DOI: https://doi.org/10.1109/JSAC.2018.2874142

Joint Optimization of Edge Computing Architectures and Radio Access Networks

Andres Garcia-Saavedra, George Iosifidis, Xavier Costa-Perez, Douglas J. Leith

***Abstract*—Virtualized Radio Access Network (vRAN) architectures and Multiple-access Edge Computing (MEC) systems constitute two key solutions for the emerging Tactile Internet applications and the increasing mobile data traffic. Their efficient deployment however requires a careful design tailored to the available network resources and user demand. In this paper, we propose a novel modeling approach and a rigorous analytical framework, MvRAN, that minimizes vRAN costs and maximizes MEC performance. Our framework selects jointly the base station function splits, the fronthaul routing paths, and the placement of MEC functions. We follow a data-driven evaluation method, using topologies of 3 operational networks and experiments with a typical face-recognition MEC service. Our results reveal that MvRAN achieves significant cost savings (up to 2.5 times) compared to non-optimized C-RAN or D-RAN systems, and that MEC pushes the vRAN functions to RUs and hence can increase substantially the network cost.**

***Index Terms*—5G, Tactile Internet, Edge Computing, Cloud RAN, Next Generation Fronthaul, Backhaul, Crosshaul**

I. Introduction

The fast increasing mobile data traffic on the one hand, and the stringent requirements of emerging services on the other, have spurred numerous efforts for the redesign of Radio Access Networks (RAN) in 5G systems. To this end, the deployment of *virtualized* RAN architectures (vRAN) and Multiple-access Edge Computing (MEC) platforms, promise to add the much-needed flexibility and intelligence at the network edge. However, the successful adoption of these solutions is an intricate problem that brings novel theoretical and practical issues that remain unresolved.

A. Background and Motivation

There has been remarkable progress in the design of RAN architectures in the last few years. The concept of *centralized* RAN (C-RAN) suggests the relocation of Base Station (BS) functions from low-cost Radio Units (RUs) to a central unit (CU) [1]. In C-RAN, the RUs perform only basic RF processing (up/down conversion, amplification, etc.) and exchange digitized (I/Q) radio samples with the CU via the *fronthaul* network. This architecture promises to reduce costs [2]–[4], and improve spectrum efficiency through the centralized control of tasks such as interference management [5]. Unfortunately, the latency and bandwidth requirements that C-RAN imposes onto the fronthaul network are hard to satisfy, and often require huge investments [3], [6].

More recently, a flexible design approach has been suggested, where a (sub)set of the BS functions are centralized based on the available network resources [3], [7]–[9]. This architecture can relax the requirements of C-RAN when needed, and several industry groups are currently working for its standardization [10], [11]. This idea builds upon the *softwarization* of RAN, which enables the flexible selection of the centralization degree (*function split*) of each BS, and the term *virtual* RAN (vRAN) has been coined to describe it. This fine-grained network design approach is considered very promising for 5G systems [10]–[12].

However, designing vRANs is a challenging problem and we currently lack a systematic methodology to solve it. At the core of this problem lie the decisions for selecting the function splits *and* the CUs-RUs routing paths which, clearly, should be jointly devised. On the one hand, the optimal function split decision for each BS depends on the capacity and latency of the route that connects RUs with the CU. On the other hand, selecting the optimal route requires information about the volume and latency needs of the RU-CU data flow, which both depend on the selected function split. *Due to the multiple vRAN split choices, this coupling makes a traditionally challenging routing problem even harder.*

The vRAN design problem is further compounded when the RAN needs to accommodate Multi-access Edge Computing (MEC) services. MEC is among the key enablers of 5G [13], and is considered instrumental for the successful support of emerging Tactile Internet applications [14]. For example, a MEC platform deployed at the RUs can effectively enable the real-time control of vehicles, a service that requires ultra-low latency communications; or, it can support a mobile face-recognition application by filtering its voluminous video streams at the edge, hence reducing network congestion.

Clearly, there is an intricate coupling between the design of vRAN and deployment of MEC services. Namely, in order to host a MEC service an RU has to implement a full-stack BS[1] (host all BS functions) [15]; and this constraints its eligible function splits. Conversely, placing all radio functions at the CU (C-RAN) provides cost gains (resource pooling) but consumes high fronthaul bandwidth. Therefore, it constraints the deployment of throughput-hungry MEC services (such as video analytics) which, for such C-RAN configurations, can only be deployed at the CU (or further in the network core). These decisions for the different BSs are intertwined as they share the same network links and compute nodes.

Besides, MEC applications can be very diverse; some create very large data flows (e.g., video streaming for surveillance),

Andres Garcia-Saavedra and Xavier Costa-Perez are with NEC Laboratories Europe, Heidelberg, Germany. George Iosifidis and Douglas J. Leith are with the School of Computer Science and Statistics, Trinity College Dublin, Ireland. G. Iosifidis acknowledges the support from the Science Foundation Ireland (SFI) under Grant Number 17/CDA/4760.

[1] In this work we focus on the challenging and most typical scenario where MEC applications interact with users' data plane, e.g., compute-intensive offloading, automotive or video caching.

while others have ultra-low latency needs, and others have small flows but high computing load. Therefore, the problem of deciding *(i)* where to place the various MEC applications, *(ii)* how to select the function split for each BS, *(iii)* and how to route the legacy and MEC traffic between RUs and CUs, is as important as challenging to solve.

B. Methodology and Contributions

We introduce a rigorous analytical framework for the deployment of MEC services and the design of vRAN. We treat this as a unified problem and propose a joint optimization approach that determines the function splits, the MEC placement, and the routing of (both legacy and MEC) data among the RUs and CUs. We model the BS operation as a *chain of functions* that successively process the user traffic, and the MEC service as an additional function running on top of them. The function implementation has certain memory requirements and induces a computing cost that may vary across RUs and CUs; and similarly the selected routing path affects the data transfer costs. Our framework optimizes a weighted objective of network expenditures and MEC delay, and allows different operators to prioritize differently these criteria. We solve the formulated problems using Benders decomposition algorithm, which progresses in iterations and reaches eventually the exact solution. We also find that, for the problems at hand, the algorithm converges in satisfactory time.

In order to obtain practical insights we study an important MEC face recognition service, using the open-source software OpenFace [16]. We measure its compute and memory requirements and how the request rate (or, load) affects the service delay. Furthermore, we analyze the properties (structure, link capacities, etc.) of three real RANs in different countries. We then employ our optimization framework to design the vRAN architecture for these networks, assuming they need to support the computing-hungry OpenFace application. We use measurement-based values for the system parameters, and conduct a thorough sensitivity analysis to asses their impact on the network costs and the MEC delay.

Our contributions can be thus summarized as follows:

- *Joint vRAN and MEC Design*. To the best of our knowledge, this is the first work introducing an analytical framework for the joint design of vRAN and MEC architectures. We propose a detailed model that accounts for the chaining of functions, the coupling among MEC and vRAN, and the network capacity constraints. Our framework is generic, and we explain how it can be used for different types of MEC services, including Tactile Internet applications with ultra-low delay needs.
- *Exact Optimization Framework*. We characterize the complexity of the arising optimization problem (NP-hard) and design an intuitive algorithm that provably yields its exact solution. We use Benders' decomposition which separates the problem into the *routing* and *network configuration* sub-problems. Our method is practical as it facilitates the solution of large problem instances, without compromises in optimality.
- *Evaluation Using Real Networks and OpenFace*. We apply our framework to actual RAN topologies, using a popular MEC service that we have *profiled* and 3GPP specs. We show that each network requires a different vRAN and MEC configuration, and study how this is affected by key system parameters. Our analysis reveals that, in these realistic conditions, the joint and flexible MEC-vRAN design provides significant benefits compared to fully centralized (C-RAN) and fully decentralized (D-RAN) architectures.

Paper organization. §II introduces the model and the problem. §III provides the problem formulation and §IV the solution algorithm. §V presents the experimental profiling of OpenFace, and analyzes the RAN data. §VI presents a thorough data-driven evaluation of our optimization framework. We review the literature in §VII and conclude in §VIII.

II. System Model and Problem Definition

A. Model and Preliminaries

For the scope of vRAN-MEC design the key splits are those in Fig. 1 [10]. Split 1 is the full-stack BS implementation at RUs (D-RAN). Split 2 enables the centralization of L3 and L2 operations and enhances mobility management. Split 3 enables higher resource pooling and multi-cell coordination but has tight delay bounds (for data transfers among the function locations) and introduces data overhead. Finally, split 4 (C-RAN) consumes very high bandwidth (load-independent), has very low delay bounds, but maximizes spectrum efficiency and hardware utilization [8]. Based on the latest proposals [8], [10], [11] we consider packet-based shared links in the fronthaul.

Demand. We focus on the uplink,[2] but our study can easily be extended for downlink. We model the demand from users associated to BS n as follows: there are inelastic MEC services (henceforth, *type* a) with hard delay requirements such as those arising in Tactile Internet [14] ($\leq$1ms); and throughput-hungry delay-elastic MEC services (henceforth, *type* b) for which we minimize the delay.[3] Type a requests of the users attached to BS n are created by a process[4] $\{M_n^a(t)\}_{t=1}^{\infty}$ with values upper bounded by μ_n^a (requests/s), creating a maximum (peak) data flow $\lambda_n^a = \lambda_a \mu_n^a$ (Mb/s) that must be routed from RU n towards the MEC service location. Similarly, for type b requests from users of BS n it is $\{M_n^b(t)\}_{t=1}^{\infty}$, with maximum value of μ_n^b, and peak load $\lambda_n^b = \lambda_b \mu_n^b$. The network supports also *legacy* user traffic (e.g., mobile broadband) that must be routed from each RU n to the CU (and then to the mobile core/Internet; out of our scope). The **total** traffic created by the users that are associated with BS n is denoted λ_n (Mb/s).

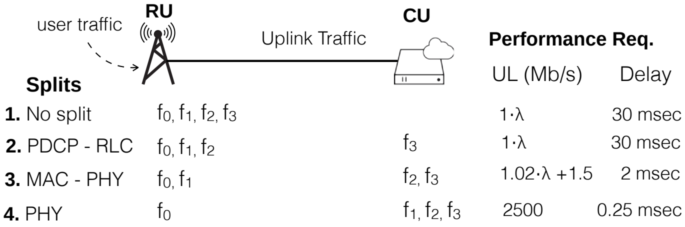
MEC and Radio Functions. The BS operation is a chain of functions [10]: f_0 corresponds to basic radio tasks (analog processing, etc.) and is placed at RUs. Assuming LTE, f_3 corresponds to PDCP, f_2 to (RLC and MAC) and f_1 models all PHY functions not in f_0. The deployment of these functions sets the delay-bandwidth requirements between the CU and RUs, as explained in Fig. 1. Similarly, functions f_4^a and f_4^b correspond to MEC services of type a and b, respectively; and can be deployed at CUs or the RUs with full-stack implementation, i.e., hosting functions f_0 up to f_3.

The execution of each function requires a certain number of *processing cycles* per unit of traffic; and hence each split

[2] Uplink is important as MEC services often process user-generated traffic.

[3] We focus on delay-sensitive services here, but our approach can be applied to different KPIs, using any mix of delay, throughput, and other criteria.

[4] Since this is a network design problem one can use the peak loads, as we do, or any expression of their mean values (see [2]. This depends on how (in)elastic the service is, i.e., whether admission control is employed or not.



	RU	CU	Performance Req.	
Splits			UL (Mb/s)	Delay
1. No split	f_0, f_1, f_2, f_3		$1\cdot\lambda$	30 msec
2. PDCP - RLC	f_0, f_1, f_2	f_3	$1\cdot\lambda$	30 msec
3. MAC - PHY	f_0, f_1	f_2, f_3	$1.02\cdot\lambda +1.5$	2 msec
4. PHY	f_0	f_1, f_2, f_3	2500	0.25 msec

Fig. 1: Bandwidth and latency requirements of key splits [10]; functions are *chained*; λ is the total traffic from the users of this BS. The case of *no split* is also shown (referred to as split 1 to facilitate discussion), and corresponds to the **Decentralized RAN (D-RAN)**. The MEC services are placed at the CU, or RU if D-RAN is selected.

induces a different processing load to the RUs and CU. Let us denote this load (cycles per Mb/s) with $\rho_{i,r}$ and $\rho_{i,c}$, for the RU and CU respectively, when split $i \in \{1, 2, 3, 4\}$ is selected. We also denote with ρ_a and ρ_b the processing load for MEC services of type a and b, respectively. The execution of functions requires also memory, and we denote with $\tau_{i,r}$ and $\tau_{i,c}$ the total memory needs (MB per request/s) for each split i (for the RU and CU, respectively); and with τ_a and τ_b the memory needs of MEC functions. Section V presents the measurements we performed and data we used to quantify these parameters.

Network and Servers. We consider a RAN with a set $\mathcal{N}$ of N RUs (or, BSs) and 1 CU.[5] These are connected with a packet-based fronthaul network[6] $G = (\mathcal{I}, \mathcal{E})$, where $\mathcal{I}$ is the superset of routers, CU (node 0) and RUs; and $\mathcal{E}$ the set of links connecting these elements. Each link $(i, j) \in \mathcal{E}$ has average capacity c_{ij} (Mb/s), and delay d_{ij} (secs). Let

$$p := \{(n, i_1), (i_1, i_2), \dots, (i_k, 0) : (i, j) \in \mathcal{E}\},$$

denote a path from RU n to CU; $\mathcal{P}_n$ is the set of all paths from RU n to the CU, and we define $\mathcal{P} = \cup_{n=1}^{N} \mathcal{P}_n$. Each $p \in \mathcal{P}$ is described by the total delay d_p of its links. RU n has processing capacity P_n (cycles/s) and memory T_n (MB); and the respective quantities for the CU are P_0 and T_0. The requirements of functions deployed at each location should not exceed these processing and memory capacities.

Costs. There is an average data transfer cost due to leasing costs, equipment utilization, etc. in the network. We denote γ_p the cost for path p (per Mb/s) and define the *routing cost vector* $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, \dots, \gamma_{|\mathcal{P}|})$. In vRAN f_2 and f_1 can be implemented in virtual machines (VMs). The cost for initiating and using a VM depends on the hardware. CUs are in central facilities and use high-end servers; hence this *cost will be lower compared to RUs* [2]. We denote with α_n (monetary units) the average cost for instantiating a VM in RU n (due to cooling, leasing fees, etc.), with β_n the average cost for serving each request (monetary units per cycle); and define the respective parameters α_0, β_0 for CU [18]. The *computing cost* vectors are $\boldsymbol{\alpha} = (\alpha_0, \alpha_1, \dots, \alpha_N)$ and $\boldsymbol{\beta} = (\beta_0, \beta_1, \dots, \beta_N)$.

B. Trade Offs and Problem Definition

The objective of the operator is to select the vRAN-MEC configuration that maximizes the MEC performance while minimizing the network costs. We consider here the delay as the key performance criterion since it affects the vast majority of MEC services; but additional factors might be relevant in other scenarios (and we later explain how our framework can include them). The cost comprises computing and routing expenses [2], [4] and since this is a network design problem we consider their maximum values (see [2]). Clearly there is an inherent tension between the function centralization that the network would prefer (as CUs are more cost-efficient), and the MEC requests the performance of which is typically maximized when they are satisfied at the very edge. The exact relationship of these two objectives depends on the network architecture and the available link, compute and memory resources.

Nevertheless, while the coupling between MEC and vRAN designs has been recently discussed, e.g., see [19], [20], the consideration of performance and cost criteria raises several hitherto unexplored trade-offs. On the one hand, placing the radio functions ($f_0 - f_3$) at RUs reduces the fronthaul network load and hence the routing costs. On the other hand, aggregating the vRAN functions at the CU reduces computing costs and offers centralized control that can improve the network's performance, e.g., through sophisticated interference management. However, some splits have very tight delay constraints and create high fronthaul traffic, and the CU might not have enough computation power to support all BSs.

Furthermore, the execution of MEC services at RUs can be beneficial because: *(i)* the requests are satisfied in proximity with users and hence possibly with low delay, and *(ii)* the RUs *filter* the MEC traffic and hence reduce the fronthaul network congestion. On the other hand, MEC execution at CU might in practice ensure faster execution if the fronthaul network is well-provisioned and the CU has much larger processing capacity than the RUs. Besides, execution at the CU can improve, in some cases, the MEC performance due to centralization. Consider for example, an Artificial Intelligence application that achieves higher precision when information from multiple BSs is jointly processed.

To account for all the above aspects, we introduce a generic optimization framework for the joint vRAN-MEC design:

MEC-vRAN Design Problem (MvRAN): Given the anticipated legacy and MEC demand $\boldsymbol{\lambda}$; a network $G = (\mathcal{I}, \mathcal{E})$ with link capacities and delays; computing and routing costs, $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$, decide *(i)* where to deploy the radio functions f_0, f_1, f_2, f_3; and the MEC services f_4^a and f_4^b; and *(ii)* how to route the traffic among the CUs and the RUs, in order to maximize the MEC performance and minimize the network costs.

III. MEC and vRAN Design

Network Configuration. We introduce the binary *configuration* variables x_{1n}, x_{2n}, x_{3n}, x_{4n}, which determine whether functions f_1, f_2, f_3 will be deployed at RU n ($x_{1n} = 1$); or function f_3 will be moved to CU ($x_{2n} = 1$); or both f_2 and f_3 will be moved to CU ($x_{3n} = 1$); or, we will place all BS functions at the CU ($x_{4n} = 1$). We also define the matrix:

$$\boldsymbol{x} = \big(\boldsymbol{x}_n = (x_{1n}, x_{2n}, x_{3n}, x_{4n}) : n \in \mathcal{N}\big).$$

[5] Single-CU systems are the most common in practice [17] and besides the RU/CU assignment is decided a priori. Hence our framework can be directly extended to multiple CUs.

[6] Since we consider diverse scenarios including fronthaul flows (in C-RAN split), backhaul flows (D-RAN) and hybrid flows (a.k.a. *midhaul, crosshaul, xhaul*), we will use either of these terms for the RU-CU traffic. Similarly, we use the terms fronthaul, backhaul or RAN for the RU-CU network.

The configuration decisions account for the function chaining constraints which dictate that f_1 (f_2) cannot be deployed at the CU unless f_3 and f_2 (f_3) are also placed there. Clearly, it is necessary to select exactly one configuration, hence:

$$x_{1n}+x_{2n}+x_{3n}+x_{4n}=1,\ \ \forall\, n\in\mathcal{N}\,. \qquad (1)$$

MEC placement. The *placement* of MEC type a and type b services is determined by the variables y_n^a and y_n^b for RU n, respectively; which are equal to one if the services are deployed at RU, and zero if placed at the CU. We define:

$$\boldsymbol{y}_a=(y_n^a\in\{0,1\}:n\in\mathcal{N}),\ \ \boldsymbol{y}_b=(y_n^b\in\{0,1\}:n\in\mathcal{N})\,.$$

These decisions are constrained by the vRAN setting, as a MEC service can be placed only after PDCP location. Hence,

$$y_n^a\le x_{1n},\quad y_n^b\le x_{1n},\ \ \forall\, n\in\mathcal{N}. \qquad (2)$$

Routing decisions. We let $z_p^{(n)}$ denote the traffic (Mb/s) routed over path $p\in\mathcal{P}_n$ (from RU n), and define the routing matrix $\boldsymbol{z}{=}(z_p^{(n)}, p{\in}\mathcal{P}, n{\in}\mathcal{N})$. Routing choices must satisfy:

$$\sum_{n\in\mathcal{N}}\sum_{p\in\mathcal{P}_n} z_p^{(n)} I_p^{ij}\le c_{ij},\ \forall\,(i,j)\in\mathcal{E},$$

with $I_p^{ij}\in\{0,1\}$ indicating if link (i,j) belongs to path p. We consider a packet-based network [8], [10] and hence multiple paths can be selected for every RU, as long as it holds:

$$\sum_{p\in\mathcal{P}_n} z_p^{(n)}=S_n,\ \ \forall\, n\in\mathcal{N}\,,$$

where S_n (Mb/s) is the flow that RU n sends to CU and depends on demand and the network configuration:

$$\begin{aligned}S_n(\boldsymbol{x}_n,y_n^a,y_n^b)=&-y_n^a\lambda_n^a-y_n^b\lambda_n^b+\lambda_n x_{1n}+\lambda_n x_{2n}\\&+(1.02\lambda_n+1.5)x_{3n}+2500x_{4n},\end{aligned}$$

and we refer the reader to Fig. 1 and (1)-(2) for the derivation of this expression. Note that when all BS functions are deployed at CU (x_{4n}=1) the flow is independent of λ_n.

Delay constraints. The flow $z_p^{(n)}$ cannot be non-zero if the delay d_p of path p exceeds the threshold required by the split that has been selected for RU n [10]. Hence, the configuration decisions $\boldsymbol{x}$ determine which paths are eligible for each RU. To capture this dependency, we partition the set of paths as follows: set $\mathcal{P}_n^{(2)}\subseteq\mathcal{P}_n$ of paths with delay larger[7] than 30msec; set $\mathcal{P}_n^{(3)}\subseteq\mathcal{P}_n$ of paths with delay larger than 2msec; and set $\mathcal{P}_n^{(4)}\subseteq\mathcal{P}_n$ of paths with delay larger than 0.25msec. Obviously, it holds $\mathcal{P}_n^{(2)}\subseteq\mathcal{P}_n^{(3)}\subseteq\mathcal{P}_n^{(4)}$. Then, for split 2 ($x_{2n}=1$) all flows in $\mathcal{P}_n^{(2)}$ should be set to zero, for split 3 ($x_{3n}=1$) set to zero all flows in $\mathcal{P}_n^{(3)}$, and for split 4 ($x_{4n}=1$) those in paths of $\mathcal{P}_n^{(4)}$. Also it is $\mathcal{P}_n^{(1)}=\mathcal{P}_n^{(2)}$.

Objective. The goal of the network operator is to minimize its costs and the delay the MEC services experience. To achieve this, we use a standard *scalarization* approach that ensures that we will obtain a Pareto optimal solution.

In detail, for routing costs we consider a linear function:

$$U_p(z_p^{(n)})=\gamma_p\sum_{n\in\mathcal{N}} z_p^{(n)},\ \ p\in\mathcal{P}\,. \qquad (3)$$

The computing costs depend on the vRAN configuration and the placement of the MEC functions. In particular we define the aggregate computing cost function for each RU n:

$$\begin{aligned}V_n(\boldsymbol{x}_n,y_n^a,y_n^b)=&\,y_n^b(\rho_b\lambda_n^b\beta_n+\alpha_n)+y_n^a(\rho_a\lambda_n^a\beta_n+\alpha_n)\\&+\alpha_n+\sum_{i=1}^{4}x_{in}\rho_{i,r}\lambda_n\beta_n\,. \qquad (4)\end{aligned}$$

The cost incurred by the CU for the needs of each BS n is:

$$\begin{aligned}V_{0n}(\boldsymbol{x}_n,y_n^a,y_n^b)=&\sum_{i=1}^{4}x_{in}(\rho_{i,c}\lambda_n\beta_0+\alpha_0) \qquad (5)\\&+(1-y_n^a)(\rho_a\lambda_n^a\beta_0+\alpha_0)+(1-y_n^b)(\rho_b\lambda_n^b\beta_0+\alpha_0)\end{aligned}$$

where the last terms indicate that when an RU does not implement a function this load is shifted to CU. Therefore the overall processing cost in the system is:

$$V(\boldsymbol{x},\boldsymbol{y})=\sum_{n=1}^{N}V_n(\boldsymbol{x}_n,y_n^a,y_n^b)+V_{0n}(\boldsymbol{x}_n,y_n^a,y_n^b), \qquad (6)$$

where $\boldsymbol{y}$ collects all the MEC placement decisions.

The delay that a MEC request experiences depends on the maximum routing delay d_{p_n} of the paths in $\mathcal{P}_n$, and the processing delay. The latter is constant for low demand, but can increase with the load for services with high CPU load, as in the case of OpenFace (see Fig. 3). Using the experiments we have conducted (explained in Sec. V) we use the following model for the delay:

$$\begin{aligned}D_n^i(y_n^i)=&(1-y_n^i)\lambda_n^i d_{p_n}+ \qquad (7)\\&\delta_1\left(\lambda_n^i y_n^i\left(\frac{\rho}{P_n}\right)+\lambda_n^i(1-y_n^i)\left(\frac{\rho}{P_0}\right)\right)+\\&\delta_2\left(\lambda_n^i y_n^i\left(\frac{\rho}{P_n}\right)+\lambda_n^i(1-y_n^i)\left(\frac{\rho}{P_0}\right)\right)^2\end{aligned}$$

where $i=\{a,b\}$ (the two MEC types), and δ_1 and δ_2 are non-negative parameters that characterize the load-delay relationship. To measure the latter, we have fit experimental data, see Sec. V. Note that for simpler MEC services with low load where the processing delay per request is constant, we could use a simpler expression (linear on λ) by setting $\delta_2{=}0$. In any case, the delay is linear with the placement decisions and grows fast (non-linearly) with the processing load, which qualitatively is in line with classical queuing models used to capture congestion-based delays in servers.

The *delay cost* of type-b MEC service can be defined as a linear function of the actual delay for all users in all BSs:

$$D^b(\boldsymbol{y}_b)=\theta\cdot\sum_{n=1}^{N}\mu_n^b D_n^b(y_n^b),$$

where $\theta>0$ transforms delay (seconds) in delay cost (monetary units). The smaller the aggregate delay a MEC user experiences, the higher is the perceived performance and hence the smaller the dis-utility (i.e., the delay cost). Conversely, for type-a MEC applications we need to guarantee for each request that $D_n^a\le D_{th}$, where D_{th} can be as small as 1ms for Tactile Internet applications.

Putting the above together, we formulate the MEC-vRAN joint design problem (MvRAN):

[7] In practice, one needs to consider slightly tighter bounds for including the radio processing delay for each configuration.

Problem 1 (MvRAN Design)**.**

$$\min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}} J(\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}) = V(\boldsymbol{x},\boldsymbol{y}) + \sum_{p\in\mathcal{P}} U_p(z_p^{(n)}) + D^b(\boldsymbol{y}_b) \tag{8}$$

$$\text{s.t.} \quad x_{1n} + x_{2n} + x_{3n} + x_{4n} = 1, \qquad \forall n \in \mathcal{N} \tag{9}$$

$$y_n^a \leq x_{1n}, \quad y_n^b \leq x_{1n}, \qquad \forall n \in \mathcal{N} \tag{10}$$

$$\sum_{i=1}^{4} x_{in}\rho_{i,r}\lambda_n + y_n^a\rho_a\lambda_n^a + y_n^b\rho_b\lambda_n^b \leq P_n, \qquad \forall n \in \mathcal{N} \tag{11}$$

$$\sum_{n=1}^{N}\Big(\sum_{i=1}^{4} x_{in}\rho_{i,c}\lambda_n + (1-y_n^a)\rho_a\lambda_n^a + (1-y_n^b)\rho_b\lambda_n^b\Big) \leq P_0 \tag{12}$$

$$\sum_{i=1}^{4} x_{in}\tau_{i,r}\mu_n + y_n^a\tau_a\mu_n^a + y_n^b\tau_b\mu_n^b \leq T_n, \qquad \forall n \in \mathcal{N} \tag{13}$$

$$\sum_{n=1}^{N}\Big(\sum_{i=1}^{4} x_{in}\tau_{i,c}\mu_n + (1-y_n^a)\tau_a\mu_n^a + (1-y_n^b)\tau_b\mu_n^b\Big) \leq T_0 \tag{14}$$

$$D_n^a(y_n^a) \leq D_{th}, \qquad \forall n \in \mathcal{N} \tag{15}$$

$$x_{1n},\, x_{2n},\, x_{3n},\, x_{4n},\, y_n^a,\, y_n^b \in \{0,1\}, \qquad \forall n \in \mathcal{N} \tag{16}$$

$$\sum_{p\in\mathcal{P}_n} z_p^{(n)} = S_n(\boldsymbol{x}_n, y_n^a, y_n^b), \qquad \forall n \in \mathcal{N} \tag{17}$$

$$\sum_{p\in\mathcal{P}_n^{(i)}} z_p^{(n)} \leq M(1 - x_{in}), \quad \forall n \in \mathcal{N},\ i \in \{1,2,3,4\} \tag{18}$$

$$\sum_{n\in\mathcal{N}}\sum_{p\in\mathcal{P}_n} z_p^{(n)} I_p^{ij} \leq c_{ij}, \qquad \forall (i,j) \in \mathcal{E} \tag{19}$$

$$z_p^{(n)} \geq 0, \ \forall p \in \mathcal{P}_n, \qquad \forall n \in \mathcal{N} \tag{20}$$

where we used a Big-M formulation in (18), with $M >> 0$, to deactivate the delay-ineligible paths for each configuration.

Theorem 1. *MvRAN Problem is NP-hard to solve.*

Proof. This result can be showed with a polynomial reduction from the multi-dimensional multiple-choice Knapsack problem ($MMKP$) [21]: there are n groups of items and m resource types; each group i has l_i items; each item j of group i has value v_{ij} and requires r_{ijk} units of type-k resource. The objective is to select exactly one item from each group so as to maximize the value of collected items subject to the constraints for each resource. Consider a restricted case of MvRAN where all paths are eligible (low delay and high capacity) and there is no routing cost. Hence, for every configuration we can trivially find a routing policy. This instance can be directly mapped to $MMKP$: configuration options are mapped to knapsack items ($l_i = 4, \forall i$), where each group is a BS, and processing and memory resource constraints to knapsack constraints. Hence, if we could solve MvRAN in polynomial time, we could do so for $MMKP$, which does not hold. □

Next, we introduce a solution method for the MvRAN design problem that leverages a tailored decomposition method.

IV. Decomposition and Optimization Algorithm

The computational complexity of this problem increases substantially for large scales. In order to expedite its solution, we employ Benders' decomposition method [22] that separates a mixed integer problem into a *slave* subproblem (with the continuous variables only) and a *master* problem (with the discrete variables). Using this approach we decompose MvRAN to the *fronthaul routing* optimization problem (*slave*) and the *vRAN-MEC configuration* problem (*master*). Based on Benders' seminal *Separation Theorem*, the solution of these two distinct problems, properly combined, will yield the exact solution of the initial problem. The algorithm progresses iteratively. In each round, it places the RAN and MEC functions (assuming fixed routing decisions), and then optimizes the routing decisions (for fixed function placement). In each iteration we create additional cuts which confine the solution, gradually leading it to the optimal point of the initial problem.

In detail, the *slave* routing problem is obtained if we fix the binary configuration variables to $\bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}$:

$$P_S(\bar{\boldsymbol{x}},\bar{\boldsymbol{y}}): \quad \min_{\boldsymbol{z}\geq\boldsymbol{0}} \sum_{p\in\mathcal{P}} \gamma_p \sum_{n\in\mathcal{N}} z_p^{(n)} \tag{21}$$

$$\text{s.t.} \sum_{n\in\mathcal{N}}\sum_{p\in\mathcal{P}_n} z_p^{(n)} I_p^{ij} \leq c_{ij}, \qquad \forall\ (i,j) \in \mathcal{I} \tag{22}$$

$$\sum_{p\in\mathcal{P}_n} z_p^{(n)} = S_n(\bar{x}_n, \bar{y}_n^a, \bar{y}_n^b), \qquad \forall\ n \in \mathcal{N} \tag{23}$$

$$\sum_{p\in\mathcal{P}_n^{(i)}} z_p^{(n)} \leq M(1 - \bar{x}_{in}),\ \forall n \in \mathcal{N},\ i \in \{1,2,3,4\} \tag{24}$$

The dual of P_S can be succinctly written as follows:

$$P_{SD}(\bar{\boldsymbol{x}},\bar{\boldsymbol{y}}): \ \max_{\boldsymbol{\pi}} \ \ g(\bar{\boldsymbol{x}},\bar{\boldsymbol{y}},\boldsymbol{\pi}) \ \text{ s.t. } \boldsymbol{H}^T\boldsymbol{\pi} \leq \boldsymbol{\gamma}, \tag{25}$$

where $\boldsymbol{\gamma}$ is the routing cost vector, matrix $\boldsymbol{H}$ is defined by the objective and constraints (22)-(23), $\boldsymbol{\pi}$ is the vector of the $|\mathcal{I}| + 5|\mathcal{N}|$ dual variables, and the dual function is linear:

$$g(\bar{\boldsymbol{x}},\bar{\boldsymbol{y}},\boldsymbol{\pi}) = -\sum_{(i,j)\in\mathcal{I}} c_{ij}\pi_{1ij} + \sum_{n\in\mathcal{N}} S_n(\bar{x_n}, \bar{y_n})\pi_{5n}$$
$$-\sum_{i=1}^{4}\sum_{n\in\mathcal{N}} M(1 - \bar{x}_{in})\pi_{in}\,.$$

The *master* function placement problem is:

$$P_M(\mathcal{C}_1,\mathcal{C}_2): \quad min_{\boldsymbol{x},\boldsymbol{y},\psi} V(\boldsymbol{x},\boldsymbol{y}) + D^b(\boldsymbol{y}_b) + \psi \tag{26}$$

$$\text{s.t.} \quad y_n^a \leq x_{1n}\,, \quad y_n^b \leq x_{1n}, \qquad \forall n \in \mathcal{N} \tag{27}$$

$$x_{1n} + x_{2n} + x_{3n} + x_{4n} = 1, \qquad \forall n \in \mathcal{N} \tag{28}$$

$$\sum_{i=1}^{4} x_{in}\rho_{ir}\lambda_n + y_n^a\rho_a\lambda_n^a + y_n^b\rho_b\lambda_n^b \leq P_n, \qquad \forall n \in \mathcal{N} \tag{29}$$

$$\sum_{n=1}^{N}\Big(\sum_{i=1}^{4} x_{in}\rho_{ic}\lambda_n + (1-y_n^a)\rho_a\lambda_n^a + (1-y_n^b)\rho_b\lambda_n^b\Big) \leq P_0 \tag{30}$$

$$\sum_{i=1}^{4} x_{in}\tau_{ir}\mu_n + y_n^a\tau_a\mu_n^a + y_n^b\tau_b\mu_n^b \leq T_n, \qquad \forall n \in \mathcal{N} \tag{31}$$

$$\sum_{n=1}^{N}\Big(\sum_{i=1}^{4} x_{in}\tau_{ic}\mu_n + (1-y_n^a)\tau_a\mu_n^a + (1-y_n^b)\tau_b\mu_n^b\Big) \leq T_0 \tag{32}$$

$$D_n^a(y_n^a) \leq D_{th}, \qquad \forall n \in \mathcal{N} \tag{33}$$

$$g(\boldsymbol{x},\boldsymbol{y},\boldsymbol{\pi}^m) \leq \psi, \qquad \forall \boldsymbol{\pi}^m \in \mathcal{C}_1 \tag{34}$$

$$g(\boldsymbol{x},\boldsymbol{y},\boldsymbol{\pi}^l) \leq 0, \qquad \forall \boldsymbol{\pi}^l \in \mathcal{C}_2 \tag{35}$$

$$\psi \geq 0, \quad x_{1n},\, x_{2n},\, x_{3n},\, x_{4n},\, y_n^a,\, y_n^b \in \{0,1\} \tag{36}$$

Algorithm 1 summarizes the optimization steps. The sets $\mathcal{C}_1$

Algorithm 1: MvRAN Decomposition Algorithm

```
1  Initialize: $\epsilon$, $\tau = 1$; $\mathcal{C}_1^{(0)} = \mathcal{C}_2^{(0)} = \emptyset$; $UB^{(0)} = -LB^{(0)} >> 1$
2  repeat
3      Solve problem $P_M(\mathcal{C}_1^{(\tau)}, \mathcal{C}_2^{(\tau)})$ to obtain $\boldsymbol{x}^{(\tau)}, \boldsymbol{y}^{(\tau)}, \psi^{(\tau)}$.
4      Set $LB^{(\tau)} = V_0(\boldsymbol{x}^{(\tau)}, \boldsymbol{y}^{(\tau)}) + \psi^{(\tau)} + \sum_n V_n(\boldsymbol{x}^{(\tau)}, \boldsymbol{y}^{(\tau)})$.
5      Solve problem $P_{SD}(\boldsymbol{x}^{(\tau)}, \boldsymbol{y}^{(\tau)})$ to obtain $\boldsymbol{\pi}^{(\tau)}$.
6      If $UB^{(\tau)} < UB^{(\tau-1)}$ then $UB^{(\tau)} =$
           $V_0(\boldsymbol{x}^{(\tau)}, \boldsymbol{y}^{(\tau)}) + g(\boldsymbol{\pi}^{(\tau)}, \boldsymbol{x}^{(\tau)}, \boldsymbol{y}^{(\tau)}) + \sum_n V_n(\boldsymbol{x}^{(\tau)}, \boldsymbol{y}^{(\tau)})$.
7      If $g(\boldsymbol{\pi}^{(\tau)}, \boldsymbol{x}^{(\tau)}, \boldsymbol{y}^{(\tau)}) < \infty$ then
           $\boldsymbol{\pi}^m \leftarrow$ extreme point
           $\mathcal{C}_1^{(\tau+1)} = \mathcal{C}_1^{(\tau)} \cup \{\boldsymbol{\pi}^m\}$.
8      If $g(\boldsymbol{\pi}^\tau, \boldsymbol{x}^\tau, \boldsymbol{y}^\tau) \rightarrow \infty$ then
           $\boldsymbol{\pi}^l \leftarrow$ extreme direction/ray
           $\mathcal{C}_2^{(\tau+1)} = \mathcal{C}_2^{(\tau)} \cup \{\boldsymbol{\pi}^l\}$.
9      $\tau = \tau + 1$.
   until $UB^{(\tau)} - LB^{(\tau)} \leq \epsilon$;
10 Set the optimal configuration as $\boldsymbol{x}^* = \boldsymbol{x}^{(\tau)}$ and $\boldsymbol{y}^* = \boldsymbol{y}^{(\tau)}$.
11 Compute the optimal routing $\boldsymbol{z}^*$ by solving $P_{SD}(\boldsymbol{x}^{(\tau)}, \boldsymbol{y}^{(\tau)})$.
```

and $\mathcal{C}_2$ contain the parameters based on which we create the cuts. In each iteration τ, we first solve the *master* problem in order to obtain the currently optimal configuration $\boldsymbol{x}^{(\tau)}, \boldsymbol{y}^{(\tau)}$, and also the surrogate parameter $\psi^{(\tau)}$ (Step 3). These are used to set the current lower bound $LB^{(\tau)}$ (Step 4). Then, we solve the dual *slave* problem P_{SD} using the current variables $\boldsymbol{x}^{(\tau)}, \boldsymbol{y}^{(\tau)}$ (Step 5). Next, we calculate the new upper bound, using the value of the relaxed master problem (Step 6). After each iteration we update the sets $\mathcal{C}_1$ and $\mathcal{C}_2$, by adding the respective values $\boldsymbol{\pi}$ in $\mathcal{C}_1$ if the dual optimal value is bounded, or in $\mathcal{C}_2$ if the dual is unbounded; these will be used in solving P_M in the next iteration. These steps are repeated until the upper and lower bounds coincide, and the solution precision can be tuned by selecting ϵ. Note that, if MvRAN is unfeasible, then we will obtain an unbounded value for the slave problem in the first iteration [22].

The *master* problem is computationally challenging but its dimension has been substantially reduced (compared to MvRAN problem) as we have replaced all the routing variables with ψ. There are several methods to expedite its solution. For example, during the first iterations we can remove the integrality constraints (LP relaxation) for the configuration variables (in the master problem), until the $UB - LB$ gap has been reduced enough. This will make P_M solvable in polynomial time. When the gap reaches a low value (i.e., $|(UB - LB| \rightarrow 0)$, we can re-introduce these integrality constraints so as to obtain a feasible solution. This method is proved to preserve optimality, since no solutions are removed by the cuts that are added during the first stage iterations [23]. The next theorem describes the Algorithm's performance.

Theorem 2. ***Performance of Algorithm 1****. It converges in a finite number of iterations to the optimal solution of MvRAN.*

Proof. The proof follows from the Partition Theorem in [22] which we can directly apply here. Namely, using a more abstract notation (for brevity) the problem's solution can be obtain by solving equivalently:

$$\min_{\boldsymbol{x},\boldsymbol{y},\psi} \; c_1^T \boldsymbol{x} + c_2^T \boldsymbol{y} + \psi \quad \text{s.t.} \;\; (\boldsymbol{x}, \boldsymbol{y}, \psi) \in \mathcal{G}\,, \tag{37}$$

where $\mathcal{G}$ is the set of constraints for all variables, created by the intersection of the constraints in $\mathcal{X}$, $\mathcal{Y}$ and the convex hull of the extreme halflines stemming from the dual slave problem (which is a polyhedral cone $\mathcal{C}$). The algorithm starts with the minimal set of constraints $\mathcal{G}^{(0)}$ (for $\mathcal{C}_1 = \mathcal{C}_2 = \emptyset$) and at each iteration τ adds one extreme halfline of the cone $\mathcal{C}$ in $\mathcal{G}^{(\tau)}$ by modifying the sets $\mathcal{C}_1^{(\tau)}$ and $\mathcal{C}_2^{(\tau)}$. Given that there are finite such constraints , and since in each iteration we add a different halfline, the algorithm terminates in a finite number of steps. The convergence to the optimal solution is ensured by the fact that, in the worst case, we will reconstruct the initial set $\mathcal{G}$. □

V. Data Analysis

In this section we experimentally study a face recognition application (OpenFace [16]) that corresponds to a type *b* MEC service in our system model. Our goal is to understand what are its memory, CPU and bandwidth requirements. We also present and analyze three real cellular RAN topologies from different operators in 3 European cities. These will be used in the sequel for evaluating our optimization framework.

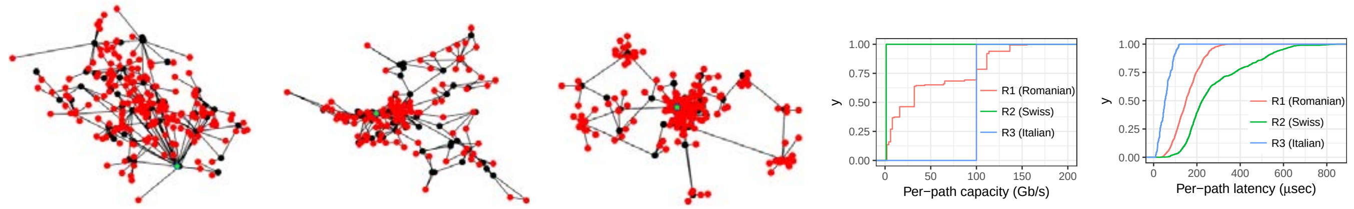
A. Network Topologies

We studied the RANs from operators in Romania (denoted R1), Switzerland (R2), and Italy (R3), with 198, 197 and 200 RUs, respectively; and we plot them in Fig. 2(a)-(c). First, we note that the *RANs are heterogeneous*. R3 has mainly fiber links, R2 wireless links and R1 has fiber, copper and wireless links. The networks differ also in the number of paths connecting the RUs with the CU. R1 has high path redundancy (mean of 6.6 paths), while in R3 several RUs have only 1 path (mean 1.6). This difference reveals the need for a tailored vRAN design for each network. Second, we observe that the RANs *do not have a typical structure*. Some RUs are located up to 20Km far from the CU (in R3), while others are in 0.1Km distance. Hence, any non-optimized heuristic vRAN configuration policy could be arbitrarily inefficient. Finally, the RANs have links with diverse capacity (ranging from 2 Gb/s to 2000Gb/s) and diverse path delay (which we calculated with a standard store-and-forward model.[8]) as shown in Fig. 2(d)-(e). We note that delay is up to 40 times higher in some paths, and that many RUs cannot support C-RAN (Split 4).

B. An Edge Computing Application: Face Recognition

We set up a scenario with a 16-core off-the-shelf server with 128 GB RAM running (virtualized) face recognition services, a standard laptop running the clients (emulating users), and a 1 Gb/s Ethernet network connecting users and services (to ensure negligible delay). We use the open-source service OpenFace and a web front-end for the client. In order to emulate multiple clients we generate virtual video devices using `v4l2loopback` kernel module, and feed the same pre-recorded video to each virtual device in order to protect repeatability. We employ two qualities in the video feed: "low resolution" with H.264/AVC codec and 840×480 resolution (usual specs from a laptop webcam); and a "high resolution" video with the same codec and 1920×1080 resolution.

[8] We conservatively used $12000/c_{ij}$, 4μs/Km (cable) or 3μs/Km (wireless), and $5\mu secs$ for transmission, propagation, and processing delay, respectively.



(a) Romanian topology (R1). (b) Swiss topology (R2). (c) Italian topology (R3). (d) Path Capacity Distribution (e) Path Delay Distribution

Fig. 2: (a)-(c): Three actual RANs in Europe: red dots indicate the RUs' locations; black dots the routers/switches; and green dot the CU location which has been placed at the EPC (most central position). (d)-(e) The eCDF of path capacity and delay in these topologies.

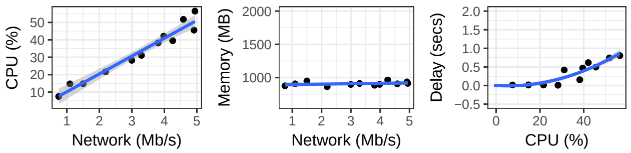


Fig. 3: Relationship between CPU/memory consumption, network load and delay performance. The load is increased by changing the video quality and the frame rate T.

We deploy in our server a *dockerized*[9] OpenFace service for every client. Each (virtual) video device sends a video frame to the service every T seconds. Upon reception, the service analyzes the picture and processes a bounding box (around the recognized face in the picture) and a label (tagging the identified person), and then feeds this information to the client. The client displays the bounding box and the label, and initiates an object tracking procedure to let the box and label follow the video stream smoothly. Each experiment lasts 100 secs and we assess both video qualities and various picture transmission rates T, measuring the service delay for every request and resource consumption every 2 secs.

To facilitate presentation, we focus on one client creating flows with different loads—namely, we vary the frame rate:[10]

$$T = \{0.128, 0.256, 0.512, 1.024, 2.048, 4.096\},$$

and we also change the video quality in order to assess the impact of the network load on the CPU consumption, memory consumption and service request delay. Fig. 3 presents the results, evincing a linearly increasing relationship between CPU utilization with network load, and constant memory utilization. In fact, the memory increases with the number of clients (each one using a different docker container) but is relatively constant with the client's load. These measurements validate our model in Problem 1 (eqs. (11)-(14)). In the last plot of Fig. 3, we depict the mean delay for each video frame request as a function of the CPU load, which suggests an increasing relationship between the two (and, subsequently, with network load) that we approximate with a quadratic fit that motivate our delay model in eq. (7). The fitted parameters of eq. (7) for the OpenFace application are $\delta_1 = 0.25$ ms, $\delta_2 = 0.25$ ms and $\rho = 0.1$.

VI. Performance Evaluation

In the following, we evaluate MvRAN using the real RANs (Fig. 2) and our OpenFace measurements. We are interested to explore how the network load and system parameters affect the number of vRAN and MEC functions that are placed at the CU, i.e., their *centralization degree*, and the different system costs. Occasionally, we compare our findings with the typical D-RAN architecture (full-stack base station implementation used in 3G/4G), and the C-RAN architecture (all functions at CU). Finally, we focus here mainly in type b MEC services as their elasticity renders the evaluation more intricate (type a impose only hard constraints).

A. Evaluation Setup

The parameters we use are summarized in Table I. We assume 1 user/TTI, 20MHz (100 PRBs), 2×2 MIMO, CFI=1, 2 TBs of 75376 bits/subframe, and IP MTU 1500B, for a high-load scenario $\lambda = 150$Mb/s for every RU. We consider an Intel Haswell i7-4770 3.40GHz CPU core as the *reference core* (RC) for the CPU capacity. From our measurements and those in [24], we estimate that f_3 yields 20% of the total consumption of a software-based LTE BS, f_2 for 15%, and f_1 65%. From [4], we calculate the computing needs of a software-based LTE RU, and we set it to 750 μs of the reference CPU for processing each 1-ms subframe [4], which translates in 75% of 1 RC. Hence, we set $\rho_{i,r} = \{0.05, 0.04, 0.00325, 0\}$ and $\rho_{i,c} = \{0, 0.001, 0.00175, 0.05\}$ RCs per Mb/s for $i = \{1, 2, 3, 4\}$, respectively. Finally, unless otherwise stated, we set $P_0 = 75$ RCs and $P_n = 2$ RCs, $\forall n \in \mathcal{N}$ and, motivated by our previous experiments with OpenFace, $\rho_a = \rho_b = 0.1$ (equal to ease presentation) with $D_{th} = 1$ ms.

Quantifying computing and routing cost is challenging as it depends on the hardware, leasing agreements, and so on. However, note that in our problem the function placement decisions are determined by the relative values of the computing cost parameters across RUs and CU (α_0, β_0 and α_n, β_n), and the ratio of computing over routing cost (γ). Hence, we perform our analysis using relative values for $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$. According to [2], the equipment cost of a D-RAN BS is estimated to \$50K whereas the respective cost of a C-RAN BS (i.e., RU with Split

Param.	Default value	Description
γ_p	0 Gb/s^{-1}	Cost* per Gb/s across path p
α_n	1	Cost* per vRAN function in RU n
α_0	$0.5\alpha_n$	Cost* per vRAN function in CU
β_n	1 RC^{-1}	Cost* per CPU usage in RU n
β_0	$0.017\beta_n$ RC^{-1}	Cost* per CPU usage in CU
θ	1 sec.$^{-1}$	Cost* per MEC type b delay
D_{th}	1 ms	MEC type a delay tolerance
$\rho_{i,r}$	$\{50, 40, 3.25, 0\}$ RCs/Gbps	CPU req. in a RU for vRAN split i
$\rho_{i,c}$	$\{0, 1, 1.75, 50\}$ RCs/Gbps	CPU req. in a CU for vRAN split i
$\rho_{a(b)}$	0.1 RCs/Mbps	CPU req. of MEC type a(b)
P_n	2 RCs	CPU capacity of RU n
P_0	75 RCs	CPU capacity of CU
λ_n	50 Mb/s	non-MEC network load at BS n
$\lambda_n^{a(b)}$	10 Mb/s	MEC type a(b) network load at BS n

* Cost in units of account or nominal monetary units

TABLE I: Model parameters; see §VI-A for justification.

[9] A (Docker) container is a light virtualization technology that ensures resource isolation features. Docker's swarm clusters are particularly useful in our case because a Docker's cluster manager takes care of balancing the clients' data load across the different OpenFace servers.

[10] Since several video frames display the same faces, T can naturally be lower than the actual video frame rate. Note than an object tracking procedure in the client allows labels to move in real-time with the video playout.

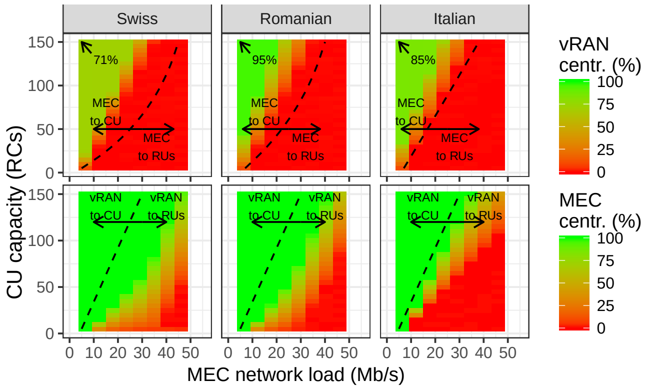


Fig. 4: vRAN (top) *vs.* MEC centralization (bottom).

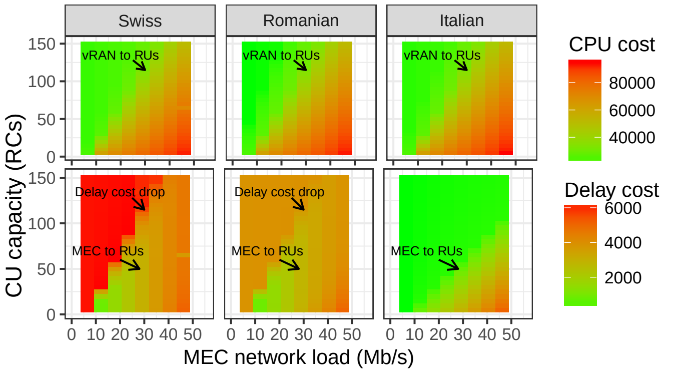


Fig. 5: Computation costs (top) *vs.* delay costs (bottom).

4, Fig. 1) is \$25K. Therefore, we consider that CU is twice cost-efficient for function instantiation i.e., $\alpha_0 = \alpha_n/2$; and we set, unless otherwise stated, $\alpha_n = 1\,\forall n \in \mathcal{N}$, to simplify the discussion. The main advantage of the CU processing cost compared to RUs comes from cooling, CPU load balancing, etc. Based on [5], we estimate this cost to be $\beta_0 = 0.017\beta_n$ (linear regression in [5, Fig.6a]). If we take as reference the processing cost at RU, then $\beta_0 = 0.017$ and $\beta_n = 1$. Finally, unless otherwise stated, we set $\theta = 1$ for the delay cost.

B. vRAN and MEC Centralization Degree

We first assess the centralization degree of both vRAN and MEC and its impact on the system cost when using MvRAN. We thus present in Fig. 4 the degree of centralization of vRAN and MEC functions for a wide range of CU processing capacity P_0 values (y-axis) and MEC network load values $\lambda_n^a = \lambda_n^b, \forall n \in \mathcal{N}$ (x-axis). The legacy (non-MEC) network load is considered 50 Mb/s. The top and bottom plot show, respectively, the percentage of vRAN functions and MEC functions placed at the CU (centralization degree).

There are two easy distinguishable areas in these plots: *(i)* low MEC network load and high computational capacity (top left corners), and *(ii)* high network load and low computational capacity (bottom right corners). The optimal configuration provided by MvRAN is very intuitive in these two areas; namely, *(i)* high vRAN and MEC centralization in the former, and *(ii)* high vRAN and MEC decentralization in the latter. In between, we can observe a wide operational regime where vRAN functions are highly decentralized whereas MEC functions are centralized in the CU.

From these results, we can conclude that we have three main operation areas: two *extreme* regimes, namely low network load and high computing capacity; and high network load and low computing capacity, that render a similar solution across topologies (max. vRAN centralization and D-RAN, respectively); and an area where the functions placement is mixed and highly dependent on the topology. In the latter we can even observe cases where some RUs implement all the BS functions but still the MEC functions are placed at the CU.

Fig. 5 shows the computational cost $V(\boldsymbol{x}, \boldsymbol{y})$ at the top plot and the delay cost $D^b(\boldsymbol{y}^b)$ at the bottom for the network configurations of Fig. 4. The results shown in the first plot are rather intuitive when are compared to Fig. 4 (top): higher vRAN/MEC centralization degree yields reduced compute costs, particularly under low network/compute demand. In order to analyze the delay costs (Fig. 5 bottom), we revisit Fig. 2, which shows that the Italian topology has in general lower path delays with respect to the other two networks (Swiss having the highest delays). This explains the noticeable delay cost differences across the three networks in the top left corner (high computational capacity and low MEC network load) of the three lower plots in Fig. 5. When the MEC network load increases—and especially for low computational capacity values—there is a region of values where the delay cost drastically improves for both Swiss and Romanian topologies, while it remains low for Italian. Remarkably, this coincides with a pronounced increase in computational cost (top plot), which is in turn caused by an increase in vRAN decentralization. The latter is necessary, despite the compute cost increase it induces, in order to release compute resources for the heavy MEC load.

C. Variable Computational and Delay Costs

We next evaluate the impact that different delay and computing cost parameters have on the centralization degree of vRAN and MEC functions when using MvRAN. To this aim, Fig. 6 shows the percentage of vRAN and MEC functions that are centralized (centralization degree) for a wide set of delay cost weights θ and compute costs $\beta_n = \beta$ when $\alpha_0 = \alpha_n = 0$ to focus on compute costs that are directly proportional with network load. For this analysis we fix the MEC and background network load to 10 and 50 Mb/s, respectively, and the CU capacity to 75 RCs. The figure depicts three regimes: (i) high delay and low computing cost parameters (top left area), with low vRAN and MEC centralization degree as we are prioritizing having MEC functions closer to the RUs to improve delay and in this regime compute pooling has relatively less benefit; (ii) low delay and high computing cost parameters (bottom right area), with high vRAN and MEC centralization as we are in the opposing case, i.e., higher computing gains yield higher benefits from compute pooling compared to the overall cost gain that improving delay would attain; and (iii) a regime in between the previous two, where vRAN has 0% centralization while MEC is highly centralized, which is explained by the fact that with these cost settings there is some benefit from computing pool but also some from delay improvement, and therefore the most balanced configuration is to release pooled computing resources from the CU to improve delay of the centralized MEC functions.

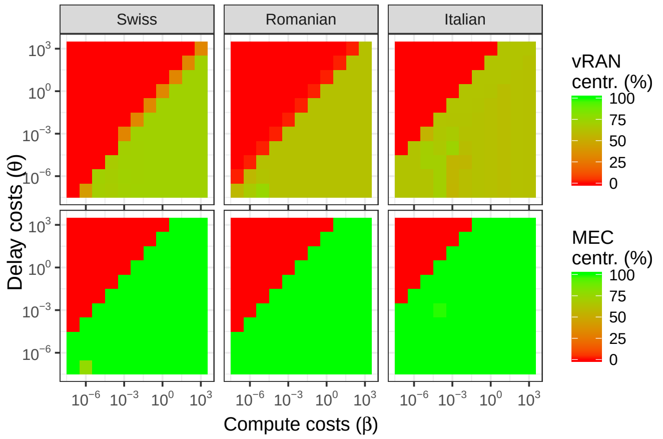


Fig. 6: Sensitivity analysis of delay costs (θ) over compute costs β.

D. *Impact of Transport Cost*

We now assess the impact of routing costs on vRAN/MEC centralization and overall system cost when using MvRAN. Fig. 7 presents the percentage of centralized vRAN and MEC functions (top) and the overall system cost (bottom) for a range of routing cost $0 \leq \gamma \leq 40$ Gb/s^{-1}. Both MEC and legacy network load is set to 50 Mb/s. When $\gamma = 0$ we have no routing cost, and when $\gamma = 40$ the routing costs are almost 40 times larger than the computing costs (for the same network load). We remark that this is a worst-case scenario (in terms of centralization advantage) because the computing costs at CU and RU are equal. We compare the MvRAN solution to C-RAN configuration (all functions placed at CU) and D-RAN (all functions at RUs). As we did earlier, we still optimize the routing for these two extreme cases, to make the comparison fair. We stress again that C-RAN, though presented for comparison purposes, is not implementable in these networks due to capacity violations.

Let us focus on the top plot of Fig. 7. As revealed in the previous subsection, MvRAN places as many functions as possible at the CU to minimize the total cost when routing costs are low. Specifically, when $\gamma < 0.25$ it reduces up to 2.5 times the cost of by D-RAN. Clearly, given the selected compute costs, vRAN centralization is beneficial due to compute resource pooling. Moreover, for low values of γ, C-RAN (which however violates network capacities in these RANs) yields the lowest cost, but MvRAN noticeable follows closely (and is implementable). As γ increases, there is a point where MvRAN and C-RAN yield the same system cost ($\gamma = 0.25$); and thereafter C-RAN cost increases linearly and even surpasses D-RAN cost when $\gamma > 1$. Conversely MvRAN achieves the lowest cost when $\gamma > 0.25$, until it converges to D-RAN (when $\gamma = 32$ Gb/s^{-1}). From that point on, MvRAN selects the same configuration as D-RAN.

E. *MvRAN Cost Savings*

We finally assess the cost savings obtained with our approach, comparing it to two extreme configurations, namely, C-RAN and D-RAN. These are two special cases of MvRAN where the function placement variables are fixed, i.e., routing is still optimized. Fig. 8 presents the relative cost increase of both configurations relative to the cost of MvRAN, using the same CU capacity and MEC load settings. We stress that the C-RAN configuration is not implementable in our topologies due to network capacity violations, but the respective cost is shown in the figure for comparison purposes. The figure shows that MvRAN always yields better cost than D-RAN, even when MEC network load is high (as some MEC functions can still benefit of a cheaper computational processing in the CU). Conversely, C-RAN might yield better cost in certain setups (again, this configuration is not deployable in any case), but MvRAN achieves a comparable cost and moreover C-RAN becomes unfeasible for a wide range of parameters as the CU computing nodes starve when the MEC load is high.

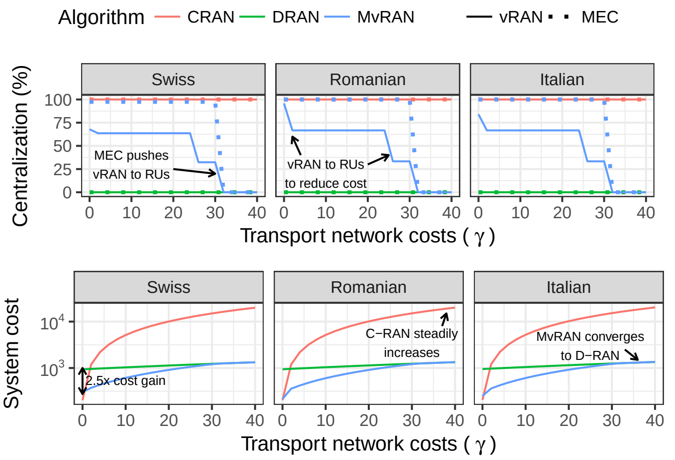


Fig. 7: vRAN/MEC centralization and system cost for various transport costs, for C-RAN, D-RAN, MvRAN.

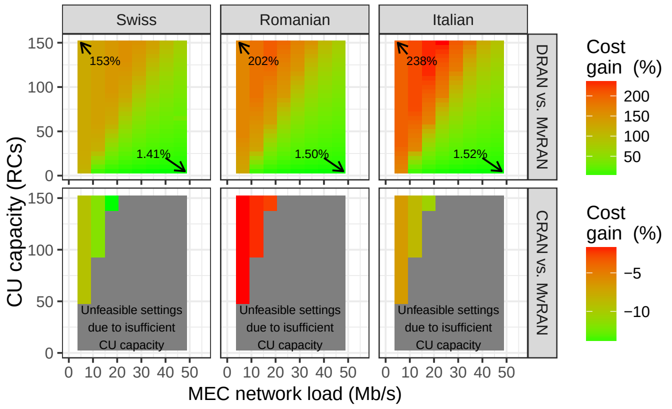


Fig. 8: Overall cost gain relative to MvRAN.

VII. Related Work

Next Generation RAN. The C-RAN idea ignited many research efforts [1]–[3], [5], [6], and was extended to combine dummy RUs with advanced BSs [25], and the suggestion for virtualized RAN [26] that promotes hybrid centralized/decentralized architectures. Also, [7] and recently [27] studied BS function splits different from pure C-RAN in order to relax its requirements. Other works presented a cost-benefit analysis [3], [10]–[12], [28], or studied implementation issues [8], [9]. Our work is aligned with these suggestions, and optimizes jointly the RAN costs and MEC delay under the most advanced scenario where a different split can be selected for each BS. The recent cloudification of RAN enable the solution of these two problems in the same time-scale (unlike the dynamic scheduling of the last-hop wireless links).

Computing in RAN: MEC has been introduced to enable new use cases from vertical industries [13] and support demanding services such as those that are envisioned in Tactile Internet [14]. Prior works have focused on the optimization at the user side [15], while here we decide how to deploy MEC in the RAN. In-network processing in RAN (Fog-RAN) has been studied in [25] where advanced RUs enable cooperative radio management, and in [29] that designed a joint transmission and caching policy. Our approach differs fundamentaly from these works as it considers how in-network processing can be used for MEC services (not solely for radio tasks), and jointly deploys the RAN-related and MEC functions.

MEC Optimization: In our recent work we showed that MEC traffic affects the BS split selection [20], and others have also emphasized the relation between RAN and MEC [19]. In practice however, supporting MEC services is an intricate problem as they induce various compute and network costs, have low-latency and high-throughput requirements. These factors compound the joint vRAN-MEC design problem which is currently unaddressed. Recent proposals for optimizing MEC include [30], [31] that consider power-constrained computing latency minimization in a single-user MEC system; and [32] which minimizes energy consumption in multi-user MEC systems. To the best of our knowledge, our work is the first addressing computing and networking delay constraints that appear in a joint MEC and vRAN design problem. This unified view is very crucial for emerging applications with stringent performance requirements needs [14].

Finally, regarding our algorithm, we used Benders' decomposition [22] that yields the exact solution of the problem. While this method does not offer convergence time guarantees, we found that in practice converges very fast for the problems at hand. Faster solutions could, perhaps, be achieved using Lagrange-based or Integer relaxation methods, or approximation techniques; yet, for this network design problem it is reasonable to prioritize optimality over solution speed.

VIII. Conclusions

Recent advances in the *cloudification* of BSs allow the joint design of both vRAN and MEC architectures (in the same time scale), and create new opportunities for the design of next generation cellular systems that support services of the Tactile Internet. Motivated by these developments, we provide a novel optimization framework for deciding how to select the function split for each BS, where to place the MEC functions, and how to route the data in the shared fronthaul network. We measure the compute, memory, and communication requirements of a typical MEC service (OpenFace) and optimize its deployment in three actual RANs. Our experimentally-validated framework selects indeed a hybrid vRAN design showing that C-RAN is not always optimal in terms of cost, and that D-RAN is not always preferable for minimizing MEC delay.

Acknowledgements

Work supported by the EC under Grant No 761536 (5G-Transformer) and by SFI under Grant No 17/CDA/4760.

References

[1] Y. Lin *et al.*, "Wireless Network Cloud: Architecture and System Requirements," *IBM Journal of R&D*, vol. 54, pp. 1–12, April 2010.

[2] V. Suryaprakash, P. Rost, and G. Fettweis, "Are Heterogeneous Cloud-Based Radio Access Networks Cost Effective?" *IEEE JSAC*, vol. 33, no. 10, pp. 2239–2251, May 2015.

[3] A. Checko *et al.*, "Evaluating C-RAN Fronthaul Functional Splits in Terms of Network Level Energy and Cost Savings," *Journal of Communications and Networks*, vol. 18, no. 2, pp. 162–172, June 2016.

[4] N. Nikaein, "processing radio access network functions in the cloud: Critical issues and modeling."

[5] P. Rost, S. Talarico, and M. C. Valenti, "The Complexity-Rate Tradeoff of Centralized Radio Access Networks," *IEEE Trans. on Wireless Comm.*, vol. 14, no. 11, pp. 6164–6176, June 2015.

[6] C.-L. I., Y. Yuan, J. Huang, S. Ma, C. Cui, and R. Duan, "Rethink Fronthaul for Soft RAN," *IEEE Comm. Magazine*, vol. 53, no. 9, 2015.

[7] U. Dotsch *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. 1–24, June 2013.

[8] C. Chang, N. Nikaein, and T. Spyropoulos, "Impact of Packetization and Functional Split on C-RAN Fronthaul Performance," *IEEE ICC*, 2016.

[9] C.-Y. Chang *et al.*, "FlexCRAN: A Flexible Functional Split Framework Over Ethernet Fronthaul in Cloud-RAN," *in Proc. of IEEE ICC*, 2017.

[10] S. C. Forum, "R6.0. Small cell Virtualization Functional Splits and Use Cases," *Document 159.07.02, Release 7*, June 2016.

[11] ——, "Study on New Radio Access Technology: Radio Access Architecture and Interfaces," *TR 38.801*, June 2016.

[12] A. Garcia-Saavedra *et al.*, "Wizhaul: On the centralization degree of cloud ran next generation fronthaul," *IEEE Transactions on Mobile Computing*, vol. 17, no. 10, pp. 2452–2466, Oct 2018.

[13] ETSI/MEC, http://www.etsi.org/technologies-clusters/technologies/multi-access-edge-computing, 2017.

[14] M. Simsek *et al.*, "5G-Enabled Tactile Internet," *IEEE Journal on Sel. Areas in Comm.*, vol. 34, no. 3, pp. 460–473, March 2016.

[15] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint Optimization of Radio and Computational Resources for Multicell Mobile-Edge Computing," *IEEE Trans. on Signal and Inf. Proc. over Networks*, vol. 1, no. 2, 2015.

[16] OpenFace-Application, "http://cmusatyalab.github.io/openface/."

[17] K. Kiyoshima *et al.*, "Commercial Development of LTE-Advanced: Applying Advanced C-RAN ," *NTT Docomo Techn. Journ., 17(2)*, 2015.

[18] J. Kuo *et al.*, "Service chain embedding with maximum flow in software defined network and application to the next-generation cellular network architecture," in *IEEE INFOCOM 2017*, May 2017, pp. 1–9.

[19] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, "Collaborative Mobile Edge Computing in 5G Networks: New Paradigms, Scenarios, and Challenges," *IEEE Communications Magazine*, vol. 55, no. 4, 2017.

[20] A. Garcia-Saavedra *et al.*, "FluidRAN: Optimized vRAN/MEC Orchestration," in *IEEE INFOCOM*, 2018.

[21] M. Akbar, M. Rahmanb, M. Kaykobadb, and G. C. Shojaa, "Solving the Multidimensional Multiple-choice Knapsack Problem by Constructing Convex Hull," *Comp. and Oper. Research*, vol. 33, 2006.

[22] J. F. Benders, "Partitioning Procedures for Solving Mixed-Variables Programming Problems," *Numer. Math.*, vol. 4, 1962.

[23] L. Qian *et al.*, "Joint BS Association and Power Control via Benders' Decomposition," *IEEE Trans. on Wireless Comm.*, vol. 12, no. 3, 2013.

[24] C. Y. Yeoh *et al.*, "Performance study of LTE experimental testbed using OpenAirInterface," in *ICACT 2016*, Jan 2016, pp. 617–622.

[25] M. Peng *et al.*, "Fog-Computing-Based Radio Access Networks: Issues and Challenges," *IEEE Network*, vol. 30, no. 4, pp. 46–53, July 2016.

[26] Ericsson and Telefonica, "Cloud RAN Architecture for 5G," *Joint White Paper, Available Online*, June 2017.

[27] X. Wang *et al.*, "Interplay of energy and bandwidth consumption in cran with optimal function split," in *IEEE ICC*, May 2017, pp. 1–6.

[28] I. Koutsopoulos, "Optimal functional split selection and scheduling policies in 5g radio access networks," in *2017 IEEE International Conference on Communications Workshops*, May 2017, pp. 993–998.

[29] S. H. Park, O. Simeone, and S. Shamai, "Joint Optimization of Cloud and Edge Processing for Fog Radio Access Networks," *IEEE Trans. on Wireless Communications*, vol. 15, no. 11, November 2016.

[30] J. Liu, Y. Mao, , and K. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *IEEE ISIT*, 2016.

[31] Y. Zhang, D. Niyato, and P. Wang, "Offloading in mobile cloudlet systems with intermittent connectivity," *IEEE Transactions on Mobile Computing*, vol. 14, no. 12, pp. 2516–2529, Dec 2015.

[32] C. You, K. Huang, H. Chae, and B. H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. on Wireless Communications*, vol. 16, no. 3, 2017.