On the Optimization of Multi-Cloud Virtualized Radio Access Networks

Fahri Wisnu Murti*, Andres Garcia-Saavedra†, Xavier Costa-Perez†, George Iosifidis*

*School of Computer Science and Statistics, Trinity College Dublin
†NEC Laboratories Europe, Heidelberg, Germany

Abstract—**We study the important and challenging problem of virtualized radio access network (vRAN) design in its most general form. We develop an optimization framework that decides the number and deployment locations of central/cloud units (CUs); which distributed units (DUs) each of them will serve; the functional split that each BS will implement; and the network paths for routing the traffic to CUs and the network core. Our design criterion is to minimize the operator's expenditures while serving the expected traffic. To this end, we combine a linearization technique with a cutting-planes method in order to expedite the *exact solution* of the formulated problem. We evaluate our framework using real operational networks and system measurements, and follow an exhaustive parameter-sensitivity analysis. We find that the benefits when departing from single-CU deployments can be as high as 30% for our networks, but these gains diminish with the further addition of CUs. Our work sheds light on the vRAN design from a new angle, highlights the importance of deploying multiple CUs, and offers a rigorous framework for optimizing the costs of Multi-CUs vRAN.**

I. INTRODUCTION

A. Motivation

The Cloud Radio Access Network (C-RAN) has recently emerged as a promising solution for building low-cost high-performance RAN in 5G+ systems. Pooling the base station (BS) functions in central servers within the RAN, reduces their implementation cost and offers unprecedented performance gains. For instance, C-RAN can support Coordinated Multi-Point (CoMP) transmissions, flexible spectrum management, and intelligent interference control [1], [2]. The last few years both academia and industry have been working to standardize C-RANs and expedite their adoption [3]. However, there are still several issues that need to be addressed before we can fully reap the benefits of these systems.

One of the key challenges is the design of optimal C-RAN architectures. This remains an open and difficult problem for many reasons. First, it is clear today that fully centralized RANs are not always implementable as they require expensive high-capacity fronthaul networks. Hence, the latest proposals enable operators to determine the centralization level (or, *functional split*), by selecting which BS functions will be hosted at the central/cloud units (CUs) and which will be kept at the distributed units (DUs). The term virtualized RAN

(vRAN) has been coined to describe these architectures, which in their most flexible version allow even a different split for each BS [4]. However, selecting the level of centralization is an intricate problem, as each functional split creates different computation load for the CUs and DUs, and different data transfer loads for the network.

Furthermore, some vRANs might need to deploy multiple CUs in order to increase the number of centralized BS functions. It is thus necessary to decide the number and deployment location of the CUs, and which DUs each of them should serve. Nevertheless, these *assignment* decisions are affected by the functional split that is selected for each BS. On top of that, while initially the vRAN fronthaul was designed using point-to-point connections (CPRI lines), there is nowadays consensus that these links should be replaced by packet-switched networks where links can be shared by multiple flows. This is more cost-effective, yet it compounds the vRAN design problem as one has, additionally, to decide the *routing* for each BS and each split.

Our goal in this work is: to tackle the vRAN design problem in its most general form by optimizing jointly the number and location of CUs; assignment of DUs to CUs; functional split for each BS; and routing of data in the network.

B. Related Work

The idea of C-RAN was followed, early-on, by the suggestion for implementing the BS functions in common hardware (cloudification) [5]; and soon after [6] analyzed the pros and cons of different splits. A detailed study of the vRAN split specifications can be found in [7], while [8], [9] studied the cost-efficiency gains of splits, and [10] the effect of packetization. However, there are few works that optimize the split configuration, see [11]. The authors in [12] select splits to minimize inter-cell interference and fronthaul load, [13], [14] consider adaptive split configuration, [15] studied jointly the splitting and RAN slicing; and our previous work [16] proposed joint routing and split selection.

The above works do not consider the availability of multiple CUs, even less so the need to determine their deployment location. However, this is a key step in vRAN design. In [17], the authors consider a tree-like structure with pre-deployed CUs and explore, heuristically, minimum-cost splits; while [18] selects also the location of CUs and formulates (but does not solve) a min-cost design problem with fixed (non-optimized) splits. Finally, in our recent work [19] we assumed

multiple and fixed number of CUs but with predetermined DU-CU assignments. Unlike these interesting works, here we decide the deployment of CUs, determining also their number, and select which DUs each of them will serve and which splits will be realized. This complicates substantially the problem, but tackles a very important and practical design dimension in vRANs. Modeling-wise, the problem is reminiscent to server placement problems [20] and VNF chain embeddings [21]. However, here the placement of functions determines the data volume and transfer delay bounds, and hence which routing options are feasible.

C. Contributions

We introduce an analytical framework by deciding: the number of CUs, their deployment locations, the DUs that each of them should serve, the split for each BS, and the routing path for each data flow. We formulate a mathematical problem that optimizes jointly these decisions by using a measurement-based 3GPP-compliant system model. The objective is to minimize the vRAN operational expenditures, including function implementation and data routing costs. Our framework is general enough and can be tailored to different system architectures or include other objective criteria such as the desirable centralization level (see [19] for definition).

The resulting formulation is a rich mixed-integer quadratic linear problem (MIQLP), with prohibitive complexity in case of large networks and multiple CU locations. Therefore we propose a novel two-stage solution process. First, we transform it to an MILP by linearizing its constraints [22], and then employ an intelligent cutting-planes method based on the seminal Benders' decomposition technique [23] to find an exact solution. This is crucial since in such network design problems any optimality gap will induce multiplicative (with time) performance and cost losses.

We evaluate our framework using measurements for the system parameters and operational RAN topologies taken from [16]. Following an extensive parameter-sensitivity analysis we find that the benefits when departing from single-CU deployments can be as high as 30%, when using 8 CUs, or 20%, for 4 CUs in these systems, but these gains diminish as we add more CUs. This reveals a threshold effect, and the need to optimize the placement decisions in order to avoid excessive deployment costs. Finally, we find that a multi-CUs vRAN can support higher level of centralization (more functions at the CUs) even for high routing costs and/or DU traffic loads.

II. PRELIMINARIES AND MODEL

Background. 3GPP introduced eight different splits but the main standardized ones are as follows [6], [13]:

- Split 1 (S1; PDCP-RLC): RRC, PDCP, and upper layers deployed at CU; RLC, MAC, and PHY at the DU.
- Split 2 (S2; MAC-PHY): MAC and upper layers at CU; PHY and RF layers at the DU.
- Split 3 (S3; PHY-RF): All functions at CU except RF layers.

Going from S1 to S3, more functionalities are centralized, which increases the cost savings (economics of scale) and performance gains, e.g., S1 supports CoMP and effective
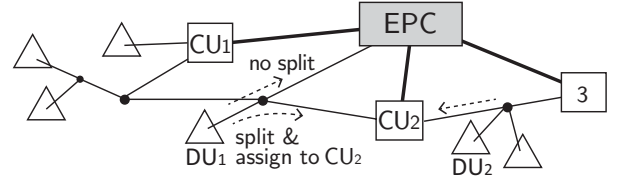

Fig. 1: The BS functions can be split between the DU and one of the CUs. DU-1 can implement a full-stack BS (no split) and route its traffic directly to EPC; or select a split and send its traffic to CU-2 which further routes it to EPC. CUs have high-capacity links to EPC; and some CUs might not be activated (e.g., 3).

MIMO implementation. However, centralizing more functions increases the volume of data that needs to be transferred to CU (can reach higher than 2.5Gbps in S3), and imposes tighter transfer delay constraints (0.25msec in S3), see [24].

The BS implements a function chain $f_0 \rightarrow f_1 \rightarrow f_2 \rightarrow f_3$ [7], where f_0 encapsulates the RF-related operations (e.g., A/D sampling) and is always placed at DUs, while f_3 is associated with PDCP and upper layers. Functions f_1 (PHY) and f_2 (RLC and MAC) can be deployed either in CUs or DUs depending on the RAN configuration. The current standardization efforts suggest a packet-based network with links that are shared by the DUs, instead of the point-to-point expensive CPRI links. Following the latest proposals, we assume that the CUs are directly connected to the network core through high-capacity (e.g., optical) links; see Fig. 1 for an example of our system.

Network. The RAN is modeled with a graph $G = (\mathcal{I}, \mathcal{E})$ where the set of nodes $\mathcal{I}$ includes the subsets: $\mathcal{N}$ of N DUs, $\mathcal{L}$ of L routers, and $\mathcal{M}$ of M possible locations for the CUs ($M << N$); and the core node (EPC) that we index with 0. We also define $\mathcal{M}_0 = \mathcal{M} \cup \{0\}$. The nodes are connected with the $\mathcal{E}$ links and each link (i, j) has average data transfer capacity of c_{ij} (b/sec). DU-n is connected to each CU-m with paths $\mathcal{P}_{nm}$, and to EPC with paths $\mathcal{P}_{n0}$. We define the set $\mathcal{P}_m = \cup_{n=1}^{N} \mathcal{P}_{nm}$ of paths connecting all DUs to CU-m; and the set of all paths $\mathcal{P} = \cup_{m \in \mathcal{M}_0} \mathcal{P}_m$. Each path p_k introduces end-to-end delay of d_{p_k} secs. The BS functions are implemented in servers using virtual machines (VMs). We denote with H_n and H_m (cycles/s) the processing capacity of DU-n and CU-m, respectively; and define as ρ_1 and ρ_2 (cycles/Mb/s) the processing load of functions f_1 and f_2 (per unit of traffic).

Demand & Cost. Without loss of generality, we focus on uplink. The users served by each DU-n, $n \in \mathcal{N}$, generate requests following an i.i.d random process $\{\Phi_n(t)\}_{t=1}^{\infty}$ that is uniformly upper-bounded by ϕ_n. This creates a data flow $\lambda_n = \lambda \phi_n$, where λ models the bytes per request. Hence, the RAN needs to admit, route and serve N different flows. The function execution is more cost-effective at the CUs [8]. Therefore, we denote with a_m and b_m the computing and instantiating VM cost (monetary units/cycle) at CU-m, and define vectors $\boldsymbol{a} = (a_m, m \in \mathcal{M})$, $\boldsymbol{b} = (b_m, m \in \mathcal{M})$. The respective costs for the N DUs are $\boldsymbol{\alpha} = (\alpha_n, n \in \mathcal{N})$ and $\boldsymbol{\beta} = (\beta_n, n \in \mathcal{N})$. Finally, ζ_k is the routing cost (monetary units/byte) for each path $p_k \in \mathcal{P}$, and we define $\boldsymbol{\zeta} = (\zeta_1, \ldots, \zeta_{|\mathcal{P}|})$. Such costs might arise because the links are leased by third-parties, or

they can model the (average) operating expenditures of the network. The flows are routed to CUs and then to the EPC through high-capacity links, or directly from the DUs to the EPC in case of no splitting.

Problem Definition. Our objective is to minimize the network operation cost while serving the user traffic, by jointly optimize the following decisions:

1) *Deployment*: which of the available CU locations to use?
2) *Assignment*: which CU should serve each (split) DU?
3) *Placement*: where to deploy f_1, f_2, f_3 for each BS?
4) *Routing*: How to route the data from the DUs to CUs?

These decisions are inherently coupled and this raises interesting trade-offs. Placing the functions at the DUs reduces the routing cost but increases the computing cost due to inefficient DU servers. On the other hand, centralizing the functions reduces computing costs at the expense of higher routing expenditures, while the deep splits restrict the routing options due to their tight delay bounds. The CU assignment decisions affect directly the routing costs and the number of available paths. Finally, the links are shared, hence the above decisions are coupled across the different DUs.

III. PROBLEM FORMULATION

Function Placement. We denote with $x_{1n}, x_{2n} \in \{0,1\}$ the decisions for deploying f_1 and f_2, respectively, to DU-n. Similarly, $y_{1nm}, y_{2nm} \in \{0,1\}$ decide the deployment of these functions to CU-m. We define the function placement vectors as $\boldsymbol{x_1} = (x_{1n} : n \in \mathcal{N}), \boldsymbol{x_2} = (x_{2n} : n \in \mathcal{N})$ for all DUs, and $\boldsymbol{y_{1m}} = (y_{1n,m} : n \in \mathcal{N}), \boldsymbol{y_{2m}} = (y_{2n,m} : n \in \mathcal{N})$ for CU-m. We further define $\boldsymbol{y_1} = (\boldsymbol{y_{1m}} : m \in \mathcal{M})$ for f_1, $\boldsymbol{y_2} = (\boldsymbol{y_{2m}} : m \in \mathcal{M})$ for f_2, and $\boldsymbol{x} = (\boldsymbol{x_1}; \boldsymbol{x_2}), \boldsymbol{y} = (\boldsymbol{y_1}; \boldsymbol{y_2})$.

The function placements are coupled. Namely, f_1 cannot be deployed at a CU unless f_2 is also placed there; while f_2 can be deployed at an DU only after f_1 [7], [16], [25], i.e.,

$$y_{1nm} \leq y_{2nm}, \quad \forall n \in \mathcal{N}, m \in \mathcal{M}, \qquad (1)$$
$$x_{2n} \leq x_{1n}, \qquad \forall n \in \mathcal{N}. \qquad (2)$$

Also, duplicate deployments should be prevented, i.e.,

$$x_{1n} + \sum_{m \in \mathcal{M}} y_{1nm} = 1, \quad x_{2n} + \sum_{m \in \mathcal{M}} y_{2nm} = 1, \ \forall n \in \mathcal{N}. \ (3)$$

Finally, the placements need to ensure that the computing capacity at each location is satisfied, hence:

$$\lambda_n \big(x_{1n}\rho_1 + x_{2n}\rho_2 + \rho_3(1 - \sum_m z_{nm}) \big) \leq H_n, \forall n \in \mathcal{N}, \ (4)$$

$$\sum_{n \in \mathcal{N}} \lambda_n (y_{1nm}\rho_1 + y_{2nm}\rho_2 + z_{nm}\rho_3) \leq H_m, \forall m \in \mathcal{M}. \ (5)$$

CU Deployment. We assign DU-n to CU-m using the binary variable z_{nm}, where $z_{nm} = 1$ if at least one function of BS-n is deployed at that CU. The assignment matrix is $\boldsymbol{z} = (z_{nm} \in \{0,1\} : \forall n \in \mathcal{N}, \forall m \in \mathcal{M})$. We model the configuration where f_1 and f_2 are at the DU-n, but f_3 at the CU-m, by setting $y_{1m} = y_{2m} = 1 - z_{nm} = 0$, hence we do

not define explicit variables for f_3. Without loss of generality, each DU can be assigned at most to one CU:

$$\sum_{m \in \mathcal{M}} z_{nm} \leq 1, \forall n \in \mathcal{N}, \qquad (6)$$

$$\text{and} \quad z_{nm} \geq y_{2nm}, \ \forall n \in \mathcal{N}, \forall m \in \mathcal{M}, \qquad (7)$$

must hold to preserve the function deployment ordering.

Data Routing. Variable r_{nm}^k (Mb/s) decides the DU-n traffic routed over path $p_k \in \mathcal{P}_{nm}$ to CU-m, and we define $\boldsymbol{r} = \big(r_{nm}^k \in \mathbb{R}_+ : \ \forall p_k \in \mathcal{P}_{nm}, n \in \mathcal{N}, m \in \mathcal{M}_0 \big)$. The routing decisions must respect the link capacities

$$\sum_{n \in \mathcal{N}} \sum_{p_k \in \mathcal{P}_n} r_{nm}^k I_{ij}^k \leq c_{ij}, \quad \forall (i,j) \in \mathcal{E} \qquad (8)$$

where it is $I_{ij}^k = 1$ if link (i,j) is included in path p_k; and also have to satisfy the flow and delay requirements of the splits:

$$\sum_{p_k \in \mathcal{P}_{nm}} r_{nm}^k = z_{nm} S_n(x_{1n}, x_{2n}), \ \forall n \in \mathcal{N}, \forall m \in \mathcal{M}, \quad (9)$$

where $S_n(x_{1n}, x_{2n})$ is the data flow (Mb/s) from DU-n which is determined by the split and user traffic of BS-n,

$$S_n(x_{1n}, x_{2n}) = x_{1n}(1.02\lambda_n + 1.5)$$
$$- x_{2n}(0.02\lambda_n + 1.5) + 2500(1 - x_{1n}).$$

(9) ensures there is no data flow from DU-n to CU-m unless the BS-m f_3 is placed at that CU, namely n is assigned to m. This captures nicely the interaction between assignment and routing, but creates a quadratic constraint term.

Note that in case of fully decentralized BSs (f_3 at the DUs), the flow needs to be routed directly to the core:

$$\sum_{p_k \in \mathcal{P}_{n0}} r_{n0}^k = \left(1 - \sum_{m \in \mathcal{M}} z_{nm} \right) \lambda_n, \ \forall n \in \mathcal{N}. \qquad (10)$$

Delay. Variable r_{nm}^k has to satisfy the delay requirements of the selected BS split [7]. To enforce this, we first classify the paths into three categories: $\mathcal{P}_{nm}^A \subseteq \mathcal{P}_{nm}$ with delay larger than 30 ms; the paths $\mathcal{P}_{nm}^B \subseteq \mathcal{P}_{nm}$ with delay larger than 2ms; and paths $\mathcal{P}_{nm}^C \subseteq \mathcal{P}_{nm}$ with delay larger than 0.25ms. Clearly, it holds $\mathcal{P}_{nm}^A \subseteq \mathcal{P}_{nm}^B \subseteq \mathcal{P}_{nm}^C$. We can ensure that only the eligible for each split paths are selected by using the constraints:

$$\sum_{p_k \in \mathcal{P}_{nm}^A} r_{nm}^k \leq T(y_{1nm} + y_{2nm}), \forall n \in \mathcal{N}, m \in \mathcal{M}, \qquad (11)$$

$$\sum_{p_k \in \mathcal{P}_{nm}^B} r_{nm}^k \leq T(1 - y_{1nm} + y_{2nm}), \forall n \in \mathcal{N}, m \in \mathcal{M}, \ (12)$$

$$\sum_{p_k \in \mathcal{P}_{nm}^C} r_{nm}^k \leq T(2 - y_{1nm} + y_{2nm}), \forall n \in \mathcal{N}, m \in \mathcal{M}, \ (13)$$

where $T \gg 0$ is used to enforce the logical coupling of the delay requirements of the splits and the eligible paths.

Objective. We wish to minimize the deployment, computation and routing costs. The computation cost for DU-n is

$$V_n(\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{z}) = \alpha_n \big(x_{1n} + x_{2n} + (1 - \sum_{m \in \mathcal{M}} z_{nm}) \big)$$

$$+ (\beta_n \lambda_n)\big(\rho_1 x_{1n} + \rho_2 x_{2n} + \rho_3(1 - \sum_{m \in \mathcal{M}} z_{nm})\big), \quad (14)$$

and the respective cost for CU-m is

$$V_m(\boldsymbol{y}_{1m}, \boldsymbol{y}_{2m}, \boldsymbol{z}_m) = b_m \sum_{n \in \mathcal{N}} \lambda_n \left(\rho_1 y_{1nm} + \rho_2 y_{2nm} + \rho_3 z_{nm}\right)$$

$$+ a_m \sum_{n \in \mathcal{N}} (y_{1nm} + y_{2nm} + z_{nm}) + \omega_m \sum_{n \in \mathcal{N}} z_{nm} \lambda_n. \quad (15)$$

where ω_m is the cost for using the CU and routing the data from there to the EPC. Note that when it holds $\sum_n z_{nm} = 0$ it means that CU-m is not used. The cost of routing data from the all DUs to CU-m is:

$$U_m(\boldsymbol{r}_m) = \sum_{n \in \mathcal{N}} \sum_{p_k \in \mathcal{P}_{nm}} \zeta_k r_{nm}^k. \quad (16)$$

Putting the above together, we can introduce the mathematical program $\mathbb{P}_1$ that minimizes the vRAN cost:

$$\boxed{\begin{aligned} \mathbb{P}_1 : \min_{\boldsymbol{r}, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}} \quad & \sum_{n \in \mathcal{N}} V_n(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{z}) + \sum_{m \in \mathcal{M}} V_m(\boldsymbol{y}_{1m}, \boldsymbol{y}_{2m}, \boldsymbol{z}_m) \\ & + \sum_{m \in \mathcal{M}_0} U_m(\boldsymbol{r}_m) \\ \text{s.t.} \quad & (1) - (13) \end{aligned}}$$

$\mathbb{P}_1$ includes integer, continuous variables, and constraints (9) involve quadratic terms (variables multiplications). Hence, it is a challenging mixed-integer quadratic program (MIQCP)[1].

IV. SOLUTION FRAMEWORK

We follow a two-stage solution approach. First, we reformulate $\mathbb{P}_1$ using a linearization technique that replaces the intricate constraint (9). Then, we decompose the problem and employ an efficient cutting-planes method that expedites the solution and finds, provably, an *exact optimal* point.

A. Linearization of constraints

The product of two integer variables in (9) can be represented by introducing auxiliary variables for every pair (n, m): $v_{1n,m} = x_{1n} z_{nm}$ and $v_{2n,m} = x_{2n} z_{nm}$, where belong to set:

$$\begin{aligned} \mathcal{V} = \Big\{ & \boldsymbol{v_1} : v_{1nm} \in \{0,1\}, \ \boldsymbol{v_2} : v_{2nm} \in \{0,1\}, n \in \mathcal{N}, m \in \mathcal{M} \\ & \mid v_{1nm} \leq x_{1n}; v_{1nm} \leq z_{nm}; \ v_{1nm} \geq x_{1n} + z_{nm} - 1; \\ & v_{2nm} \leq x_{2n}; \ v_{2nm} \leq z_{nm}; v_{2nm} \geq x_{2n} + z_{nm} - 1 \Big\}, \end{aligned}$$

using a reformulation similar to [22], we define the problem:

$$\boxed{\begin{aligned} \mathbb{P}_2 : \quad & \min_{\substack{\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{v} \in \mathcal{V}, \\ \boldsymbol{y}, \boldsymbol{r} \succeq 0}} \quad J_F(\boldsymbol{r}, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{v}) \\ & \text{s.t.} \quad (1) - (8), (10) - (13) \\ & \sum_{p_k \in \mathcal{P}_{nm}} r_{nm}^k = S_{nm}(v_{1nm}, v_{2nm}), \ \forall n \in \mathcal{N}, m \in \mathcal{M} \end{aligned}}$$

J_F is the objective of $\mathbb{P}_1$, and we set $S_{nm}(v_{1nm}, v_{2nm}) = z_{nm} S_n(x_{1n}, x_{2n})$. $\mathbb{P}_2$ is equivalent to problem $\mathbb{P}_1$.

[1]Reduction from the multidimensional-knapsack problem; see also [16].

B. Decomposition

We use the Benders' method [23], which decomposes $\mathbb{P}_2$ to a *Master* sub-problem $\mathbb{P}_{2M}$ that optimizes the binary variables for fixed routing; and to a *Slave* program $\mathbb{P}_{2S}$ that optimizes routing for fixed split and assignment decisions:

$$\begin{aligned} \mathbb{P}_{2S} : \quad & \min_{\boldsymbol{r} \succeq \boldsymbol{0}} J_F(\boldsymbol{r}, \bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}, \bar{\boldsymbol{v}}) \\ & \text{s.t.} \quad (8), (10) - (13) \\ & \sum_{p_k \in \mathcal{P}_{nm}} \bar{r}_{nm}^k = S_{nm}(\bar{v}_{1nm}, \bar{v}_{2nm}), \quad \forall n \in \mathcal{N}, m \in \mathcal{M}. \end{aligned}$$

Following the standard practice in Benders techniques, we will be using the dual Slave problem:

$$\mathbb{P}_{2SD} : \quad \max_{\boldsymbol{\pi}} \ h(\boldsymbol{\pi}, \bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}, \bar{\boldsymbol{z}}, \bar{\boldsymbol{v}}) \quad \text{s.t.} \quad H^\top \boldsymbol{\pi} \preceq \boldsymbol{\zeta}, \quad (17)$$

where $\boldsymbol{\pi}$ is the vector of dual variables and matrix H collects the necessary coefficients.

The Master problem $\mathbb{P}_{2M}$ optimizes the discrete decisions and a proxy continuous variable $\theta \geq 0$:

$$\begin{aligned} \min_{\theta, \boldsymbol{x}, \boldsymbol{z}, \boldsymbol{v} \in \mathcal{V}, \boldsymbol{y}} \quad & J_F(\bar{\boldsymbol{r}}, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{v}) + \theta \\ \text{s.t.} \quad & (1) - (8) \\ & h(\boldsymbol{\pi}^\xi, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{v}) \leq \theta, \ \forall \boldsymbol{\pi}^\xi \in \mathcal{C}_O \quad (18) \\ & h(\boldsymbol{\pi}^\xi, \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{v}) \leq 0, \ \forall \boldsymbol{\pi}^\xi \in \mathcal{C}_F \quad (19) \end{aligned}$$

where (18)-(19) are the optimality and feasibility cuts, respectively, which gradually construct the entire constraint set of $\mathbb{P}_2$. The intuition behind this method is that the optimal solution can be found before a full re-construction is built.

C. Algorithm

The detailed process is described in Algorithm 1. It finds the optimal solution iteratively. Firstly, it solves the Master problem ($\mathbb{P}_{2M}$) to find the currently optimal integer decision variables $(\boldsymbol{x}^\tau, \boldsymbol{z}^\tau, \boldsymbol{v}^\tau, \boldsymbol{y}^\tau)$ and surrogate variable (θ^τ) for every iteration τ (Step 2). These values are used to set the current lower bound $LB^{(\tau)}$ (Step 3). Then, we can solve ($\mathbb{P}_{2SD}$) and get $\boldsymbol{\pi}^\tau$ by using $\boldsymbol{x}^\tau, \boldsymbol{z}^\tau, \boldsymbol{v}^\tau, \boldsymbol{y}^\tau$ (Step 4). Then, using the Master problem, we can obtain a new upper bound (Step 5-7). In every iteration, $\mathcal{C}^O$ (if $\mathbb{P}_{2SD}$ is bounded) and $\mathcal{C}_F$ (if $\mathbb{P}_{2SD}$ is unbounded) are enriched with new cuts (Step 8-12); and will be used to solve $\mathbb{P}_{2M}$ in the next iteration. The steps are repeated until upper and lower bound coincide, and reach optimal solution, $UB - LB \longrightarrow 0$ (Step 14).

Algorithm 1 finds, provably, the solution of $\mathbb{P}_2$ (exact if we set $\epsilon = 0$.), as the following theorem states.

Theorem 1. *Algorithm 1 converges in a finite number of iterations to the optimal solution of problem (P2).*

Proof. We prove that the conditions of Theorem 6.3.4 in [26] hold for our case.

1) $\boldsymbol{r}$ is a non-empty and convex set since the domain of every r_{nm}^k is in $\mathbb{R}_+$.
2) $J_F(\boldsymbol{r}, \bar{\boldsymbol{x}}, \bar{\boldsymbol{y}}, \bar{\boldsymbol{v}})$ and (11)-(13) are convex; and (9)-(10) are linear.

Algorithm 1: Decomposition Algorithm

Initialize:
$\tau = 0; \mathcal{C}_O^{(0)} = \mathcal{C}_F^{(0)} = \emptyset; UB^{(0)} = -LB^{(0)} >> 1; \epsilon$

1 **repeat**
2   Solve $\mathbb{P}_{2M}(\mathcal{C}_O^\tau, \mathcal{C}_F^\tau)$ to get $\theta^\tau, \boldsymbol{x^\tau}, \boldsymbol{y^\tau}, \boldsymbol{z^\tau}, \boldsymbol{v^\tau}$
3      $LB^\tau =$
         $\sum_n V_n(\boldsymbol{x_1^\tau}, \boldsymbol{x_2^\tau}, \boldsymbol{z^\tau}) + \sum_m V_m(\boldsymbol{y_{1m}^\tau}, \boldsymbol{y_{2m}^\tau}, \boldsymbol{z_m^\tau}) + \theta^\tau$
4   Solve $\mathbb{P}_{2SD}(\boldsymbol{x^\tau}, \boldsymbol{y^\tau}, \boldsymbol{z^\tau}, \boldsymbol{v^\tau})$ to obtain $\boldsymbol{\pi^\tau}$.
5   **if** $UB^\tau < UB^{\tau-1}$ **then**
6          $UB^\tau =$
           $\sum_n V_n(\boldsymbol{x_1^\tau}, \boldsymbol{x_2^\tau}, \boldsymbol{z^\tau}) + \sum_m V_m(\boldsymbol{y_{1m}^\tau}, \boldsymbol{y_{2m}^\tau}, \boldsymbol{z_m^\tau}) +$
           $h(\boldsymbol{\pi^\tau}, \boldsymbol{x^\tau}, \boldsymbol{y^\tau}, \boldsymbol{z^\tau}, \boldsymbol{v^\tau})$;
7   **end**
8   **if** $h(\boldsymbol{\pi^\tau}, \boldsymbol{x^\tau}, \boldsymbol{y^\tau}, \boldsymbol{z^\tau}, \boldsymbol{v^\tau}) < \infty$ **then**
9          $\mathcal{C}_O^{\tau+1} = \mathcal{C}_O^\tau \cup \{\boldsymbol{\pi^m}\}$ % add extreme point;
10   **else**
11          $\mathcal{C}_F^{\tau+1} = \mathcal{C}_F^\tau \cup \{\boldsymbol{\pi^m}\}$.   % add extreme ray;
12   **end**
13      $\tau = \tau + 1$.
14 **until** $UB^{(\tau)} - LB^{(\tau)} \leq \epsilon$
15 Optimal configuration, assignment, deployment:
     $\boldsymbol{x^*} = \boldsymbol{x^\tau}; \boldsymbol{y^*} = \boldsymbol{y^\tau}, \boldsymbol{z^*} = \boldsymbol{z^\tau}$
16 Optimal routing $\boldsymbol{r^*}$ from $\mathbb{P}_{SD}(\boldsymbol{x^\tau}, \boldsymbol{y^\tau}, \boldsymbol{z^\tau}, \boldsymbol{v^\tau})$

---

3) $\boldsymbol{r}$ is closed and bounded by (8), and each r_{nm}^k has a real value (RU-n traffic) in $\mathbb{R}_+$, so the constrains are also continuous for every fixed binary variable.

4) We have finite number for all parameters and r is bounded by (8), so the solution is also finite. The inequalities and equalities are set at least there is a feasible solution, so, the first-order constraint qualification also holds for the condition and there exist optimal Langrangian multipliers.

Hence, all conditions of the theorem are satisfied and the convergence does not depend on the initialization. $\quad\square$

V. RESULTS AND DISCUSSION

In this section we present a battery of numerical tests using real datasets and topologies, aiming to examine the:

- optimal multi-CUs vRAN configurations in real networks;
- benefits of deploying multiple CUs instead of only one;
- cost-effectiveness of optimizing the CU locations and DU-CU assignments, compared to non-optimized deployments.
- the effect of routing cost and DU traffic on the cost and centralization level for a single and multiple CUs.

A. Network Topology and Evaluation Setup

We evaluate our model in the actual networks taken from [16] of Fig. 2. RAN N1 consists of a core node, 198 DUs, 15 CU candidate locations, and routers. RAN N2 has a core node, 197 DUs, 15 CU candidate locations, and routers. The network parameters such as delay, location, distance, and link capacities are derived from the actual data or using other measurement studies. We pre-calculate the DU-CU candidate paths by applying the k-th shortest path algorithm [27]. The distance of DUs to the CUs and the core network ranges from
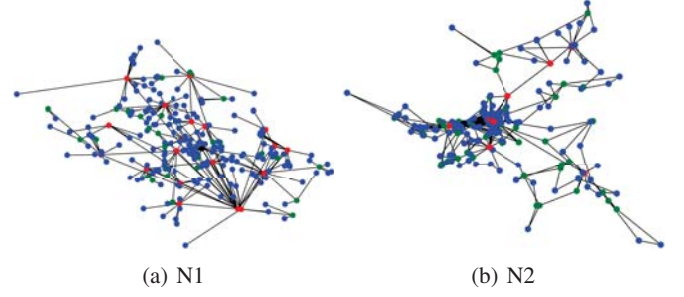

(a) N1   (b) N2

Fig. 2: Two actual operational RANs [16]. Black, blue, green, and red color dots represent the core network, DUs, routers, and CU's candidates, respectively. The RANs are visualized according to its coordinate location (longitude and latitude).

0.1km to 25km and the respective path delays vary up to $257.61\mu s$ (N1) and $1152.69\ \mu s$ (N2). As candidate locations for deploying CUs, we have selected the network nodes with the highest network degree (more central locations)[2].

We set the system parameters according to actual testbed measurements and previous works [8], [16], [25]. The default DU load is $\lambda_n = 150$ Mbps for each DU[3]. For CPU capacity, we use a *reference core* (RC), Intel i7-4770 3.4GHz, and set the maximum computing capacity to 75 RCs for each CU, and to 2 RCs for each DU. We assume that the default cost of CU-m VM instantiation is a half[4]. of DU-n ($a_m = \alpha_n/2$) [8], [16], [25] but we also explore the impact of different ratios. The CU processing cost is set to $b_m = 0.017\beta_n$ according to our measurements in [25]. Then, the CU deployment cost is assumed at least 30 times higher of the CU processing cost. This value is calculated based on the comparison server setup price ($20K) and data processing cost ($653.54) [8]. Finally, the routing cost per path grows linearly with distance, $\zeta = c_d \times d_{km}$, where c_d is the cost per Km and captures how expensive is each link (can be different for each network).

B. Increasing the CU candidate locations

Our first experiment increases gradually the available CU locations and studies how this impacts the system cost (in monetary units) and the RAN design. The deployment cost per CU is higher as we add more locations, i.e., $\omega_m \leq \omega_{m+1}$, which is intuitive as, naturally, the cheapest locations are selected first. Fig. 3a shows that there are $6.65\%, 8.79\%, 13.10\%$, and 16.36% cost savings on average by adding more candidate location ($M = 13$) compared to just a single location in N1, for various scenarios of routing cost and CU/DU computing efficiency gains. Similar findings hold for N2, Fig. 3b. The trend of cost reduction continues as the candidate CU locations increase, albeit the additional gains are fast decreasing and we reach a state (after $M = 13$ in this network) where adding more CU locations does not bring further cost savings.

---

[2]These networks do not have CUs, thus we followed this intuitive approach to select candidate locations that we then feed to our optimization framework.
[3]This value corresponds to 2×2 MIMO, 1 user/TTI, 20 Mhz, 2 TBs of 75376 bits/subframe and IP MTU 1500B.
[4]This is based on the typical cost of C-RAN BSs that is half of D-RAN BSs, both for macro ($50K and $25K) and micro BS($20K and $10K)
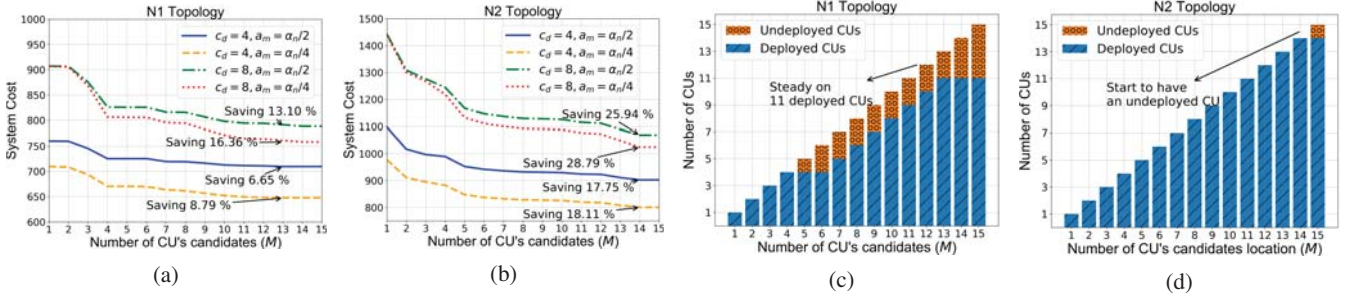
Fig. 3: Impact of increasing locations to: (a-b) system cost; (b-c) the number of actually deployed CUs.
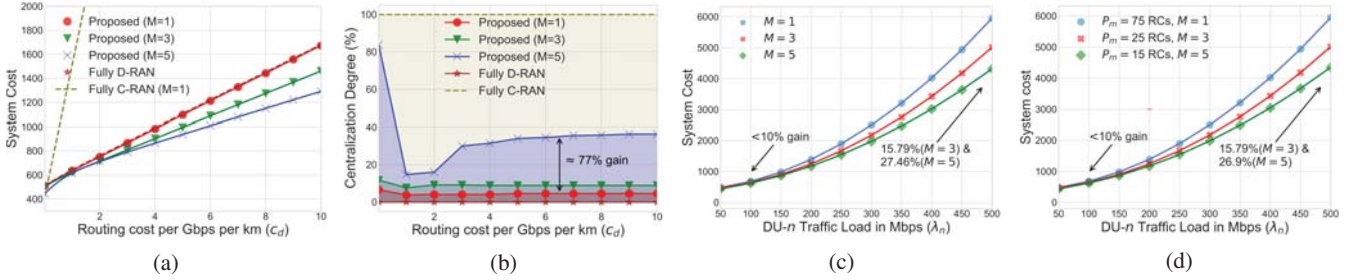

Fig. 4: Impact of the routing cost to (a) system cost; (b) centralization degree. (c)-(d) Impact DU traffic load to system cost.

Figures 3c-3d show the distribution of deployed CUs and non-deployed ones, i.e., available locations that were not used by our solution. In N1, the deployment rate (portion of available CU locations that are actually deployed) is 100% when the candidate locations are up to $M = 4$. Beyond that, the deployment decreases and reaches a constant value with 11 deployed CUs when the available locations are $M = 13$. However, in N2, we have non-deployed CUs after $M = 14$ which shows that, despite following a similar trend, the deployment decision is profoundly affected by the specific candidate locations and network structure. These results are representative as we have used real networks.

Findings: *1) As the CU available locations increase, our framework saves significant costs (16.36% in N1 and 28.79% in N2), before the gains diminish. 2) The higher routing costs and lower compute costs of CUs lead to substantial benefits. 3) Network characteristics affect the number of used CUs.*

C. Impact of Routing Cost and Traffic

We study the effect of routing cost on the system cost and the centralization that our solution achieves i.e., number of functions deployed at the CUs. Aside from comparing single and multiple CU candidate locations, we also benchmark against C-RAN and D-RAN solutions (two extreme cases). In fully C-RAN, all functions except f_0 are placed at the CU, while in D-RAN they are all deployed at the DU. In this case, the deployment cost of CU is set to zero ($\omega_m = 0$) for D-RAN (we do not need to deploy any CU), and the traffic load of each DU is directly routed to the core network. In this experiment, the routing cost (/Km) increases from $c_d = 0.01$ (very low) to $c_d = 10$ (very high). Note that networks N1 and N2 cannot support fully C-RAN solutions, hence the results for this configuration are hypothetical and presented for reference.

Fig. 4a compares the system cost of our framework to fully C-RAN and D-RAN. The C-RAN cost increases significantly with the routing cost c_d, and eventually exceeds the cost of D-RAN (for $c_d \approx 0.01$ per Gbps). For a single CU location and the specific network parameters[5], the optimized configuration is just slightly better than D-RAN ($0 - 1\%$) because the CU location is near to the core and far from the DUs, thus most functions cannot be deployed in CU, Fig. 4b. By considering more candidate locations we can obtain approximately 13.10% ($M = 3$) and 23.15% ($M = 5$) cost savings at $c_d = 10$. Fig. 4b shows the impact of the routing cost to function centralization[6]. Overall, with $M = 5$ locations we can maintain the centralization compared to the lower candidates. The result shows around $6.2 - 72.09\%$ and $10.9 - 77.16\%$ centralization gains at routing cost $0.01 - 10$ per Gbps/Km distance.

Next, we evaluate the impact of the DU-n traffic load on the system cost. We consider two scenarios. Firstly, the computational capacity of each CU is set with the same value, 75 RCs. Secondly, the aggregate of all CUs capacity is set to be $P_{tot} = 75$ RCs. Consequently, every CU has different capacity, i.e., $P_m = P_{tot}/M$, in each scenario as M changes. Our goal here is to understand which design option is preferable: to have a single CU location with very high capacity server, or multiple smaller CUs with lower capacity?

---

[5]This result is affected by, e.g., the cost of deploying a CU, the distance of this location compared to the core, etc. In general, even a single CU location can significantly reduce costs, see [16] for examples.

[6]Percentage of all BS functions that are deployed at CUs; it is 100% for fully C-RAN and 0% for D-RAN.
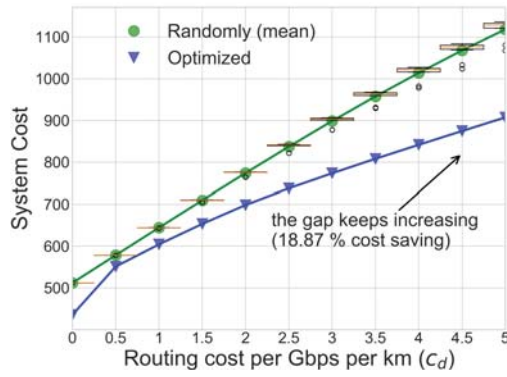
Fig. 5: The comparison of randomly and optimized (proposed) deployment as the routing cost increases.

We use the same deployment and VM computational cost for both scenarios. Fig. 4c clearly shows that the multiple CUs scenario performs better. Indeed, we can gain around 15.79% ($M = 3$) and 27.46% ($M = 5$) cost reduction at 500Mbps, when we consider three ($M = 3$) and five CU locations ($M = 5$) compared to a single CU (total computing capacity is the same in both cases). The gap is getting higher as the traffic load increases. In addition, Fig. 4d shows that having multiple candidate locations yields extra saving of 15.79% ($M = 3$) and 26.9% ($M = 5$) compared to a single candidate, even when the capacity of servers is smaller.

Findings: *1) The cost-savings of having multiple CUs is getting higher in line with the increase of the routing cost and the traffic load. 2) More CU locations can maintain a higher centralization degree (77% gain). 3) Having multiple CUs with lower capacity can achieve extra cost-savings (26.9%) than a single CU with the same aggregate capacity.*

D. Random vs Optimized Deployment of CUs

Finally, we compare the case where the CUs are deployed randomly instead of optimizing these decisions (as our algorithm does). We will show that it is crucial not only to use multiple CUs but to carefully optimize their placement. The first approach deploys the CUs in random locations (among those being available), while the second uses our framework and optimizes their position. In detail, we set the number of available CU positions to $M = 15$ and force to choose only 3 deployed CUs, by selecting randomly (10 iterations) or by optimizing the deployment. Fig. 5 clearly shows that our framework outperforms the randomly deployed scenario by 18.87%. These cost savings increase with the routing cost. Clearly, when it is more expensive to route data, such non-optimized random decisions will have higher negative impact.

Finding: *The gains of multi-CUs vRAN cannot be maximized unless their number and location is optimized (18.87% gain at $c_d = 5$) for the specific network and load.*

VI. CONCLUSION

There is currently a flurry of standardization and other industry activities, aiming to make vRAN the de facto solution for next generation access networks. To this end, our work fills an important gap as it tackles the vRAN design problem in its most general form. Using a standards-compatible system model, we develop a rigorous optimization approach that selects jointly the number and location of (multiple) CUs; assigns to them the DUs; and finds the optimal split levels and routing paths for each flow. Our framework is general and can be readily tailored to different scenarios, e.g., when the operator needs to enforce some level of centralization or when the cost functions are convex non-linear.

REFERENCES

[1] I. F. N. T. R. W. Group, "IEEE 5G and Beyond Technology Roadmap White Paper," IEEE, Tech. Rep., October 2017.
[2] N. Alliance, "5g white paper," Next Generation Mobile Networks (NGMN), Tech. Rep., October 2017.
[3] A. Checko *et al.*, "Cloud ran for mobile networks—a technology overview," *IEEE Comm. Surveys & Tutorials*, vol. 17, no. 1, 2015.
[4] Nokia, "Mobile anyhaul white paper," Nokia, Tech. Rep., 2017.
[5] Y. Lin, L. Shao, Z. Zhu, Q. Wang, and R. K. Sabhikhi, "Wireless Network Cloud: Architecture and System Requirements," *IBM Journal of Research and Development*, vol. 54, April 2010.
[6] U. Dötsch *et al.*, "Quantitative analysis of split base station processing and determination of advantageous architectures for lte," *Bell Labs Technical Journal*, vol. 18, no. 1, June 2013.
[7] S. C. Forum, "R6.0. small cell virtualization functional splits and use cases, document 159.07.02," Tech. Rep. Release 7, 2016, version 14.0.0.
[8] V. Suryaprakash, P. Rost, and G. Fettweis, "Are heterogeneous cloud-based radio access networks cost effective?" *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 10, Oct 2015.
[9] K. Garikipati, K. Fawaz, and G. Shin, "Rt-opex: Flexible scheduling for cloud-ran processing," in *Proc. of ACM CoNEXT*, 2016.
[10] C. Chang *et al.*, "Impact of packetization and functional split on c-ran fronthaul performance," in *Proc. of IEEE ICC*, 2016.
[11] L. M. P. Larsen, A. Checko, and H. L. Christiansen, "A survey of the functional splits proposed for 5g mobile crosshaul networks," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, 2019.
[12] D. Harutyunyan and R. Riggio, "Flex5g: Flexible functional split in 5g networks," *IEEE Trans. on Netw. & Service Man.*, vol. 15, no. 3, 2018.
[13] A. M. Alba, J. H. G. Velásquez, and W. Kellerer, "An adaptive functional split in 5g networks," in *IEEE INFOCOM Workshops*, 2019.
[14] Y. Alfadhli, M. Xu, S. Liu, F. Lu, P. Peng, and G. Chang, "Real-time demonstration of adaptive functional split in 5g flexible mobile fronthaul networks," in *Proc. of IEEE OFC*, 2018.
[15] B. Ojaghi, F. Adelantado, E. Kartsakli, A. Antonopoulos, and C. Verikoukis, "Sliced-ran: Joint slicing and functional split in future 5g radio access networks," in *Proc. of IEEE ICC*, 2019.
[16] A. G.-Saavedra, X. Costa-Perez, D. Leith, and G. Iosifidis, "Fluidran: Optimized vran/mec orchestration," in *Proc. of IEEE INFOCOM*, 2018.
[17] N. Mharsi, M. Hadji, D. Niyato, W. Diego, and R. Krishnaswamy, "Scalable and cost-efficient algorithms for baseband unit (bbu) function split placement," in *Proc. of IEEE WCNC*, 2018.
[18] O. Arouk *et al.*, "Cost optimization of cloud-ran planning and provisioning for 5g networks," in *Proc. of IEEE ICC*, 2018.
[19] A. G.-Saavedra *et al.*, "On the centralization degree of cloud ran next generation fronthaul," *IEEE Trans. Mob. Comp.*, vol. 17, no. 10, 2018.
[20] G. Rodolakis *et al.*, "Replicated server placement with qos constraints," *IEEE Trans. on Parallel and Distr. Sys.*, vol. 17, no. 10, 2006.
[21] R. Cohen, L. Eytan, J. Naor, and D. Raz, "Near optimal placement of virtual network functions," in *in Proc. of IEEE INFOCOM*, 2015.
[22] A. Gupte *et al.*, "Solving mixed integer bilinear problems using milp formulations," *SIAM Journal on Optimization*, vol. 23, no. 2, 2013.
[23] J. F. Benders, "Partitioning Procedures for Solving Mixed-Variables Programming Problems," *Numer. Math.*, vol. 4, 1962.
[24] 3GPP, "Study on new radio access technology: Radio access architecture and interfaces," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 38.801, 03 2017, version 14.0.0.
[25] A. G.-Saavedra *et al.*, "Joint optimization of edge computing architectures and radio access networks," *IEEE JSAC*, vol. 36, no. 11, 2018.
[26] C. A. Floudas, *Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications*. New York, N.Y.: Oxford University Press, 1995.
[27] J. Y. Yen, "Finding the k shortest loopless paths in a network," *Management Science*, vol. 17, no. 11, 1971.