchestrating resources and/or services using Domain Orchestrators belonging to multiple administrative domains.

15.3.1 5GEx interfaces and APIs

As shown in Figure 117, there are three main interworking interfaces and APIs identified in the 5GEx architecture framework. The MdO exposes service specification APIs (Customer-to-Business, C2B) that allow business customers to specify their requirements for a service on interface I1. The MdO interacts with other MdOs via

interface I2 (Business-to-Business, B2B) to request and orchestrate resources and services across administrative domains. Finally, the MdO interacts with Domain Orchestrators via interface I3 APIs to orchestrate resources and services within the same administrative domains.

Different steps are needed for the service provision in a multi-domain environment. The following ones can be identified as basic stages in the service provision: *(i)* discovery; *(ii)* request; *(iii)* fulfillment; and *(iv)* assurance. Then, the aforementioned interfaces should incorporate capabilities for each of these steps. This deals to consider different implementations for each of such stages.From the perspective of functional capabilities of such interfaces, the functional split considered on each of them is related to service management (-S functionality), VNF lifecycle management (-F), catalogues (-C), resource topology (-RT), resource control (-RC) and monitoring (-Mon). The identification and specification of these interfaces is currently being defined, and it will be fully described in 5GEx deliverable 2.2, due initially for September 2017.

The association of the stages with the described functional split is as follows:

- Discovery phase will be accomplished by I_x-C and I_x-RT interfaces
- Service request phase will be accomplished by I_x-S interface
- Fulfillment phase will be accomplished by I_x-F and I_x-RC interfaces
- Assurance phase will be accomplished by I_x-Mon interface (even fulfillment actions through I_x-F and I_x-RC interfaces are expected as well during this phase as result of the data collected for monitoring and service assurance).

15.3.2 5GEx functional architecture

The 5GEx framework reference architecture has been further developed, defining the different components and interfaces into the functional model shown in Figure 117. This architecture extends the ETSI MANO NFV management and orchestration framework, in order to implement Network Service and Resource orchestration across multiple administrative domains, which may belong to different infrastructure operators or service providers.

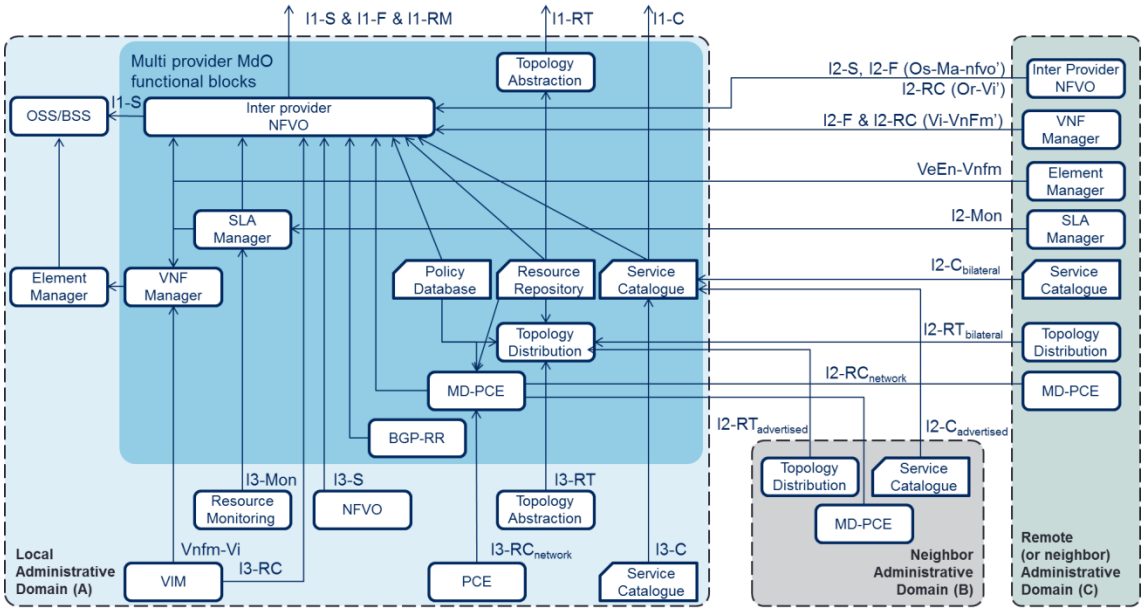


Figure 117: Functional model of multi domain orchestration

Figure 117 highlights three different administrative domains (A, B and C) involved in the Multi-Domain service/resource orchestration process. All the providers in 5GEx are considered to contain the same components and modules (the Operator-Operator relationships are symmetrical in 5GEx), although in Figure 117 the complete view is only shown for the provider on the left for illustration purposes, just showing exemplary consumer-provider roles with arrows from consumer to provider functional blocks. In the figure, Operator Domain A (left-hand) consumes virtualization services of Operator B (transit domain, in the middle) and Operator C (right-hand).

For multi provider Network Service orchestration, the Multi-domain Orchestrator (MdO) offers Network Services by exposing an OSS/BSS-NFVO interface to other Multi-domain Orchestrators from other providers. For multi-provider resource orchestration, the MdO presents a VIM-like view and exposes an extended NFVO-VIM interface to other Multi-domain Orchestrators. The Multi Provider MdO exposes a northbound interface (I1-S) through which an MdO customer (e.g., a vertical industry) sends the initial request for services. It handles command and control functions to instantiate network services. Such functions can include the request for the instantiation and interconnection of Network Functions (NFs). Interface I2-S is meant to perform similar operations between MdOs of different administrative domains.

Interfaces I3-R and I2-R are used to keep an updated global view of the underlying infrastructure topology exposed by domain orchestrators.

The service catalogue exposes available services to customers on interface I1-C and to other MdO service operators on interface I2-C.

Finally, resource orchestration related interfaces are broken up to I2-RC, I2-RT, I2-Mon to reflect resource control, resource topology and resource monitoring respectively.

Furthermore, the notation introduced before is generalized and also used for interface I3 and I1.

15.3.3 Integration of 5G-Crosshaul and 5G-Exchange Architectures

The affordability of sharing 5G-Crosshaul infrastructurespertaining to different administrative domains extending multi-domain concepts from 5GEx has been analysed in [78] and [79]. After the review of both 5G-Crosshaul and 5GEx architectures, it becomes clear that functional adaptation is feasible for allowing the trading of 5G-Crosshaul slices through 5G-Exchange. However there are yet some gaps that would require from certain extension in 5G-Crosshaul for full compliance with a 5GEx ecosystem. This section summarizes both aspects as follows.

15.3.3.1 *Integration in 5GEx of existing functional blocks from 5G-Crosshaul architecture*

Statistics and monitoring of Crosshaul resources. The current 5G-Crosshaul architecture supports the collection of both IT and network statistics, as well as analytic reports elaborating on top of the previous mentioned monitoring information. All of this could be reported as part of the 5GEx I2-Mon interface, providing operational information to other administrative domains requesting Crosshaul services.

Topology and Inventory. The topology information is critical in a multi-domain environment in order to make the right decisions for placement of functions and connectivity. 5G-Crosshaul supports both network and IT topology and inventory reporting, and thus enables the dissemination of this information outside the Crosshaul domain borders. The population of this topology and inventory information can ride on top of 5GEx I2-RT interface, for feeding the Resource and Topology functional blocks of the MdOs of the other provider domains in the Exchange.

Provisioning and Control of resources. The XCI in 5G-Crosshaul facilitates the control of the networking resources for adapt the underlying forwarding elements to the needs of the flows to be transported in the Crosshaul area. This capability can be easily integrated in 5GEx by mapping it to the I2-RC interface.

VNF management and orchestration. 5G-Crosshaul permits to accomplish the full management of the VNF lifecycle via the XCI. The APIs offered by 5G-Crosshaul for this can be homologated to the I2-F interface in 5GEx.

With the integration in a multi-domain environment, the 5G-Crosshaul XCI and the applications on top of it (as defined nowadays) become the 5GEx multi-domain orchestrator. Thanks to the recursiveness properties of XCI, also a dedicated XCI could be devoted to multi-domain aspects interacting as a client with a XCI instance below focused on the Crosshaul domain.

Interestingly, the *VIMaP* functional block in 5G-Crosshaul provides additional capabilities for planning as an extension to the usual VIM functionality. These planning capabilities can be quite useful on assisting the decisions for placement and connectivity in certain services, as the VNFaaS proposition in 5GEx. In this sense, the I2-F interface from 5GEx could be augmented to support the interaction with the *VIMaP* module in 5G-Crosshaul in this direction.

15.3.3.2 *Proposition of additional functional blocks in 5G-Crosshaul for full compliance with 5GEx architecture*

There are instead some other functions not yet fully available in 5G-Crosshaul. The more notorious capabilities are the ones related to business support. Here there is a brief summarization of the findings:

Business support. Specially, the population of the services supported in 5G-Crosshaul in terms of catalog of services is not yet defined. This capability is necessary for advertising the capabilities of each Crosshaul environment in an area in terms of networking and computing resources, as well as some value added services that could complement the offer.

In order to complement the 5G-crosshaul architecture, the proposal here is to define a new functional module on top of XCI in charge of disseminating to other domains the Crosshaul capabilities supported in such domain. This new block, the *5G-Crosshaul Service Catalogue* would be placed at the same level as the other applications defined in 5G-Crosshaul (e.g., Resource Management, Energy Management, etc). In addition, this block is required to support 5GEx I2-C interface for integration on 5GEx ecosystem.

Service specification and request. In a multi-domain environment such as 5GEx it is necessary to have a common understanding on the services offered by each of the participants in the Exchange. To do that, the same semantincs and abstractions have to be handled by the different administrative domains in order to ensure consistency. Such abstractions at technical level imply the utilization of common information and data models for the resources to be configured and used. In the case of integrating 5G-Crosshaul in a 5GEx environment, the former has to support the request of services through 5GEx I2-S interface.

15.4 Application Requirements

Table 40 summarizes the requirements collected from different applications for the neightbouring domains, namly RAN and mobile core network domains.

Table 40: Application requirements for the RAN and/or Core Netowrk Domains

Applications	From RAN	To RAN	From Mobile Core	To Mobile Core
RMA on joint Routing and C-RAN Functional Splits	• Selection of RAN functional split per RRH/BBU (e.g. from 5G-NORMA); • User and/or BS load; • Location of RRH and XPUs to host vBBUs (for clustering).	Suggestion for the selection of RAN functional split per RRH/BBU. (as the results of the RMA)	None	None
RMA on joint Path Computation and Virtual Network Function Placement	None	None	None	None
EMMA for XPFE/XPUs	None	None	None	None
EMMA for mmWave	User /SmallCell load and associated cell	User traffic handling by smallcell or overlaying macro	None	None
EMMA for High Speed Train Scenario	Context information (location of the BS along the rail track) indicating the location of the train	None	None	None
EMMA for multi-tier networks with energy harvesting capabilities	• BS load • Harvested energy • Battery status • Q-learning features (internal parameter of the local agent running Q-learning algorithm) The BS load is a standard measurements already adopted in the	The SBS to be influenced have to receive their correspondent H parameter. i.e., through dedidated messages of the SDN controller.	None	None

	X2 interface, comminly used for ICIC. The energy related parameters can be obtained directly by the harvesting system, that commonly is equipped with sensors devoted to this. Finally, the Q-learning parameters can be obtained by the local agents opportunelty formatting them.			
MMA for Traffic Offloading	User location (non-3GPP networks). IP and MAC addresses of the Point of Connection (PoC) to the network and IP/MAC addresses of the user.	None	User location (3GPP networks)	None
MMA for High-Speed Train Scenario	Context RAN information including HO decision and path switch request such as New eNB UE X2AP ID, Old eNB UE X2AP and eNB to UE S1AP ID S1 Application Protocol (S1AP). (3GPP TS 36.413 version 12.3.0 Release 12) X2 Application Protocol (X2AP) (3GPP TS 36.423 version 12.3.0 Release 12)	None	None	None
VIMaP	None	None	None	None
CDNMA	User location (also can be obtained through MMA)	None	None	None

TVBA	User location (also can be obtained through MMA)	None	Location of the Video Headend.	None
MTA for High-Speed Train Scenario	None	None	None	None

15.5 Summary

For the interaction with neighboring network domains (i.e., RAN and mobile core) we foresee both hierarchical and peer-to-peer structures. This is in line with the architectures of 5G-NORMA and 5G-Exchange projects. In terms of interfaces, the 5G-Ex project has been working on the design of the interfaces and APIs towards other networks or administrative domains. The interfaces can be not only used for EBI/WBI, but can also be applied to the NBI/SBI of the 5G-Crosshaul MANO to interact with RAN and core network domain controllers. In this way, both hierarchical and peer-to-peer structures can be implemented. The detailed 5G-Ex design and the modelling of these interfaces can be found in the 5G-Ex deliverables (http://www.5gex.eu/).

16 Conclusions

This document provides a detailed, hands-on description of the final design of every 5G-Crosshaul application defined in the scope of WP4, detailing the design, the internal algorithms, the implementation procedure, the KPIs that it addresses, and the methodology that was followed to evaluate and validate it. Validation and evaluation were performed either by simulation, emulation or implementation. For evaluation of the application, benchmark methods were selected for comparing against the proposed solutions for the evaluation of the selected KPIs. The presented results not only validate the designed application functionalities but also demonstrate the performance gain of the applications compared to benchmark approaches. In addition to the results reported in this document, some of the implementation and evaluation results based on the integrated test-bed are reported in WP5 deliverables. Additionally, application requirements related to RAN and mobile core were spelled out, in an effort to clarify interactions with neighboring domains, highlighting the relationship of 5G-Crosshaul with other PPP Phase I projects.

17 APPENDIX

17.1 Algorithms for optimal VNFs placement and network paths allocation in EMMA

17.1.1 System model

Our model is based on two graphs, a logical one and a physical one, exemplified in Figure 118 and Figure 119. Figure 120 summarizes our constraints and objective function, while Figure 121 recaps the notation we use.

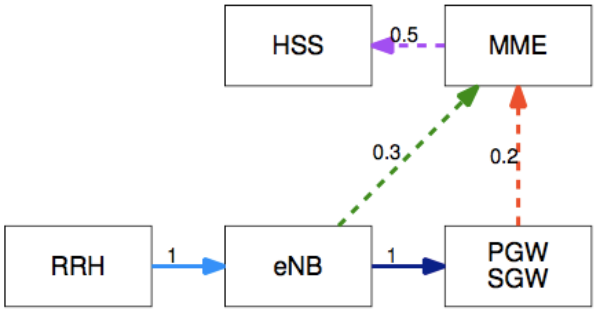


Figure 118: Simplified logical graph for vEPC. Solid lines correspond to user traffic, dashed lines to signaling traffic.

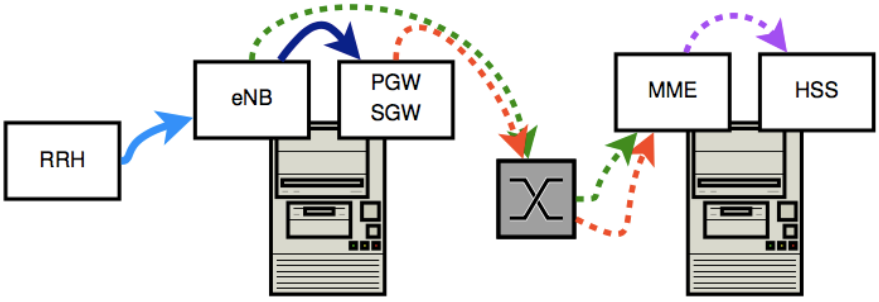


Figure 119: Example implementation of the logical graph above over a physical network. Each line corresponds to a physical flow, i.e., to a τ-variable; their color and style match the logical flows in the logical graph.

$$l(e,v_2,v_3)=\sum_{v_1\in\mathcal{V}}\ell(e,v_1,v_2)\chi(v_1,v_2,v_3)+l(e,v_2)\chi(e,v_2,v_3),\quad \forall e\in\mathcal{E},v_2,v_3\in\mathcal{V}. \tag{1}$$

$$\sum_{(i,c)\in\mathcal{L}}\tau_{i,c}(e,v_1,v_2)=t_c(e,v_1,v_2)+p_c(e,v_1,v_2),\quad \forall c\in\mathcal{C},e\in\mathcal{E},v_1,v_2\in\mathcal{V}. \tag{2}$$

$$\sum_{(c,j)\in\mathcal{L}}\tau_{c,j}(e,v_2,v_3)=t_c(e,v_2,v_3)+\sum_{v_1\in\mathcal{V}}p_c(e,v_1,v_2)\chi(v_1,v_2,v_3),\quad \forall c\in\mathcal{C},e\in\mathcal{E},v_2,v_3\in\mathcal{V}. \tag{3}$$

$$\sum_{e\in\mathcal{E}}\sum_{v_1,v_2\in\mathcal{V}}\tau_{i,j}(e,v_1,v_2)\le x_{i,j}C_{i,j},\quad \forall(i,j)\in\mathcal{L}. \tag{4}$$

$$x_{i,j}\le\min\{y_i,y_j\},\quad \forall(i,j)\in\mathcal{L}. \tag{5}$$

$$\sum_{v\in\mathcal{V}}\sum_{e\in\mathcal{E}}\sum_{v_1\in\mathcal{V}}[r(v)p_c(e,v_1,v)+\rho(c)t_c(e,v_1,v_2)]\le k(c),\quad \forall c\in\mathcal{C}. \tag{6}$$

$$p_c(e,v_1,v)\le\delta(c,v)k(c),\quad \forall c\in\mathcal{C},e\in\mathcal{E},v,v_1\in\mathcal{V}. \tag{7}$$

$$\delta(c,v)\le y_c,\quad \forall c\in\mathcal{C},v\in\mathcal{V}. \tag{8}$$

$$l(e,v)=\sum_{(e,j)\in\mathcal{L}}\tau_{e,j}(e,v,v),\quad \forall e\in\mathcal{E},v\in\mathcal{V}. \tag{9}$$

$$E_0=\sum_{c\in\mathcal{C}}\sum_{v\in\mathcal{V}}f_0\left(\delta(c,v)\right); \tag{10}$$

$$E_{\text{proc}}=\sum_{c\in\mathcal{C}}f_{\text{proc}}\left(\sum_{v\in\mathcal{V}}r(v)\sum_{e\in\mathcal{E}}\sum_{v_1\in\mathcal{V}}p_c(e,v_1,v)\right); \tag{11}$$

$$E_{\text{idle}}=\sum_{c\in\mathcal{C}}f_{\text{idle}}\left(y_c\right); \tag{12}$$

$$E_{\text{sw}}=\sum_{(i,j)\in\mathcal{L}}f_{\text{sw}}\left(\sum_{e\in\mathcal{E}}\sum_{v_1,v\in\mathcal{V}}\tau_{i,j}(e,v_1,v)\right). \tag{13}$$

$$\min_{x,u}E=E_0+E_{\text{proc}}+E_{\text{idle}}+E_{\text{sw}}. \tag{14}$$

Figure 120: Optimization model

17.1.2 The logical graph

The logical graph, exemplified in Figure 118, accounts for where the traffic comes from, and how it is processed. Its vertices are either edge switches or VNFs. With reference to Figure 118, RRH is the only edge node, eNB, GW, MME and HSS are the VNFs.

On the logical graph, we have logical flows $\ell(e,v_1,v_2)$, representing data originating from edge switch e and going from VNF v1 to VNF v2. Additionally, we indicate with *l(e,v)* flows that start from edge e and are first processed at VNF v, e.g., from the RRH to the eNB in Figure 118.

It is important to stress that there is no flow conservation in the logical graph. As an example, in to Figure 118 we see a user flow of 1 traffic unit going from the RRH to eNB and thence to the gateway, which triggers some additional signaling traffic from the eNB and the gateway to the MME; indeed, for each VNF v, the generalized flow conservation law stated in (1) in Figure 120 holds.

The first member of (1) is the logical flow outgoing from VNF v2 and directed to VNF v3. Such a quantity is the product between logical flows incoming into v2, from either a VNF v1 or an edge switch e, and a factor χ(v1,v2,v3). Each value χ(v1,v2,v3) represents the logical flow directed to v3 that is generated when a unit of traffic coming from v1 is processed at VNF v2. With reference to the eNB in Figure 118, we have χ(RRH, eNB, GW) = 1, while χ(RRH, eNB, MME) = 0.3. Similarly, for the gateway, we have χ(eNB, GW, MME) = 0.2. At the MME we have flow conservation, i.e., χ(eNB, MME, HSS) = χ(GW,MME,HSS) = 1. Notice that we abuse the notation and allow the first index of χ to be an edge switch as well as a VNF.

It is important to stress that χ-values can also be lower than one, e.g., a firewall dropping some of the ingoing traffic. Also notice that χ-values different from one can happen for both user traffic (e.g., the eNB example earlier) and signaling (as in the case of the firewall).

17.1.3 The physical graph

In the physical graph, vertices are represented by edge switches and the switches. In 5G scenarios, core switches have computational capabilities k(c). It is possible to have k(c) = 0, corresponding to switches that can host no VNFs. Notice that for software switches, e.g., based on Lagopus [80], the computational capabilities k(c) are used for both switching and processing tasks.

Physical links links $(i,j) \in \mathcal{L} \subseteq (\mathcal{C} \cup \mathcal{E})^2$ have a capacity $C_{i,j}$, corresponding to the maximum amount of traffic that can go from switch i to switch j.

Our main variable is represented by physical flows $\tau_{i,j}(e,v_1,v_2)$, representing the amount of traffic that was originated from edge e, last visited VNF v_1, will next visit VNF v_2, and is now traveling on link (i, j) If the flow has never been processed, i.e., is going from e to its first VNF v, we will conventionally set $v_1 = v_2$ and write $\tau_{i,j}$ (e, v, v).

Given a core switch c, we denote as $t_c(e, v_1, v_2)$ the amount of traffic originated from e, that last visited VNF v_1 and will next visit VNF v_2 that is transiting by c, i.e., is not processed therein. Similarly, $p_c(e, v_1, v_2)$ is the corresponding traffic that is processed at core switch c, i.e., the v_2 step is performed at c.

A first constraint we need to impose is that traffic coming into core switch c is either processed or transiting, as stated in (2). A similar constraint is (3): it concerns outgoing traffic, which is given by the sum of transiting traffic and the traffic resulting from the processing of traffic at c, as foreseen by (1). In other words, (2)–(3) enforce ordinary flow conservation for the traffic that is transiting at core switch c, i.e., use c as a switch, and generalized flow conservation for the traffic that is processed at c, i.e., use its computational capabilities. In (3), v_1 is the last VNF that traffic visited before arriving at c, v_2 is the VNF implemented at c, and v_3 is the next VNF that processed traffic will have to visit.

Next, we need to ensure that we only use enabled switches and links, and do not use them for more than their capacity. We define two sets of binary variables, $x_{i,j}$ and y_c, telling whether edge (i,j) and core switch c respectively are enabled.

For links, we need to impose (4); furthermore, (5) ensures that no link can be enabled if either end thereof is disabled.

As for processing, each traffic unit processed by VNF v requires r(v) computational capacity, and, assuming c is a software switch, each unit of traffic transiting by a core switch c consumes ρ(c) computational capacity [4]. We need to ensure that the computational capacity of each switch c is sufficient for both, as stated in (6). Additionally, no processing can be done for VNFs that are not deployed at a given switch. We track this through a binary variable δ(c,v) expressing whether an instance of VNF v is deployed at core switch c, and impose (7). Finally, disabled core switches cannot host any VNF, as ensured by (8).

At last, we need to ensure that logical and physical flows match. To this end, it is sufficient that we ensure that each logical flow l(e,v) going from edge switch e to VNF v there are corresponding physical flows of the type $\tau_{e,j}$ (e, v, v), as ensured in (9).

Symbol	Type	Meaning
$\chi(v_1, v_2, v_3)$	Parameter	How much traffic meant for VNF v_3 results from the processing at VNF v_2 of one unit of traffic that was last processed at VNF v_1
$\mathcal{C}$	Set	Set of core switches
$C_{i,j}$	Parameter	Capacity of link $(i,j) \in \mathcal{L}$
$\delta(c, v)$	Binary variable	Whether we deploy VNF $v \in \mathcal{V}$ at core switch $c \in \mathcal{C}$
$\mathcal{E}$	Set	Set of edge switches
E_0, f_0	Function	Energy consumption due to placing a VNF at a core switch
$E_{\text{idle}}, f_{\text{idle}}$	Function	Energy consumption due to activating a core switch
$E_{\text{proc}}, f_{\text{proc}}$	Function	Traffic-dependent energy consumption due to processing
$E_{\text{sw}}, f_{\text{sw}}$	Function	Traffic-dependent energy consumption at links
$k(c)$	Parameter	Computational capability of core switch $c \in \mathcal{C}$
$\mathcal{L}$	Set	Set of links
$\ell(e, v_1, v_2)$	Parameter	Logical flow originated at $e \in \mathcal{E}$ and going from VNF $v_1 \in \mathcal{V}$ to VNF $v_2 \in \mathcal{V}$
l(e,v)	Parameter	Logical flow originating at $e \in \mathcal{E}$ and first being processed at VNF $v \in \mathcal{V}$
$p_c(e, v_1, v_2)$	Continuous variable	How much traffic coming from users connected to edge switch $e \in \mathcal{E}$ for service that was last processed at VNF v_1 is processed by an instance of VNF v_2 deployed at core switch c
$r(v)$	Parameter	How much computational capability is required to process one traffic unit of VNF $v \in \mathcal{V}$
$\rho(c)$	Parameter	How much computational capability is consumed by one unit of traffic transiting by switch $c \in \mathcal{C}$
$\tau_{i,j}(e, v_1, v_2)$	Continuous variable	How much traffic coming from users connected to edge switch $e \in \mathcal{E}$ that was last processed at VNF v_1 and meant to be next processed at VNF v_2 goes through link $(i,j) \in \mathcal{L}$
$t_c(e, v_1, v_2)$	Continuous variable	How much traffic originating from e that was last processed at VNF v_1 and meant to be next processed at VNF v_2 transits (without processing) by core switch $c \in \mathcal{C}$
$\mathcal{V}$	Set	Set of VNFs
$x_{i,j}$	Binary variable	Whether link $(i,j) \in \mathcal{L}$ is active
y_c	Binary variable	Whether core switch $c \in \mathcal{C}$ is active

Figure 121: Notation

[4] Non-software switches can be modeled by setting ρ(c) = 0.

17.1.3.1 *Energy and Objective*

There are four things that lead to energy consumption:

- placing a VNF on a core switch, resulting in a consumption E_0 due to, e.g., Docker overhead;
- using said VNF, resulting in a consumption of E_{proc} depending upon the computational resources used;
- activating a core switch, resulting in a consumption of E_{idle};
- having traffic going through links, resulting in a consumption of E_{sw} depending upon the traffic of each link.

We can express these four components as stated in (10)—(13). Finally, by combining them all, we can state our objective (14), i.e., minimize the total energy consumption.

17.1.4 Online Algorithms

The problem stated earlier falls into the MILP category, and is thus impractical to solve in real time. We can however solve a relaxed version thereof, where binary variables are allowed to take any value in [0, 1]. Optimal solutions to the relaxed models cannot be directly used to manage (or plan) a network; however, they can provide useful guidelines to that end.

Our basic idea is to leverage the software-defined nature of our network to integrate optimization with our SDN controller, i.e., optimize problems as a part of our network management strategy. Specifically, we proceed as outlined in

1. we initialize the system with a feasible (albeit potentially suboptimal) solution;
2. after that, we periodically:
    a. check that the current network configuration is adequate to the current (and/or future) traffic demand;
    b. if not so, enable additional core switches and/or links as needed;
    c. check whether there are core switches and/or links that can be disabled;
    d. if so, update the current network configuration accordingly.

Items (a)–(b) and (c)–(d) correspond to the fixProblems and saveEnergy procedures respectively, described next. It is worth pointing out that the fixProblems and saveEnergy procedures are designed to take no action if no action is warranted, and therefore there is no harm in cascading them. As an example, fixProblems will never take any action the first time it is executed after an initial solution is generated, as that solution is guaranteed to be feasible. Similarly, saveEnergy is unlikely to find network elements to disable if fixProblems had to enable some anew.

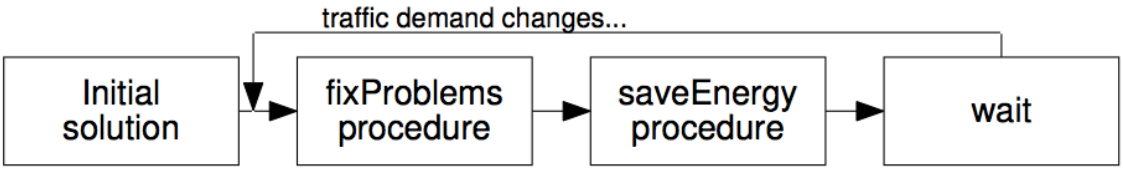


Figure 122: Our online algorithms. We begin by obtaining an initial feasible solution; after that, we periodically check the current solution for problems (procedure fixProblems), and for opportunities to deactivate some network elements (procedure saveEnergy).

17.1.4.1 *Initial Solution*

Notice that initial solution S^{curr} has to be feasible, but does not has to be optimal. Indeed, it can come as the output as one of the heuristics we reviewed earlier, e.g., [81]. Alternatively, it be crafted by solving a version of our problem where:

- all core switches and links are active;
- there is an instance of all VNFs active on each core switch;
- the other variables are set by solving the resulting problem.

The resulting solution will be highly suboptimal, as we are very likely to needlessly activate switches and/or links and to place useless VNF instances, all of which increase the power consumption. On the plus side, the problem to solve is LP, as all binary variables are fixed; furthermore, we can prove that:

Property. If a problem instance is feasible, then there is at least one feasible solution where the x, y and δ variables are all set to 1.

Proof. Let us consider a feasible solution S^0, where some of the binary variables are set to zero and others to one. By hypothesis, S^0 is feasible; what we need to prove is that changing all binary variables to one can never make us violate a constraint. This follows by inspection of (4), (5), (7), (8): if they hold for the variable values in S^0, then they still hold when all binary variables are set to one.

17.1.4.2 *The fixProblems procedure*

Algorithm The `fixProblems` procedure.

Require: $\mathcal{S}^{\text{curr}}$

1: $\mathcal{P} \leftarrow$ **new** problem()
2: $\mathcal{P}$.fix($x_{i,j} \leftarrow x_{i,j}^{\text{curr}}, \quad \forall (i,j) \in \mathcal{L}$)
3: $\mathcal{P}$.fix($y_c \leftarrow y_c^{\text{curr}}, \quad \forall c \in \mathcal{C}$)
4: $\mathcal{P}$.fix($\delta(c,v) \leftarrow \delta^{\text{curr}}(c,v), \quad \forall c \in \mathcal{C}, v \in \mathcal{V}$)
5: **solve**($\mathcal{P}$)
6: **if** $\mathcal{P}$.is_feasible **then**
7: **return**
8: **if** (4)$\in \mathcal{P}$.IIS **then**
9: $\mathcal{P}$.relax($x_{i,j}$: $x_{i,j}^{\text{curr}} = 0$)
10: $\mathcal{P}$.relax(y_c: $y_c^{\text{curr}} = 0$)
11: $\tilde{x}, \tilde{y} \leftarrow$ **solve**($\mathcal{P}$)
12: $(i^\star, j^\star) \leftarrow$ **choose** from $\mathcal{L}$ with probabilities $\tilde{x}_{i,j}$
13: $\mathcal{P}$.fix($x_{i^\star,j^\star} \leftarrow 1$)
14: $\mathcal{P}$.fix($y_i \leftarrow 1; y_j \leftarrow 1$)
15: **goto** Line 5
16: **if** (6)$\in \mathcal{P}$.IIS **then**
17: $\mathcal{P}$.relax($\delta(c,v)$: $\delta^{\text{curr}}(c,v) = 0$)
18: $\tilde{\delta} \leftarrow$ **solve**($\mathcal{P}$)
19: $c^\star, v^\star \leftarrow$ **choose** from $\mathcal{C} \times \mathcal{V}$ with probabilities $\tilde{\delta}(c,v)$
20: $\mathcal{P}$.fix($\delta(c^\star, v^\star) \leftarrow 1$)
21: **goto** Line 5

Algorithm 8: The fixProblems procedure

The high-level goal of the fixProblems procedure is to check whether the current network configuration is able to cope with the current traffic demand; if this is not the case, then we take one or more of the following actions: (i) enabling additional core switches; (ii) enabling additional links; (iii) deploying additional VNF instances.

Specifically, as detailed in Algorithm 8, we take as an input a feasible solution S^{curr} (Line 0). We then proceed, in Line 1–Line 4, to creating a new instance P of problem, where all binary variables are fixed to their values in S^{curr}. In Line 5, we solve such a problem: if it is feasible, then no action is required and the algorithm exits (Line 7). Otherwise, we look at why the problem is unfeasible, by inspecting its irreducible inconsistent subsystem (IIS), i.e., the subset of constraints such that removing any of them would make the problem feasible. This set allows us to discriminate between the different reasons that can make the network unable to operate properly (hence, the problem unfeasible).

If constraint (4) (mandating that no link is used for more than its capacity) is in the IIS, then we need to activate some more links and/or switches. To decide which ones, we relax all x- and y-variables related to switches and links that were inactive in $\mathbf{S^{curr}}$ (Line 9–Line 10) and solve the new problem (Line 11). We then choose one link to activate, with a probability proportional to its relaxed x(i,j) value (Line 12), and fix the corresponding x-value to 1 (Line 13), as well as the y-values of its endpoints (Line 14). We then go back to Line 5 and test the new solution (Line 15); if it is again infeasible, we will activate further links until feasibility is achieved.

We proceed in a similar way if constraint (6) is in the IIS, i.e., if we have a server capacity issue. We relax variables δ, allowing for more VNFs to be deployed if need be, and solve the new problem obtaining the relaxed values δ (Line 17–Line 18). We then have to decide where to place the new VNFs: we do so by selecting a core switch c* and a VNF v* at random, with a probability proportional to the relaxed values δ(c, v) (Line 19), fix the corresponding δ-variable to 1 (Line 20). Finally, we go back to testing the new solution (Line 21).

It is important to stress how all the problems we solve as a part of this algorithm are LP: in both Line 5, Line 11 and Line 18 all binary variables are either fixed (to 0 or 1) or relaxed. It follows that those problems can be solved in polynomial time; indeed, embedded optimization [82] on low-power hardware is now commonplace in several application domains.

17.1.4.3 *The saveEnergy procedure*

We can think of the *saveEnergy* procedure as the dual of *fixProblems*. Our aim is to identify core switches and/or links that we can disable, as well as VNF instances we can remove from the switches they run into, so as to reduce our power consumption without impairing our ability to serve the traffic, i.e., without making the problem infeasible. As in the *fixProblems* procedure, we solve a sequence of LP problems with fixed or relaxed variables, obtaining guidance on the decisions we should make and their effects.

Algorithm The `saveEnergy` procedure.

Require: $\mathcal{S}^{\text{curr}}$

1: $\mathcal{P} \leftarrow$ **new** problem()
2: $\mathcal{P}$.fix$(x_{i,j} \leftarrow 0, \quad \forall (i,j) \in \mathcal{L}\colon x_{i,j}^{\text{curr}} = 0)$
3: $\mathcal{P}$.fix$(y_c \leftarrow 0, \quad \forall c \in \mathcal{C}\colon y_c^{\text{curr}} = 0)$
4: $\mathcal{P}$.fix$(\delta(c,v) \leftarrow 0, \quad \forall c \in \mathcal{C}, v \in \mathcal{V}\colon \delta(c,v) = 0)$
5: $\mathcal{P}$.relax$(x_{i,j}, \quad \forall (i,j) \in \mathcal{L}\colon x_{i,j}^{\text{curr}} = 1)$
6: $\mathcal{P}$.relax$(y_c, \quad \forall c \in \mathcal{C}\colon y_c^{\text{curr}} = 1)$
7: $\mathcal{P}$.relax$(\delta(c,v), \quad \forall c \in \mathcal{C}, v \in \mathcal{V}\colon \delta(c,v) = 1)$
8: **solve**$(\mathcal{P})$
9: $(x^\star, y^\star) \leftarrow \arg\min_{(x,y)\in\mathcal{L}\colon x_{x,y}^{\text{curr}}=1} \tilde{x}_{i,j}$
10: $c^\star \leftarrow \arg\min_{c\in\mathcal{C}\colon y^{\text{curr}}(c)=1} \tilde{y}(c)$
11: $d^\star, v^\star \leftarrow \arg\min_{c,v\in\mathcal{C}\times\mathcal{V}\colon \delta^{\text{curr}}(c,v)=1} \tilde{\delta}(c,v)$
12: $\mathcal{P}_2 \leftarrow$ **copy**$(\mathcal{P})$
13: **if** $\tilde{x}_{i^\star,j^\star} < \tilde{y}(c^\star) \wedge \tilde{x}_{i^\star,j^\star} < \tilde{\delta}(d^\star, v^\star)$ **then**
14: $\quad \mathcal{P}_2$.fix$(x_{i^\star,j^\star} \leftarrow 0)$
15: **if** $\tilde{y}(c^\star) < \tilde{x}_{i^\star,j^\star} \wedge \tilde{y}(c^\star) < \tilde{\delta}(d^\star, v^\star)$ **then**
16: $\quad \mathcal{P}_2$.fix$(y(c^\star) \leftarrow 0)$
17: $\quad \mathcal{P}_2$.fix$(x_{i,j} \leftarrow 0, \quad \forall (i,j) \in \mathcal{L}\colon i = c^\star \vee j = c^\star)$
18: $\quad \mathcal{P}_2$.fix$(\delta(c,v) \leftarrow 0, \quad \forall c \in \mathcal{C}, v \in \mathcal{V}\colon c = c^\star)$
19: **if** $\tilde{\delta}(d^\star, v^\star) < \tilde{x}_{i^\star,j^\star} \wedge \tilde{\delta}(d^\star, v^\star) < \tilde{y}(c^\star)$ **then**
20: $\quad \mathcal{P}_2$.fix$(\delta(d^\star, v^\star) \leftarrow 0)$
21: **solve**$(\mathcal{P}_2)$
22: **if** $\mathcal{P}_2$.is_feasible **then**
23: $\quad \mathcal{P} \leftarrow \mathcal{P}_2$
24: $\quad$ **goto** Line 1
25: **else**
26: $\quad$ **return** $\mathcal{P}$

Algorithm 9: The saveEnergy procedure

In Algorithm 9, we take the current solution S^{curr} as an input. We then create an instance P of the problem where the binary variables that in the current solution have value zero are fixed to zero (Line 2–Line 4), and those that have currently value one are relaxed (Line 5–Line 7). This reflects the fact that we are not looking for new network elements to activate, but rather to currently-active elements to deactivate. We look for such elements by solving the problem instance P (Line 8); notice that all binary variables therein are fixed or relaxed, so the problem is LP.

In Line 9–Line 11 we identify the link, core switch, and pair of core switch and VNF that are active in the current solution and have the lowest value of the associated relaxed variable (respectively $x_{i,j}$, y_c, and $\delta(c,v)$). Intuitively, these are the network elements we will most likely able to deactivate without impairing network functionality. We check this by creating a copy of problem instance P (Line 12) and fixing to 0 the binary variable associated to the network element with the lowest score (Line 13–Line 20). If that element is a core switch, we also need to disable the links using it and the VNF instances it used to host (Line 17–Line 18).

The difference between problem instances P and P_2 is that exactly one network element that was active in P is deactivated in P_2 (this implies that P_2 is also LP). In Line 21, we solve P_2 and check it is feasible. If that is the case, then we use P_2 as our new solution, and try to further enhance it (Line 23–Line 24). Otherwise, the algorithm returns P, the last feasible solution we tried, and exits (Line 26).

In summary, Algorithm 9 disables zero or more network elements, i.e., core switches, links, or VNF instances. The element to disable is chosen based on the value taken by the corresponding relaxed variables, and after each change we check that the resulting network configuration can serve its load, i.e., the corresponding problem instance is feasible.

17.1.4.4 *Novelty*

Our approach is novel in several important ways, that distinguish it from the 40+ works examined in [83]:

1. first and foremost, the scope of our work: we jointly account for the dimensions of (i) VNF placement, (ii) traffic steering; (iii) server activation, and (iv) network management, e.g., activating/deactivating core switches, which none of the existing works accounts for.
2. at the modeling level: including edge switches, i.e., points at which the traffic originates, in the VNF graph, and in general accounting for the fact that traffic does not originate at the first VNF;
3. as far as objectives are concerned: adopting energy-saving as a top priority and using detailed and realistic energy models, as opposed to hand-waving remarks

like "maximizing server utilization is tantamount to minimizing energy consumption";
4. from a solution strategy viewpoint: optimizing in the loop, i.e., using optimization as a tool rather than something to try and fail at the beginning.

Of course, not everything about our approach is novel. Specifically, the following aspects are shared with earlier works:

- formulating a MILP problem and then proposing a heuristics;
- using a DAG, or indeed a graph, for VNFs;
- taking an iterative approach and scheduling one flow or placing one VNF at a time.

17.2 Additional information regarding the consolidated design of RMA

17.2.1 Mathematical System Model

We consider a scenario comprised of M CUs $\mathcal{B}:=\{B_1,\cdots,B_M\}$, N RUs $\mathcal{R}:=\{R_1,\cdots,R_N\}$, and a packet-based network connecting RUs to CUs by means of packet-switching nodes $\mathcal{V}:=\{v_1,\cdots|\mathcal{V}|\}$. Nodes communicate via network links such that $l_{i,j}=1$ denotes an existing link between nodes i and j and $l_{i,j}=0$ otherwise. The collection of all nodes is denoted by $\mathcal{N}:=\mathcal{B}\cup\mathcal{R}\cup\mathcal{V}$. Note that we do not make any assumptions on topology structure in an attempt to shed some light for arbitrary large-scale scenarios. Also, without loss of generality we will focus on the downlink case.

17.2.1.1 *Flexible RAN functional split*

RUs are in charge of analog processing and digital-to-analog conversion. The remaining functionality of a traditional BS (modulation, HARQ, scheduling, etc.) is split into a set of H atomic functions $\mathcal{F}:=\{f_1,\cdots,f_H\}$ that can be executed by either a CU or an RU though, importantly, they must be processed sequentially. This is known as *flexible functional split*. As we explained above, offloading RAN functionality into a cloud platform (CU) has the advantages of lower operational costs (e.g. common refrigeration, single-point maintenance, etc.) and capacity gains to users (joint signal processing, coordinated resource allocation, etc.). However, the delay and throughput requirements for the transport network between CUs and RUs become more stringent when a larger number of functions are offloaded.

To model this, we let $\theta_{m,n}\subseteq\mathcal{F}$ denote the subset of BS n functions assigned to CU m. In turn, $\overline{\theta}_n:=\mathcal{F}\setminus\bigcup_{m\in\mathcal{B}}\theta_{m,n}$ is the subset assigned to RU n. We impose a one-to-one mapping between CUs and RUs and therefore $\theta_{m,n}=\varnothing\,\forall m\in\mathcal{B}\setminus\{A_n\}$, where A_n returns the CU assigned to RU n, i.e., we do not consider chaining functions across different CUs. In this way, we can model traditional scenarios like C-RAN (setting $\theta_{A_n,n}=\mathcal{F}$ and $\overline{\theta}_n=\varnothing\,\forall n\in\mathcal{R}$), traditional D-RAN (with $\theta_{A_n,n}=\varnothing$ and $\overline{\theta}_n=\mathcal{F}\;\forall n\in\mathcal{R}$), and any other configuration between these two.

17.2.1.2 *Routing and CU assignment*

We assume a flexible transport protocol (NGFI) that is able to carry IQ samples from CUs to RUs as well as traffic from different BS splits and BH traffic onto the same substrate (switching-based) Crosshaul infrastructure. Hence, our network must transport N flows (each associated with one RU) with different demands that depend on the functional split of each BS. The path followed by flow n is comprised of a subset of the forwarding nodes $\mathcal{P}_{m,n} \subseteq \mathcal{V}$ from CU $m = A_n$ to RU n such that for any $v_i \in \mathcal{P}_{m,n}$ there is exactly another $v_{j\neq i} \in \mathcal{P}_{m,n}$ with $l_{v_i,v_j} = 1$ (i.e. no loops). If CU $m \neq A_n$ (does not serve RU n), then $\mathcal{P}_{m,n} = \varnothing$. The network between CUs and RUs must satisfy the delay/throughput requirements of a given RAN split $\boldsymbol{\theta} := \{\theta_{m,n} \mid \forall m \in \mathcal{B}, \forall n \in \mathcal{R}\}$. In turn, the transport capacity depends on the routing choices $\mathcal{P} := \{\mathcal{P}_{m,n} \mid \forall m \in \mathcal{B}, \forall n \in \mathcal{R}\}$ between CUs and RUs. Thus, as said earlier, we face a problem where routing $\mathcal{P}$ and BS splits $\boldsymbol{\theta}$ must be optimized *jointly*.

17.2.1.3 *RU Clustering*

The main challenge of our problem is the large space of candidate solutions, i.e. a network has $(k \cdot M)^N \cdot (|\mathcal{F}| + 1)^N$ possible settings, where k is the number of possible paths between any CU/RU. In order to reduce such huge space, we leverage the fact that joint processing of different BSs mostly makes sense if it is done within the same CU. Moreover, there is little gain to do joint processing of a set of BSs with different functional splits. We thus assume that the set of RUs is partitioned into $\mathcal{Q} := \{q_1, \ldots |\mathcal{Q}|\}$ clusters where q_i contains a subset of RUs constrained to the same split choice and CU assignment. Note that RUs can still follow a different route to their CU even if they belong to the same cluster. For instance, $|\mathcal{Q}| = 2$ in our toy example. This brings down the space of candidate solutions to $k^N \cdot M^{|\mathcal{Q}|} \cdot (|\mathcal{F}| + 1)^{|\mathcal{Q}|}$. Since our focus is on joint routing and RAN centralization, we will rely on state-of-the-art clustering mechanisms.

17.2.2 Optimization Framework

Our goal is threefold: (i) pair CUs to RUs (or clusters of RUs), (ii) set the paths between CUs and RUs, and (iii) choose the functional decomposition of BSs in an optimal way. To this aim, we rely upon estimates (e.g. in peak hour) of user demands at each radio site and capacity/latencies in each network link. We expect configuration changes across the Crosshaul (if possible) to happen in large timescale (hours, days or even weeks) and so we do not target adaptation to very quick events, e.g. fading events in the wireless channel. Although the capacity gains *vs.* cost trade-offs due to function centralization could be modeled to some extent, the actual benefits are much broader than such quantitative measures (see Table 5 and are hard to model in general). For instance, maintenance is simplified by centralizing functions which should then reduce costs; however, the extent of these gains depends on several factors which differ across operators. In the literature today we can find some models on the trade-offs between full centralization (C-RAN) and distributed BSs (D-RAN). However, these are only two

split options out of the many we consider here. Recent experimental studies has shown computational costs of different functions of Open Air Interface (OAI)'s LTE protocol stack (a well-known software-defined radio implementation); however, they do not model the gains of centralization, which remain an open issue.

In this paper, given such research gap, we use a simple model that serves our purpose, to study the implications of flexible RAN centralization in a packet-based FH, and leave a more accurate modeling of different splits gains/costs for future work. Thus, we aim at maximizing the ***degree of centralization*** γ of our system subject to network constraints:

$$\max_{\boldsymbol{\theta},\mathcal{P}} \gamma := \left[\frac{1}{N}\sum_{n\in\mathcal{R}}\left(\alpha\sum_{f\in\theta_n}\varsigma_f(s_n)+\sum_{f\in\overline{\theta}_n}\varsigma_f(s_n)\right)\right]^{-1}$$

$$\text{s.t.}$$

$$\sum_{i\neq j\in\mathcal{P}_{m,n}} l_{i,j}\cdot\delta_{l_{i,j}} \leq d(\theta_{m,n},s_n), \quad \forall m\in\mathcal{B},\forall n\in\mathcal{R}$$

$$\sum_{m=1}^{M}\sum_{n=1}^{N} I(i,j,\mathcal{P}_{m,n})\cdot b(\theta_{m,n},s_n) \leq \beta_{l_{i,j}}, \forall i\neq j\in\mathcal{N}$$

where $\varsigma_f(s_n)$ models the computational burden of function f given configuration s_n (MIMO setting, bandwidth, etc.) of BS n, and $0 \leq \alpha \leq 1$ models the relative cost savings when a function is centralized, i.e., if $\alpha = 1$, centralizing a function is as costly as maintaining the function co-located with the RU and therefore our problem becomes a simple feasibility problems where D-RAN and C-RAN configurations provide the same cost to the system. Conversely, if $\alpha < 1$ there is a cost saving when a function is centralized (e.g. coming from pooling gains) and our optimization problem will lean towards C-RAN as much as possible. The first set of constraints handle delay requirements, where $\delta_{l_{i,j}}$ is the latency of link $l_{i,j}$, which we assume is an additive constant for simplicity (i.e. we do not consider queueing effects)[5] and $d(\theta_{m,n},s_n)$ is the delay requirement of split $\theta_{m,n}$ and setting s_n. We skip jitter constraints because they can be mitigated with buffers at the receivers at the cost of latency. Second set of contraints handles capacity violations, where $\beta_{l_{i,j}}$ is the total bit-rate capacity of link $l_{i,j}$, $I(i,j,\mathcal{P}_{m,n})$ is an indicator function which is 1 if nodes i and j are contained in $\mathcal{P}_{m,n}$ and 0 otherwise, and $b(\theta_{m,n},s_n)$ gives the throughput demanded by an RU with split $\theta_{m,n}$ and setting s_n (including all protocol overheads). For instance, Table 5 shows values of $d(\cdot)$ and $b(\cdot)$ for which, without loss of generality, we consider the functions, splits and costs given in Table 41 (the shaded and white area highlight the functions running in a CU and an RU, respectively), and assume $s_n\ \forall n$ given in Table 5 to simplify our evaluation.

Table 41: Computational costs based on CPU execution times taken from experimental

[5] Frame preemption (802.1Qbu) and scheduled traffic (802.1Qbv) can be used mitigate queueing effects.

measurements for some splits in Table 5

LTE subfunction (f)	Others	RLC	MAC	PHY 2	PHY 1
Relative cost (ς_f)	0.2	0.01	0.14	0.17	0.48
Split 1 (B in Table 5)	CU	RU			
Split 2 (C in Table 5)	*CU*		RU		
Split 3 (E in Table 5)	CU			RU	
Split 4 (G in Table 5)	CU				RU
Split 5 (J in Table 5)	CU				

17.2.3 Algorithms

We let the triple $X_n := \{A_n, \theta_{A_n,n}, \mathcal{P}_{A_n,n}\}$ describe a configuration of BS n, and $\mathbf{X} := \{X_n \mid \forall n \in \mathcal{R}\}$ be a candidate solution to our problem. Our goal is to find the optimal $\mathbf{X}^*$ that maximizes the degree of centralization γ. Since this is an integer non-linear optimization problem ($b(\cdot)$, $d(\cdot)$ or $\varsigma_f(\cdot)$ can be discontinuous non-linear functions), we focus on combinatorial algorithms. In order to reduce complexity, we constraint our combinatorial search to settings where all BSs within the same cluster have the same split and CU assignment. We propose two algorithms: a nearly-optimal branch-and-bound backtracking algorithm (BBB), and a low-complex greedy approach (GA).

17.2.3.1 *Backtracking Branch-and-Bound (BBB)*

BBB is an optimal branch-and-bound approach that explores a discrete space of candidate solutions. This type of algorithms has the advantage of being highly parallelizable, i.e. suitable for cloud computing platforms. First, we store in $\Pi_n := \{\mathcal{P}_{m,n}^{(1)} \cdots \mathcal{P}_{m,n}^{(k)} \mid \forall m \in \mathcal{B}\}$ k candidate paths between each CU m and RU n, for all $n \in \mathcal{R}$. This can be readily obtained with k-shortest path routing versions of Dijkstra, for example, with complexity $\mathcal{O}(MNk(L + |\mathcal{N}|\log|\mathcal{N}|))$ with $:= \sum_{i,j\in\mathcal{N}} l_{i,j}$.

Then, the space of candidate solutions can be represented as a tree where a node i in level τ represents one possible setting $X_{\psi(\tau)}^{(i)}$ for RU $\psi(\tau)$, where $\psi(\tau)$ is a function that maps a level τ to RU n ($\psi(0) = \varnothing$ is the root level) and i is a point in the configuration space $\mathcal{C}_n := \{(m, \theta_{m,n}, \mathcal{P}_{m,n}) \mid m \in \mathcal{B}, \theta_{m,n} \in 2^{\mathcal{F}}, \mathcal{P}_{m,n} \in \Pi_n\}$. A full branch represents thus a candidate **X** (see Algorithm 10).

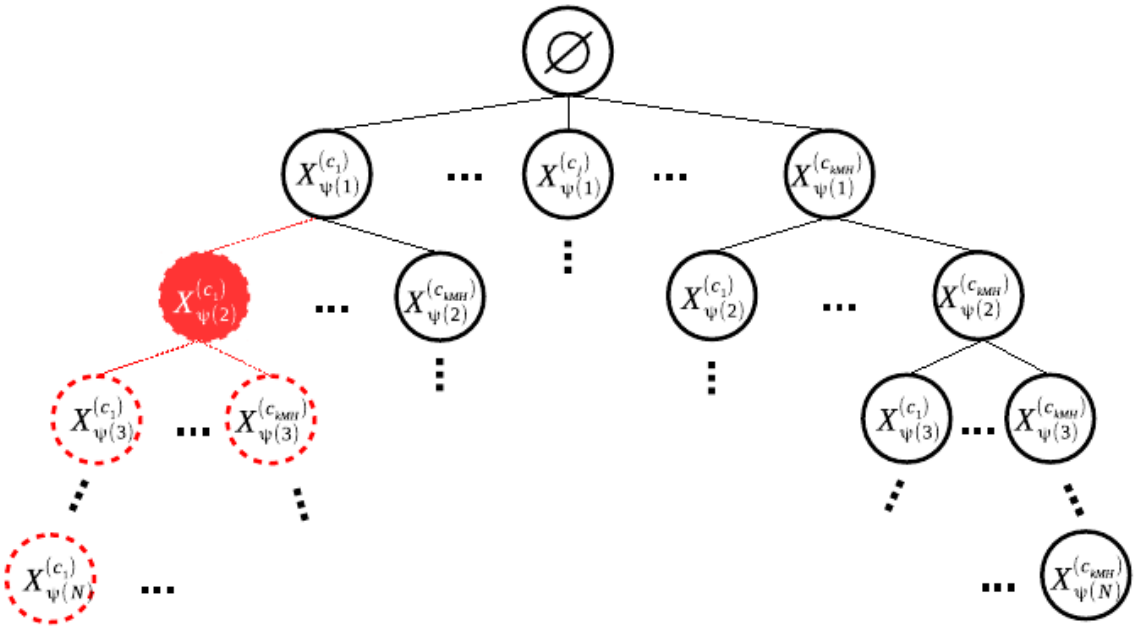


Figure 123: BBB algorithm. Partial candidate ($\{X_{\psi(1)}^{(c_1)}, X_{\psi(2)}^{(c_1)}, \cdots\}$) violates constraints and all hanging branches are thus pruned.

17.2.3.2 *Greedy Algorithm (GA)*

Unfortunately, the complexity of branch-and-bound approaches does not scale well in general, particularly if an initial upper bound is not properly chosen. For this reason, we now propose a low-complex combinatorial algorithm described in Algorithm 10.

We make use of a heuristic *score function* that pre-evaluates how "good" configuration X_n is. To this aim, we define F_{X_n} as

$$F_{X_n} := \frac{W(\theta_{m,n})}{\sum_{p \in \Phi(\mathcal{P}_{m,n})} \frac{b(\theta_{m,n}, s_n)}{\min\limits_{\{l_{i,j} | \forall i,j \in p\}} \beta_{l_{i,j}}}},$$

where $W(\theta_{m,n}) := \left(\alpha \sum_{f \in \theta_i} \varsigma_f(r_i) + \sum_{f \in \overline{\theta}_i} \varsigma_f(r_i)\right)^{-1}$ is a *reward* function: the individual degree of centralization of BS n when using split $\theta_{m,n}$ and CU m. In the denominator, function $\min_{\{l_{i,j} | \forall i,j \in p\}}$ represents the maximum capacity of a path p (i.e. the bottleneck link on that path) and $b(\theta_{m,n}, s_n)$ is the throughput requirement of BS n when using configuration X_n (i.e. functional split $\theta_{m,n}$). Additionally, $\Phi(\mathcal{P}_{m,n})$ is a set that collects all paths $p \in \bigcup_{j \in \mathcal{R} \setminus n} \Pi_j$ (all potential paths for all CU/RU pairs) that *share some link with the path considered in configuration* X_n, $\mathcal{P}_{m,n}$. In this way, the denominator of the above equation sums up across the load required by flow n relative to the maximum capacity of path p *for all paths that share some link with the path under consideration by configuration* X_n. This serves us as a rough estimation of the penalty incurred by any split choice and path combination. *Therefore,* F_{X_n} *represents*

the reward of using path $\mathcal{P}_{m,n}$ and split $\theta_{m,n}$ (i.e. configuration X_n) relative to an estimation of the burden such choice would add to the whole system. If, e.g., a path p shares many links with other paths that could potentially be used by other flows, the denominator would contribute to lower the score of X_n, which is maximum for disjoint paths (with no links in common) and high-centralization functional splits. In this initialization phase, we pre-compute $F_{X_n^{(i)}} \forall i \in \mathcal{C}_n, \forall n \in \mathcal{R}$. We will use this information in our branching phase. The algorithm assumes that $F_{X_n^{(i)}}$ has been computed for all RU $n \in \mathcal{R}$ and all $i \in \mathcal{C}_n$. (Although $\mathcal{C}_n$ is built using a set of pre-computed routes, this algorithm is not constrained to choose them). Note that we abuse notation and we let θ_q denote the functional split $\theta_{A_n,n}$ for any RU n in cluster $q \in \mathcal{Q}$ (since all of them shall have the same setting).

Then GA algorithm greedily increases the functional split setting of the cluster $q \in \mathcal{Q}$ with largest ψ_q such that

$$\psi_q := \sum_{n \in q} \max_{\forall i \in \mathcal{C}_n} \left\{ F_{X_n^{(i)}} \mid \theta_q = \{\theta_q^*, f_{|\theta_q^*|+1}\} \right\}$$

ψ_q is another score function that we use as a heuristic approach to favor higher degrees of centralization of RUs in larger clusters with a roughly higher likelihood to satisfy constraints. The intuition is that larger clusters contribute to increasing ψ_q (because we are summing up across all RUs in the cluster) and, given that a higher F_{X_n} has (roughly) higher chances of meeting constraints (because RU n overlaps with less and higher-capacity links used by others), clusters with RUs that have good "best potential configurations" will also contribute to increasing ψ_q. Thus, clusters with higher ψ_q are in better position to cause less *damage* if their split is increased but also have higher improvement over the degree of centralization (because there are more RUs in the cluster).

This score is computed in step (11) of Algorithm 10. The algorithm tests BS configurations greedily in an orderly fashion based on the above score. Step (13) selects the cluster with highest aggregated score and increases the degree of centralization of all the BSs within the cluster (step (14)). Then, for every split change, we invoke find_routes(·) (step (15)) to find a CU assignment first and a feasible routing instance second. GA algorithm has a worst-case complexity of $\mathcal{O}(|\mathcal{Q}||\mathcal{F}| + 1)$ times that of the CU assignment and routing algorithms which we present next. Our CU assignment algorithm is a simple heuristic that pairs each cluster to the CU which is closest (in terms of lowest latency) to the centroid of the cluster. This can be readily done with complexity $\mathcal{O}(M(L + |\mathcal{N}|\log|\mathcal{N}|))$ applying Dijkstra.

We propose two flavors of GA algorithm: **GA-RR** where find_routes(·) implements a routing function based on randomized rounding **and GA-GR** where find_routes(·) implements a greedy routing algorithm using Dijkstra algorithm in an orderly fashion.

Algorithm 1 Greedy algorithm.

```
function GREEDY_OPT(B, R, V, Q)
    /*Initialization*/
    for ∀q ∈ Q do
        θ'_q = ∅
    end for
    (P', γ') = find_routes(B, R, V, Q, θ')
    while |P'| = |R| do
        /*All flows could be routed*/
        (θ*, P*, γ*) ← (θ', P', γ')
        for ∀q ∈ Q, |θ_q| < |F| do
            ψ_q = Σ_{n∈q} max_{∀i∈C_n} { F_{X_n^(i)} | θ_q = {θ*_q, f_{|θ*_q|+1}} }
        end for
        q ← arg max(ψ)
        θ'_q ← {θ*_q, f_{|θ*_q|+1}}
        (P', γ') = find_routes(B, R, V, Q, θ')
    end while
    return (θ*, P*, γ*)
end function
```

Algorithm 10: Greedy Algorithm

17.3 Baseline algorithm for VIMaP Virtual Infrastructure placement

We have designed and developed a baseline algorithm for the implemented service of allocating a graph of interconnected virtual machines, referred to as the Virtual Machine Graph (VMG) allocation problem.

In this section we first describe the general Virtual Machine Graph (VMG) allocation problem. Then, we present a reduction of the problem based on constructing the aggregated VMG solution graph, where the objective is to find groups of VMs to be allocated together in the same substrate hosting nodes. This reduction is modeled based on a constrained mapping function. Finally, a heuristic algorithm solution to this problem is proposed.

17.3.1 Virtual Machine Graph allocation problem definition

We model the substrate infrastructure as a directed graph and denote it by $G^S = (N^S, H^S, L^S)$, where N^S is the set of substrate switching nodes, H^S is the set of substrate hosting nodes (DCs) and L^S denotes the set of substrate links $l_s = (u; v)$; $l_s \in L^S$ and $u, v \in N^S \cup H^S$.

Virtual machine graph request (VMGP). We denote by a directed graph $G^V = (H^V, L^V)$ the VMGP request. H^V denotes the set of virtual hosts (VMs) and L^V denotes the set of links between virtual hosts. Now we define a set of capacity functions for the substrate and virtual resources. Each host *h* (physical or virtual) is attributed with a set of *A*

attributes whose capacities are denoted as $c_{CPU}(h)$, $c_{MEM}(h)$, $c_{STO}(h)$ (we consider only CPU, memory and storage as host attributes), and each link l has a bandwidth capacity bw(l). We also denote P^S as the set of free loop paths in the substrate network between hosting nodes. The objective is to find a mapping function for all virtual hosts and links to the substrate infrastructure.

Next, a reduction of the problem is proposed and the constrains in terms of capacities for hosts and links are introduced.

17.3.2 VMG mapping problem

To solve the above described problem, we propose a first reduction which consists in: a) finding a VM allocation among the substrate hosting nodes and, b) find a feasible allocation solution for the links connecting VM in different hosting nodes. It is assumed that several virtual hosts can be placed in the same substrate hosting node if enough computing resources are available in the substrate node for the aggregated capacity of the virtual hosts allocated to it.

The mapping function can be split as hosting and link mapping, where the **hosting mapping function** satisfies that the sum of the capacities for each virtual host does not exceed the hosting nodes.

In order to compare the sizes of the hosts (physical or virtual) in relative terms, we define the function weight, as the weighted sum of the individual computing capacities, using α, β and γ to weight up.

$$w(h) = \alpha\ c_{CPU}(h) + \beta\ c_{MEM}(h) + \gamma\ c_{STO}(h)$$

The link mapping function satisfies the constrained shortest paths.

17.3.3 Baseline VMG embedding algorithm

The problem has been reduced to find a feasible allocation for the solution graph G, considering two steps:

Step 1: Following a Greedy procedure, we select the minimum number of substrate hosting nodes with enough capacity to allocate all the virtual hosts in H^V, which are embedded sequentially following a First Fit approach (Algorithm 11).

Step 2: Based on the selected H, we employ the Constrained Shortest Path First (CSPF) algorithm to find a feasible path in the substrate network, for each s-t pair allocated in different substrate hosting nodes (Algorithm 12).

Algorithm 11 first computes the Greedy and the First Fit (FF) host mapping procedure to find the minimum cluster with enough capacity to allocate virtual hosts within the VMG request. Firstly, it sorts the substrate host set in decreasing order by weight and it sequentially allocates the virtual hosts into the substrate hosting nodes with higher capacities. As a result, this function returns the solution subset with minimum size.

Algorithm 1 GreedyFFHostMapping(H^S, H^V)

Input: H^S: Substrate hosting nodes, H^V: Virtual hosts.
Output: $H', H^{S'}$: host solution set
  Sort $H^S = h_1^s, h_2^s, ..., h_n^s$ in decreasing order by its weight.
  $H^{S\prime} \leftarrow \emptyset$
  $H' \leftarrow \emptyset$
  $H^{v\prime} \leftarrow H^V$
  while $\sum_{\forall h^{s\prime} \in H^{S\prime}}(c_a(h^{s\prime})) < \sum_{\forall h^v \in H^V}(c_a(h^v))$,
    $\forall a \in A$ **do**
    $h^s \leftarrow H^S.pop()$
    $currentC_a \leftarrow c_a(h^s), \forall a \in A$
    $current_s \leftarrow \emptyset$
    for v **in** $H^{v\prime}$ **do**
      if oneOf $currentC_a < c_a(v), \forall a \in A$ **then**
        $H^{v\prime} \leftarrow H^{v\prime} - current_s$
        break
      else
        $current_s \leftarrow current_s \cup \{v\}$
        $currentC_a \leftarrow currentC_a - c_a(v), \forall a \in A$
      end if
    end for
    $H' \leftarrow H' \cup current_s$
    $H^{s\prime} \leftarrow H^{s\prime} \cup h^s$
  end while
  return $M^H : H' \mapsto H^{S\prime}$

Algorithm 11: GreedyFFHostMapping

Algorithm 12 receives the host solution subset and both substrate and virtual links of the VMG request. Based on the host mapping solution, for each virtual link, a feasible path p between nodes allocated. We use the CSPF algorithm with the bw(u; v) as a constrain parameter. If there is a feasible path for each l the mapping solution is returned.

Algorithm 2 CSPF Link Mapping ($H', H^{S'}, L^S, L^V$)

Input: $H', H^{S'}$: substrate host solution set,
  L^S: Input substrate links,
  L^V: Input links request
Output: $M : (H', H^{S'}), (L', P^{S'})$: Mapping solution from $G^V \mapsto G^S$
  for (u, v) **in** L^V **do**
    if $h'_u \neq h'_v$ **then**
      $L' \leftarrow L' \cup (u, v)$
    end if
  end for
  for $l'(u, v)$ **in** L' **do**
    $p^{s\prime} \leftarrow CSPF(G^S, h_u^{s\prime}, h_v^{s\prime}, bw(l'))$
  end for
  return $M : (H', L') \mapsto (H^{S'}, P^{S'})$

Algorithm 12: CSPF Link Mapping Algorithm

For further details, please see [84]

17.4 Algorithms for joint Path Computation and Virtual Network Function Placement Service in Resource Manager Application

RMA algorithm for joint Path Computation and Virtual Network Function Placement (*PC-VNFP*) service is formulated as an equivalent Integer Linear Programming (ILP) problem. This formulation not only focus on computing the optimal path in infrastructure composed of XPFEs and XPUs together with the optimal placement of the VNFs on XPUs, but also seeks to minimize the utilization of resources.

A possible way of approaching this complex task of joint placement of services and routing of the flows while ensuring the optimum resource utilization is through solving an optimization problem which consists of minimizing the overall cost function in terms of (i) **VNF placement**, and (ii) **Flow allocation.** The solution to the ILP formulation is made in terms of the minimization of the objective function (defined in Eq. (1)) subject to (s.t.) several different constraints that stands for the PC-VNFP service provided by the RMA.

$$U = \min\left(C_{VNF} + C_f + C_d\right) \ , \qquad (1)$$

where C_{VNF} is the cost associated to deploying a VNF over an XPU node; C_f denotes a fixed parametric cost and C_d is a dynamic parametric cost. For the sake of developing the ILP problem formulation, the relevant algorithm parameters are being introduced in Table 42 and Table 43. Generically, network nodes are denoted here as u and v. Implicitly, it is assumed that a switching element is connected to an XPU in such a way to enable network wide connectivity (i.e. connecting the XPU to XPFEs). A network flow is a traffic from/to the mobile core network to/from users, depending on whether it is respectively downlink or uplink traffic. Realistically, to each flow is associated bandwidth and latency demand, which the RMA algorithm must fulfil to be successful. In the remainder of this section the ILP formulation below relies also on the work done in [85] and [44].

Table 42: Description of Network-specific Parameters.

Parameter	Description
$G(V,E)$	Network graph to topology, *V* is set of nodes (XPFEs and XPUs) and *E* is set of edges.
$W_{(u,v)}$	{0,1}: 1 if there exists an edge between node *u* and *v*; 0 otherwise
$c(u,v)$	Capacity of link (*u*,*v*) between node *u* and *v*
$l(u,v)$	Latency of link (*u*,*v*).
$k^c_{(u,v)}$	Fixed cost of using the edge (*u*,*v*) i.e. if any amount traffic, greater than zero, passes through the edge (*u*,*v*), this is the cost paid

$k^d_{(u,v)}$	Dynamic cost of unit flow that passes through edge (u,v)
h^n_i	Fixed cost of instantiating a VNF of type n on node $i \in V$
O_v	Cores available at node $v \in V$. Each core can support one VNF
U_s	Number of flows service $s \in S$ can support on one core
M^n_i	{0,1}: 1 if service $n \in S$ can be supported at node i, 0 otherwise

Table 43: Description of Flow-specific Parameters.

Parameter	Description
F	Set of all flows in the network
s^f	Start node of flow $f \in F$
t^f	Destination node of flow $f \in F$
d^f	Capacity demand of flow $f \in F$
l^f	Latency demand of flow $f \in F$. Latency that flow incurs while moving from source to destination should be less than this value.
K	Set of all services (i.e. VNFs) that can be placed on XPU nodes
C^f	Service chain of flow $f \in F$. Set of services (i.e. VNFs) that flow $f \in F$ needs to traverse in a specific order. i.e. $n_1 \rightarrow n_2 \rightarrow \cdots \rightarrow n_l$ where $n_i \in K$
C^f_{st}	$[n_{sf} \rightarrow C^f \rightarrow n_{tf}]$ Service chain of flow $f \in F$ which includes s^f and t^f nodes. To ensure that flow starts at node s^f and ends at node t^f, two virtual services n_{sf} and n_{tf} are introduced at s^f and t^f, nodes respectively. Since n_{sf} and n_{tf} services are only present at s^f and t^f nodes, these nodes are selected as start and end node of the flow path.

Variables which are needed in the ILP problem formulation are provided herein below.

- $x^{f(n\rightarrow m)}_{(u,v)}$: {0,1}. 1 if edge (u,v) is used to reach from service n to m in the FG C^f of flow $f \in F$; 0 otherwise.
- $x_{(u,v)}$: {0,1}. 1 if any flow passes over edge (u, v); and 0 otherwise. This is not a decision variable. In specific,

$$x_{(u,v)} = \begin{cases} 1, & \text{if } \sum_{f\in F}\sum_{(n\rightarrow n)\in C_{st}{}^f} x^{f(n\rightarrow m)}_{(u,v)} + \sum_{f\in F}\sum_{(n\rightarrow n)\in C_{st}{}^f} x^{f(n\rightarrow m)}_{(v,u)} > 0 \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

The condition shown in equation (2) can also be reorganized in the following way

$$x_{(u,v)} \le \sum_{f \in F} \sum_{(n \to n) \in C_{st}^{\ f}} x_{(u,v)}^{f(n \to m)} + \sum_{f \in F} \sum_{(n \to n) \in C_{st}^{\ f}} x_{(v,u)}^{f(n \to m)} \quad (3)$$

$$x_{(u,v)} \ge x_{(u,v)}^{f(n \to m)}, \forall f \in F, \forall (n \to m) \in C_{st}^{f}$$

$$x_{(u,v)} \ge x_{(v,u)}^{f(n \to m)}, \forall f \in F, \forall (n \to m) \in C_{st}^{f}$$

- S_i^{fn} : {0,1}. 1 if service $n \in C_{st}^{f}$ is at node *i* for flow $f \in F$, and 0 otherwise.

- X_{ia}^{n} : {0,1}. 1 if service $n \in K$ is placed on node *i*, 0 otherwise. This is not a decision variable. In other words, it holds that

$$x_{ia}^{n} = \begin{cases} 1, & \text{if } \sum_{f \in F} S_{ia}^{fn} > 1, \forall n \in C^{f}, \forall i \in V, \forall a \in O_i \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

Also in this case the above equation can be reorganised in the following manner:

$$x_{ia}^{n} \le \sum_{f \in F} S_{ia}^{fn} > 1, \forall n \in C^{f}, \forall i \in V, \forall a \in O_i \quad (5)$$

$$x_{ia}^{n} \ge S_{ia}^{fn}, \forall n \in C^{f}, \forall i \in V, \forall a \in O_i$$

Detailed cost components

The RMA algorithm seeks to find the optimal placement of VNFs that minimizes the fragmentation of resources in the network, provided that the demands of the flows can be satisfied. The different cost components description is provided in detail below.

VNF Deployment Cost: This is the cost to deploy a VNF on a node, with the constraint as follows

$$C_{VNF} = \sum_{i \in V} \sum_{n \in K} \sum_{a \in O_i} h_i^{n} x_{ia}^{n} \quad (6)$$

Link Usage Fixed Cost: Cost incurred whenever a link is used by any of the flows. This cost is generally technology dependent.

$$C_f = \sum_{(u,v) \in E} k_{(u,v)}^{c} x_{(u,v)} \quad (7)$$

Link Usage Dynamic Cost: Dynamic cost is based on the amount of link resources used by flows. This is cost per unit of flow going through a link.

$$C_d = \sum_{(u,v) \in E} k_{(u,v)}^{d} \sum_{f \in F} \sum_{(n \to m) \in C_{st}^{f}} d^{f} x_{(u,v)}^{f(n \to m)} \quad (8)$$

The cost function to minimise was shown in Eq. (1). Furthermore, the terms edge and link are used interchangeably.

Optimization Constrains

Edge Capacity Constraint: Each edge has a capacity limit that must not be violated. This constraint makes surethat the sum of all the flows passing through an edge never exceeds the capacity of the link.

$$\sum_{f \in F} \sum_{(n \to m) \in C_{st}^f} d^f x_{(u,v)}^{f(n \to m)} \leq c(u,v) \qquad \forall (u,v) \in E \tag{9}$$

Flow Latency Constraint: Each flow has a latency constraint that must not be violated. A flow should not experience latency greater than its latency constraint.

$$\sum_{(n \to m) \in C_{st}^f} \sum_{(u,v) \in E} l(u,v) x_{(u,v)}^{f(n \to m)} \leq l^f \qquad \forall f \in F \tag{10}$$

Edge Constraint: Only edges which are present in the physical network are used when going from one service to another in the service chain.

$$x_{(u,v)}^{f(n \to m)} \leq w_{(u,v)} \qquad \forall (u,v) \in E, \quad (n \to m) \in C_{st}^f \tag{11}$$

Flow Constraint:$\forall_i \in V$, $(n \to m) \in C_{st}^f$where service m is after service n in the service chain C_{st}^f . Flow constraint ensures that there is a single continuous path between pair of nodes on which service n and m are placed. Here, n_{sf} , n_{tf} $\in C_{st}^f$ and they are placed at s^f and t^f, respectively. This ensures that flow path starts at node s^f and ends at node t^f.

$$\sum_{j \in V} x_{(i,j)}^{f(n \to m)} - \sum_{k \in V} x_{(k,i)}^{f(n \to m)} = \sum_{a \in O_i} S_{ia}^{fn} - \sum_{a \in O_i} S_{ia}^{fm} \tag{12}$$

Service Selection Constraint: Service $n \in C_{st}^f$ can only be selected on nodes that can host service n.

$$S_{ia}^{fn} \leq M_i^n \quad \forall n \in C_{st}^f, \quad \forall i \in V, \forall a \in O_i \tag{13}$$

Service Node Selection: Single node is selected to host a service in the service chain C_{st}^f of flow$f \in F$.

$$\sum_{i \in V} \sum_{a \in O_i} S_{ia}^{fn} = 1 \quad \forall n \in C_{st}^f, \quad \forall i \in V \tag{14}$$

Node Capacity Constraint: We can keep track of number of cores being used. Each free core at a node can host a single service. So number of services hosted at node is limited by the number of cores available at that node.

$$\sum_{n \in K} \sum_{a \in O_i} X_{ia}^n \leq |O_i| \quad \forall i \in V \tag{15}$$

Service Capacity Constraint: Each service deployed in an XPU node can only serve up to a certain flow demand. For instance, service n can serve 1000 Mbit/s, while service m can service 100 Mbit/s on a single core.

$$\sum_{f \in F} \sum_{n \in C^f} d^f S_{ia}^{fn} \leq U_n \quad \forall i \in V, \forall a \in O_i \tag{16}$$

Single VNF per core: This constraint makes sure that there can be only one VNF instance per core on a node.

$$\sum_{n \in K} X_{ia}^{n} \leq 1 \quad \forall i \in V, \forall a \in O_i \tag{17}$$

17.4.1.1 *Sub-Optimal Solution*

A heuristic algorithm for the PC-VNFP service is developed to obtain a sub-optimal allocation of flows withing an acceptable execution time ensuring optimized use of network and compute resources.

This heuristic is a greedy algorithm without using the ILP. This solution for PC-VNFP service also intends to minimize the overall cost function in terms of (i) **VNF placement**, and (ii) **Flow allocation**. Here, during the search for an optimal minimum cost path for jointly allocating a flow and placement of services, this solution considers only the first K minimum cost path instead of the all the possible paths. Finally, considering the several different constraints that stand for the PC-VNFP service one best option is chosen from those K candidate solutions. The details are described in Algorithm 13.

Algorithm 1: Sub-optimal solution for RMA PC-VNFP services

```
1  Get Network topology and capacity information;
2  Get XPU nodes capacity and availability;
3  Get S as the set of services to be deployed in the XPU nodes;
4  F ← set of flows to be allocated;
5  for for each flow f ∈ F do
6      Get s ← Source of flow f;
7      Get d ← Destination of flow f;
8      Get capacity and latency demand of the flow;
9      Get d ← Destination of flow f;
10     find K number of shortest paths from s to d ;
11     minimum_cost_path ← 0;
12     for for each path p ∈ K shortest path do
13         cost_path_p ← total cost of satisfying service chain and traversing the flow;
14         if cost_path_p < minimum_cost_path then
15             minimum_cost_path ← cost_path_p;
16         end
17     end
18     if minimum_cost_path! = 0 then
19         allocate the flow in the minimum cost path;
20     end
21     else
22         no suitable solution is found to satisfy the flow request;
23     end
24 end
```

Algorithm 13: Algorithm for a sub-optimal solution of RMA PC-VNFP service

18 References

[1] B. Fortz and M. Thorup, "Optimizing OSPF/IS-IS weights in a changing world," *IEEE journal on selected areas in communications,* vol. 20, pp. 756--767, 2002.

[2] 5G-Crosshaul, "IR5.4 – Experimental evaluation results," 5G-Crosshaul, 2017.

[3] 5G-Crosshaul, "D4.1 - Initial design of 5G-Crosshaul Applications and Algorithms," 2016.

[4] A. Pal, "Approximation algorithms for covering and packing problems on paths," Ph.D. dissertation, 2014.

[5] J. M. Kleinberg, "Approximation algorithms for disjoint paths problems," Ph.D. dissertation, 1996.

[6] K. Elbassioni and et al., "Approximation algorithms for the unsplittable flow probelm on paths and trees," *KIPIcs,* vol. 18, 2012.

[7] L. Epstein and et al., "Online capacitated interval coloring," *SIAM Journal on Discrete Mathematics,* vol. 23, no. 2, pp. 882-841, 2009.

[8] V. T. Chakaravarthy and et al., "Varying bandwidth resource allocation problem with bag constraints," in *IEEE IPDPS*, 2010.

[9] Y. Azar and O. Regev, "Combinatorial algorithms for the unsplitable flow problem," *Algorithmica,* vol. 44, no. 1, pp. 49-66, 2005.

[10] T. W. Kuo and et al., "Deploying chains of virtual network functions: On the relation between link and server usage," in *INFOCOM*, 2016.

[11] Y. Lin and et al., "Wireless network cloud: architecture and system requirements," *IBM Journal of Research and Development,* vol. 54, 2010.

[12] V. Suryaprakash and et al., "Are heterogeneous cloud-based radio access networks cost effective?," *IEEE Journal on Selected Areas of Communication,* vol. 33, no. 10, pp. 2239-2251, 2015.

[13] T. Wan and P. Ashwood-Smith, "A performance study of CPRI over Ethernet with IEEE 802.1Qbu and 802.1Qbv enhancements," in *IEEE Globecom*, 2015.

[14] U. Dotsch and et al., "Quantitative analysis of split base station processing and determination of advantageous architectures for LTE," *Bell Labs Tech. Journal,*

vol. 18, no. 1, pp. 105-128, 2013.

[15] D. Wubben and et al., "Benefits and impact of cloud computing on 5G signal processing: flexible centralization through cloud-RAN," *IEEE Signal Processing Magazine,* vol. 31, no. 6, pp. 35-44, 2014.

[16] C. Y. Chang and et al., "Impact of packetization and scheduling on C-RAN fronthaul performance," in *IEEE Globecom*, 2016.

[17] N. E. Khoury, S. Ayoubi and C. Assi, "Energy-Aware Placement and Scheduling of Network Traffic Flows with Deadlines on Virtual Network Functions," in *IEEE CloudNet*, 2016.

[18] S. Tadesse, C. Casetti, C. Chiasserini and G. Landi, "Energy-efficient Traffic Allocation in SDN-based Backhaul Networks: Theory and Implementation," in *IEEE CCNC*, 2017.

[19] S. Tadesse, C. F. Chiasserini and F. Malandrino, "Energy Consumption Measurements in Docker," in *IEEE COMPSAC*, 2017.

[20] Intel, "Core i7-7700 Datasheet," [Online]. Available: https://ark.intel.com/products/97128/Intel-Core-i7-7700-Processor-8M-Cache-up-to-4 20-GHz..

[21] N. E. Khoury, S. Ayoubi and C. Assi, "Energy-Aware Placement and Scheduling of Network Traffic Flows with Deadlines on Virtual Network Functions," in *IEEE CloudNet*, 2016.

[22] Cisco Systems Inc, "Cisco visual networking index global mobile data traffic forecast update 2013-2018," White Paper, Feb. 2013.

[23] A. S. G. Andrae and T. Edler, "On global electricity usage of communication technology: Trends to 2030," *Challenges,* vol. 6, p. 117, 2015.

[24] Mohammed H. Alsharif, Rosdiadee Nordin, and Mahamod Ismail, "Survey of Green Radio Communications Networks: Techniques and Recent Advances," *Journal of Computer Networks and Communications,* vol. 2013, p. 13, 2013.

[25] P. Dini, M. Miozzo, N. Bui and N. Baldo, "A Model to Analyze the Energy Savings of Base Station Sleep Mode in LTE HetNets," in *EEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing*, Beijing, 2013.

[26] H. Al Haj Hassan, L. Nuaymi, and A. Pelov, "Renewable energy in cellular networks: A survey," in *IEEE Online Conference on Green Communications*

(GreenCom), Oct 2013.

[27] G. Lee, W. Saad, M. Bennis, A. Mehbodniya, and F. Adachi, "Online ski rental for scheduling self-powered, energy harvesting small base stations," in *IEEE International Conference on Communications (ICC)*, Kuala Lumpur, Malaysia, May 2016.

[28] M. Mendil, A. D. Domenico, V. Heiries, R. Caire, and N. Hadjsaid, "Fuzzy Q-Learning based Energy Management of Small Cells Powered by the Smart Grid," in *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (IEEE PIMRC)*, Valencia, Spain, 2016.

[29] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, MIT Press, 1998.

[30] R. A. C. Bianchi, M. F. Martins, C. H. C. Ribeiro, and A. H. R. Costa, *IEEE Transactions on Cybernetics,* vol. 44, no. 2, pp. 252-256, Feb. 2014.

[31] C. M. Bishop, Pattern Recognition and Machine Learning (Information Science and Statistics), New York: Springer-Verla, 2006.

[32] Yaser S. Abu-Mostafa, Malik Magdon-Ismail, and Hsuan-Tien Lin, Learning from Data, AMLBook, 2012.

[33] M. Miozzo, L. Giupponi, M. Rossi, P. Dini, "Switch-On/Off Policies for Energy Harvesting Small Cells through Distributed Q-Learning," in *2nd IEEE WCNC workshop on Green and Sustainable 5G Wireless Networks (GRASNET 2)*, San Francisco (CA), USA, 2017.

[34] J. Baranda, M. Miozzo, P. Dini, J. Núñez, J. Mangues, "Backhaul Routing and Base Station Sleep Mode Engagement in Energy Harvesting Cellular Networks," in *International Conference on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWIM)*, Malta, 2016.

[35] The Network Simulator 3 (ns3), [Online]. Available: https://www.nsnam.org/.

[36] EU EARTH, "D2.3: Energy Efficiency analysis of the reference systems, areas of improvements and target breakdown," Deliverable D2.3, www.ict-earth.eu, 2010.

[37] 5G-Crosshaul, "D5.1 - Report on validation and demonstration plans," Oct 2016.

[38] "Lithium Batteries Failures".*available on-line at: http://www.mpoweruk.com/lithium_failures.htm.*

[39] M. Miozzo, D. Zordan, P. Dini, M. Rossi, "SolarStat: Modeling Photovoltaic Sources through Stochastic Markov Processes," in *IEEE Energy Conference (ENERGYCON)*, Dubrovnik, Croatia, 2014.

[40] M. Zeng, W. Fang, J. Rodrigues and Z. Zhu, "Orchestrating multicast oriented NFV trees in inter-DC elastic optical networks," in *ICC*, 2016.

[41] W. Fang, "Joint spectrum and IT resource allocation for efficient vNF service chaining in inter-datacenter elastic optical networks," *IEEE Commun. Lett.,* vol. 20, pp. 1539-1542, 2016.

[42] H. Moens and F. De Turck, "VNF-P: A model for efficient placementof virtualized network functions," in *CNSM*, 2014.

[43] S. Clayman, E. Maini, A. Galis, A. Manzalini and N. Mazzocca, "The dynamic placement of virtual network functions," in *NOMS*, 2014.

[44] S. Mehraghdam, M. Keller and H. Karl, "Specifying and placing chains of virtual network functions," in *CloudNet*, 2015.

[45] M. Xia, "Network function placement for NFV chaining in packet/optical datacenters," *Journal of Lightwave Technology,* vol. 33, pp. 1565-1570, 2015.

[46] S. Zhang, Q. Zhang, H. Bannazadeh and A. Leon-Garcia, "Routing algorithms for network function virtualization enabled multicast topology on SDN," *IEEE Trans. Netw. Serv. Manag.,* vol. 12, pp. 580-594, 2015.

[47] M. Zeng, W. Fang and Z. Zhu, "Orchestrating tree-type VNF forwarding graphs in inter-DC elastic optical networks," *Journal of Lightwave Technology,* vol. 33, pp. 1565-1570, 2015.

[48] Y. Wang, P. Lu, W. Lu and Z. Zhu, "Cost-Efficient Virtual Network Function Graph (vNFG) Provisioning in Multi-Domain Elastic Optical Networks," *Journal of Lightwave Technology,* 2017.

[49] Net2plan, "The open-source network planner," online http://www.net2plan.com/.

[50] A. Castro, L. Velasco, C. Chen, J. Yin, Z. Zhu, R. Proietti and S. Ben Yoo, "Brokered Orchestration for End-to-End Service Provisioning across Heterogeneous Multi-Operator (Multi-AS) Optical Networks," *Journal of Lightwave Technology ,* vol. 34, no. 34, 2016.

[51] S. Ben Yoo, "Multi-domain Cognitive Optical Software Defined Networks with Market-Driven Brokers," in *ECOC Conference*, 2014.

[52] R. Vilalta and e. al., "The need for a control orchestration protocol in research projects on optical networking," in *EUCNC*, 2015.

[53] R. Vilalta, A. Mayoral, J. Baranda, J. Nuñez, R. Casellas, R. Martínez, J. Mangues and R. Muñoz, "Hierarchical SDN orchestration of wireless and optical networks with E2E provisioning and recovery for future 5G networks," in *OFC*, 2016.

[54] R. Casellas and e. al., "SDN Orchestration of OpenFlow and GMPLS Flexi-Grid Networks With a Stateful Hierarchical PCE [Invited]," *Journal of Optical Communications and Networking,* vol. 7, no. 1, pp. a106-a117, 2015.

[55] X. Cao, N. yoshikane, T. Tsuritani, I. Morita, M. Shiraiwa and N. Waad, "Functional Service Design with SDN Orchestration across Heterogeneous Multi-domain Networks," in *OFC*, 2016.

[56] L. Cominardi, F. Giust, C. J. Bernardos and A. de la Oliva, "Distributed mobility management solutions for next mobile network architectures," *Computer Networks,* no. 121, pp. 124-136, 2017.

[57] F. Giust, C. J. Bernardos and A. de la Oliva, "Analytic Evaluation and Experimental Validation of a Network-Based IPv6 Distributed Mobility Management Solution," *IEEE Transactions on Mobile Computing,* vol. 13, no. 11, pp. 2484-2497, November 2014.

[58] K. Alexandris, N. Nikaein, R. Knopp and C. Bonn, "Analyzing X2 Handover in LTE/LTE-A," in *Wireless Networks: Measurements and Experimentation (WINMEE)*, 2012.

[59] L. Zhang, T. Okamawari, T. Fujii, "Experimental Analysis of TCP and UDP during LTE Handover," in *IEEE Vehicular Technology Conference (VTC Spring)*, 2012.

[60] M. P. Wylie-Green and T. Svensson, "Throughput, capacity, handover and latency performance in a 3GPP LTE FDD field trial," in *IEEE Global Telecommunications Conference (GLOBECOM)*, 2010.

[61] M. S. Pan, T. M. Lin, W. T. Chen, "An Enhanced Handover Scheme for Mobile Relays in LTE-A HighSpeed Rail Networks," *IEEE Transactions on Vehicular Technology,* 2015.

[62] C. Bouras, P. Ntarzanos and A. Papazois, "Cost modeling for SDN/NFV based mobile 5G networks," in *IEEE ICUMT*, 2016.

[63] T. M. Knoll, "Life-cycle cost modelling for NFV/SDN based mobile networks," in *IEEE CTTE*, 2015.

[64] B. Naudts et al,, "Techno-economic Analysis of Software Defined Networking as Architecture for the Virtualization of a Mobile Network," in *IEEE EWSDN*, 2012.

[65] Juniper Networks, Inc., "The Elastic CDN Solution," Juniper Networks, Inc., 2014.

[66] "Ryu Controller," Available online at https://osrg.github.io/ryu/.

[67] 5G-Crosshaul, "D5.2 - Report on validation and demonstration results," Dec 2017.

[68] IETF, Network Working Group, "RFC 1112, Host Extensions for IP Multicasting".

[69] IETF, Network Working Group, "RFC 2236, Internet Group Management Protocol, Version 2".

[70] IETF, Network Working Group, "RFC 3376, Internet Group Management Protocol, Version 3".

[71] IETF, Network Working Group, "RFC 4604, Using Internet Group Management Protocol Version 3 (IGMPv3) and Multicast Listener Discovery Protocol Version 2 (MLDv2) for Source-Specific Multicast".

[72] 5G-Crosshaul, "D1.1 - Initial specification of the system," Jun 2016.

[73] "5G-NORMA Project," [Online]. Available: https://5g-ppp.eu/5g-norma/.

[74] 5G-NORMA, "Deliverable D4.1 RAN architecture components – intermediate report," Nov 2016.

[75] 5G-NORMA, "Deliverable D5.1 Definition of connectivity and QoE/QoS management mechanisms – intermediate report," Nov 2016.

[76] "5GEx Project," [Online]. Available: https://5g-ppp.eu/5gex/.

[77] 5GEx, "Deliverable 2.1, 5GEx Initial System Requirements and Architecture (public version)," December 2016.

[78] L. Contreras, C. Bernardos, A. de la Oliva, X. Costa-Pérez and R. Guerzoni, "Orchestration of Crosshaul Slices from Separated Administrative Domains," in *EuCNC*, 2016.

[79] L. Contreras, C. Bernardos, A. de la Oliva and X. Costa-Pérez, "Sharing of Crosshaul Networks via a Multi-Domain Exchange Environment for 5G Services," in *IEEE International Workshop on Multi-Provider Network Slicing and Virtualization (MPNSV) 2017, co-located with NetSoft 2017*, 2017.

[80] Lagopus Project, Lagopus Software Switch, [On line]. Available: http://lagopus.org.

[81] N. Gazit, F. Malandrino and D. Hay, "Competition between network operators and content providers in SDN/NFV core networks," in *IEEE INFOCOM SWFAN Workshop*, 2016.

[82] J. Mattingley and S. Boyd, "CVXGEN: A code generator for embedded convex optimization," in *Optimization and Engineering*, 2002.

[83] J. G. Herrera and J. F. Botero, "Resource allocation in NFV: A comprehensive survey," *IEEE Transactions on Network and Service Management,* 2016.

[84] A. Mayoral, R. Vilalta, R. Muñoz, R. Martínez and R. L. V. Casellas, "The need for a Transport API in 5G for Global Orchestration of Cloud and Networks Through a Virtualized Infrastructure Manager and Planner," *Journal of Optical Communications and Networking,* vol. 9, no. 1, 2017.

[85] A. Mohammadkhan, S. Ghapani and G. Liu, Virtual function placement and traffic steering in flexible and dynamic software defined networks, 21st IEEE International Workshop on Local and Metropolitan Area, 2015, pp. 1-6.

[86] 5G-Crosshaul, "IR4.3 - Evaluation Experiments," 2017.

[87] 5G-Crosshaul, "D2.2 - Integration of physical and link layer technologies in 5G-Crosshaul network nodes".