On User Mobility in Dynamic Cloud Radio Access Networks

Diala Naboulsi, Assia Mermouri, Razvan Stanica, Hervé Rivano, Marco Fiore

On User Mobility in Dynamic Cloud Radio Access Networks

Diala Naboulsi*, Assia Mermouri†, Razvan Stanica†, Herve Rivano†, Marco Fiore‡†
*Concordia University, Montreal, Canada
†Université de Lyon, INSA Lyon, Inria, CITI, F-69621, Villeurbanne, France
‡CNR – IEIIT, Corso Duca degli Abruzzi 24, 10129 Torino, Italy
d_naboul@encs.concordia.ca, {assia.mermouri, razvan.stanica, herve.rivano}@insa-lyon.fr, marco.fiore@ieiit.cnr.it

***Abstract*—The development of virtualization techniques enables an architectural shift in mobile networks, where resource allocation, or even signal processing, become software functions hosted in a data center. The centralization of computing resources and the dynamic mapping between baseband processing units (BBUs) and remote antennas (RRHs) provide an increased flexibility to mobile operators, with important reductions of operational costs. Most research efforts on Cloud Radio Access Networks (CRAN) consider indeed an operator perspective and network-side performance indicators. The impact of such new paradigms on user experience has been instead overlooked. In this paper, we shift the viewpoint, and show that the dynamic assignment of computing resources enabled by CRAN generates a new class of mobile terminal handover that can impair user quality of service. We then propose an algorithm that mitigates the problem, by optimizing the mapping between BBUs and RRHs on a time-varying graph representation of the system. Furthermore, we show that a practical online BBU-RRH mapping algorithm achieves results similar to an oracle-based scheme with perfect knowledge of future traffic demand. We test our algorithms with two large-scale real-world datasets, where the total number of handovers, compared with the current architectures, is reduced by more than 20%. Moreover, if a small tolerance to dropped calls is allowed, 30% less handovers can be obtained.**

I. Introduction

Mobile traffic is currently growing at a higher rate than the one observed for web traffic during the "Internet bubble" [1]. Not only human users are moving towards greedier services, but also new machine-type communications, with entirely different traffic patterns, need to be accommodated on the mobile networks. This puts an increased pressure on the radio access network (RAN), which must cope with an ever-larger number of users presenting heterogeneous traffic demands.

Mobile operators are addressing this issue by developing more flexible and efficient RAN mechanisms, as well as through a densification of the access network. However, adding more base stations (BSs) comes at the price of increased interference and important financial costs: antennas, also known as remote radio heads (RRHs), need to be installed on a high point and connected to a baseband processing unit (BBU) nearby. A growing problem is the fact that the high computational load of the BBU requires the installation of a cooling system, increasing the energy consumption and further limiting the locations where a BS can be installed.

As a consequence, the centralized RAN concept appeared [2]. It commends that only RRHs are installed on site, and connected through an optical fiber link with their BBUs, which are instead gathered in a data center. With the parallel development of virtualization techniques, the Cloud RAN (CRAN) architecture emerged [3]. In CRAN, radio access functions such as signal processing, resource allocation or mobility management become software functions, running on any of the available computational resources [4]. The one-to-one mapping between RRH and BBU is therefore no longer needed. Multiple BBUs are implemented in a BBU pool and one BBU can handle several RRHs at a time. Moreover, since the load generated by an RRH varies with time, a dynamic mapping between RRHs and BBUs can further improve the usage of the computational resources [5].

Several major aspects of the CRAN architecture have been addressed in the last few years: the possible cost and energy savings [6], the limits imposed by the fronthaul connecting the antenna site and the data center [7], or the interference appearing between RRHs [8]. The CRAN implications on the user side, however, still remain to be understood. Therefore, the question we address in this paper is: *how does this new architecture, proposed with operator costs in mind, affect the mobile users?* More precisely, we focus on mobility management in CRAN, observing that user equipments (UEs) are associated on the RAN with a BBU, which can serve several RRHs. This means that an UE movement between two different RRHs does no longer result in a handover, if both RRHs are handled by the same BBU. While this indicates a reduction in the number of handovers, seemingly improving the user experience, we notice a new type of handover, specific to the CRAN environment and related to the dynamic mapping between BBU and RRH: when an RRH changes its BBU, all the users covered by that RRH have to change their BBU association as well. The question in this case is whether the overall number of handovers in the system decreases or not. The answer we give in this paper is that, through a careful design of the RRH-BBU mapping mechanism, the number of handovers can indeed be decreased, in addition to a consistent reduction in the number of BBUs used in the system. This work makes the following contributions:

i) We describe a new type of handover, specific to mobile networks with a virtualized radio access. This Reconfiguration Handover (RHO) appears when the software functions related to an RRH migrate from one BBU to another. From our

knowledge, this phenomenon has never been analyzed, or even observed, in the CRAN literature.

ii) We propose a time-varying graph model of the CRAN architecture, capturing the dynamics of BBU-RRH mapping. By creating clusters of nodes in this time varying graph, we study the evolution of the number of BBUs required and the number of handovers existing in the system.

iii) By modifying one of the most widely used community detection algorithms, the Louvain method [14], we obtain an oracle-type solution for the clustering problem in our time-varying graph. This gives us the BBU-RRH mapping minimizing the total number of handovers in a CRAN architecture. Our results show that the number of handovers can be reduced by more than 20% with respect to a non-CRAN approach.

iv) We propose an online solution for BBU-RRH mapping, taking into account only the recent past and a short-time prediction of future network activity. This practical online approach is shown to give results that are close to the oracle-based solution. Moreover, if a small additional call drop probability is allowed, the number of handovers can be reduced by more than 30% compared to a non-CRAN architecture.

In the remainder of this paper, we place our work in the context of the state of the art in Sec. II, followed by a description of handover management in CRAN in Sec. III. In Sec. IV, we present our time-varying graph CRAN model. We then introduce our oracle-type and online algorithms in Sec. V. Sec. VI presents our assumptions and dataset, while in Sec. VII and Sec. VIII, we evaluate the proposed strategies, before drawing our conclusions in Sec. IX.

II. Related Works

Several works have studied dynamic CRAN topology reconfiguration solutions, triggered by the initial study by Liu *et al.* [5], showing that, in a centralized RAN, the reconfiguration of the BBU-RRH mapping can find a satisfying trade-off between performance gains, in terms of throughput, and infrastructure costs. These conclusions are obtained through real-world experiments, which remain however small scale (4 RRHs and a pool of 4 BBUs). The benefits of dynamic reconfiguration of the CRAN at a larger scale have been studied analytically [9] and through simulation [10]. Both studies conclude that four times less BBUs are required in CRAN when compared with a classical RAN. Our results are in line with these findings, our oracle algorithm showing a reduction in the number of BBUs by 70% during the day and by up to 99% during night-time.

Other algorithms in the literature compute the BBU-RRH mapping in order to optimize resource allocation [11], to reduce the interference between neighboring RRHs [8], or to minimize the computational efforts of the BBU pool [6]. With the exception of this latter example, which uses 3G mobile traffic traces from 21 cell sites, the other examples in the literature focus on small scale scenarios (a few RRHs), using synthetic network traffic. We evaluate our BBU-RRH mapping strategies using two large-scale mobile traffic datasets covering the entire urban areas of Abidjan and Dakar. Moreover, we focus on the impact of CRAN on mobile users, by analyzing the number of handovers in the system.

As far as our knowledge goes, the only studies analyzing mobility management and handovers in CRAN are the ones by Liu *et al.* [12] and Sundaresan *et al.* [13]. However, in [12], the authors zoom in on the details of the handover mechanism in CRAN, showing that a centralized and virtualized RAN results in shorter connection interruptions and reduced signaling overhead. In this work, we take a complementary approach and show that the overall number of handovers can be reduced as well when moving to a CRAN architecture. As for the work in [13], it aims at selecting optimal BBU-RRH configurations with respect to the users mobility profiles: static or mobile users. Our work tackles a different problem. We aim at efficiently managing handovers in CRAN, in the presence of a novel type of handover, which we uncover.

III. Handovers in CRAN

The handover mechanism is one of the building blocks of a mobile network. It represents the procedure launched over the access network allowing a UE to change its association from one BBU to another. In traditional cellular networks, because of the one-to-one mapping between a BBU and an RRH and their colocation in the same BS, this also translates into changing the RRH that covers the UE. However, we underline the fact that the handover is a function of the Radio Resource Control (RRC) protocol, running on the UE and the BBU, without any participation of the RRH. A handover takes place only when the UE is connected at the RRC level, i.e. only when the user moves while accessing the mobile network.

During the handover procedure, ongoing UE packet flows can be either re-routed to the new BBU or lost. In both cases, the communication is disrupted, impacting the quality of experience (QoE) perceived by the UE. In fact, handovers have been observed to lead to a 10% increase in video session abandonment rates [15]. Web QoE has also been shown to be affected by handovers, with most web sessions abandonned in presence of handovers [16]. IP-level measurements confirm these results as well, showing disconnections that can reach tens of seconds [17]. Accordingly, handovers play a critical role on customers satisfaction and need to be properly handled.

With its virtualized architecture, CRAN is transforming traditional BS equipments, with a predefined fixed coverage, into dynamic BSs whose coverage area can change over space and time. This enables a high degree of flexibility when it comes to network management. For example, several RRHs with a reduced load could be mapped with the same BBU, sharing its resources (e.g. time slots, codes or resource blocks, depending on the deployed cellular technology). When the load associated with the RRHs becomes larger than the capacity of the BBU, a new BBU can be created in the BBU pool, and a part of the RRHs migrate to this newly created entity. Hence, a CRAN architecture can adapt the number of resources and the computation power to the network load but, at the same time, it can introduce some new phenomena in the mobile networks world. This is the case for handovers, whose

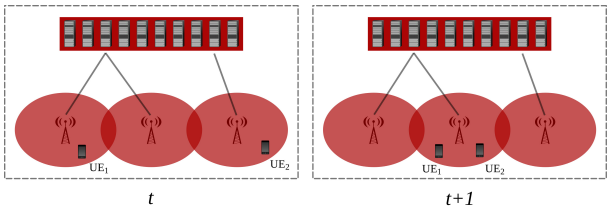


Fig. 1. Illustration of an MHO occurring between successive snapshots t and $t+1$. The mobile user UE_1 does not encounter an MHO, while UE_2 does.

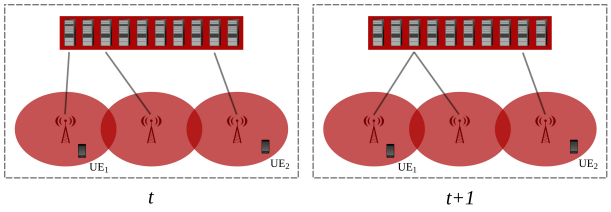


Fig. 2. Illustration of an RHO at successive snapshots t and $t+1$. Static user UE_1 encounters an RHO, while static user UE_2 does not.

concept is extended as a result of dynamic BS creation. In fact, a handover in a CRAN can be due to user movement, in which case we refer to it as a Mobility Handover (MHO), or it can be due to topology reconfiguration, in which case we refer to it as a Reconfiguration Handover (RHO).

We rely on Fig.1 to illustrate an MHO and explain how it differs from the case of a traditional RAN. In the considered scenario, user equipment UE_1 is moving from the coverage area of an RRH to another; both the source RRH and the destination RRH are connected to the same BBU, and remain like this as the user is moving. As a result, UE_1 will not encounter a handover. Instead, user equipment UE_2, also shown in the figure, will encounter a handover, as it moves between coverage areas corresponding to RRHs that are mapped on distinct BBUs. We note that, in a traditional network, where each RRH is mapped to only one BBU, both UE_1 and UE_2 would encounter a handover.

In Fig.2, we illustrate a representative scenario for RHO. Once again, we consider two user equipments UE_1 and UE_2, which are this time fixed. Suppose that, at consecutive network snapshots t and $t+1$, the network uses the BBU-RRH mappings portrayed in the figure. Focusing on UE_1, although the user remains covered by the same RRH, it will encounter an RHO as the system is reconfigured. The reason behind this is that the corresponding RRH switches its mapping to a new BBU. However, this will not be the case for UE_2, whose corresponding RRH remains mapped to the same BBU.

We note that RHOs do not exist in a traditional RAN and they are the consequence of the flexibility introduced by network function virtualization. However, these consequences, with a significant impact on the user QoE, as discussed above, have not been addressed in the CRAN literature, mostly focused on cost and throughput metrics. We argue that, with this extension of the handover concept to static users, there is a critical need for adequate strategies allowing to manage handovers in CRAN. In our study, we address this aspect and aim at reducing the number of handovers in the network, by tuning accordingly the network topology, i.e. the mapping between RRHs and BBUs.

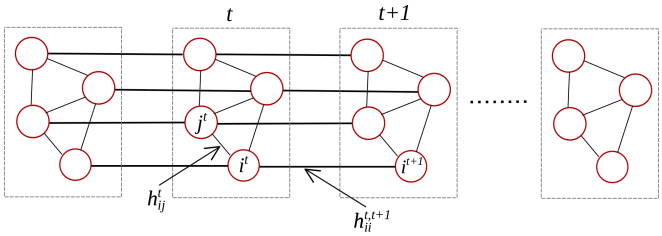


Fig. 3. Time-varying graph representation.

IV. CRAN System Model

We employ a time-varying graph model [18] in our study in order to represent the behavior of the network over time. In a time-varying graph $H(\mathcal{N},\mathcal{L})$, the set of nodes $\mathcal{N}$ can be written as $\mathcal{N} = \bigcup_{t\in\mathcal{T}} \mathcal{N}^t$, where $t \in \mathcal{T}$ is a static, discrete representation of the graph, called a *snapshot*. At the same time, the set of edges $\mathcal{L}$ represents all the interactions between nodes during the different snapshots, as well as the continuity of a node i through the different snapshots. This latter property is obtained by adding an edge between i^t, the instance of node i in snapshot t, and i^{t+1}, its instance during the next snapshot $t+1$. Fig.3 provides a visual representation of the time-varying graph we use to model a CRAN, as described below.

A. Time-varying Graph Representation

We consider the set of snapshots $\mathcal{T}$ to represent the access network traffic, with each snapshot t providing information about UEs demand over a time interval ΔT_s. ΔT_s represents the shortest possible scheduling time interval for a particular cellular network standard, e.g. a GSM frame or an LTE sub-frame. Each couple of consecutive snapshots in $\mathcal{T}$ are separated by a time interval ΔT_r, with $\Delta T_s \leq \Delta T_r$. ΔT_r refers to the smallest time interval for CRAN reconfiguration, i.e. the shortest possible duration for a BBU-RRH mapping.

We use i^t to denote an RRH i in the network during snapshot t, and $\mathcal{R}^t$ to represent the set of RRHs in snapshot t. We refer to the complete set of RRHs over $\mathcal{T}$ as $\mathcal{R}$. We then construct a time-varying graph structure $G(\mathcal{R},\mathcal{E})$, as illustrated in Fig.3.

In this graph, the set of edges $\mathcal{E} = \{\mathcal{E}_m \bigcup \mathcal{E}_r\}$ represents the potential MHO and RHO in the system. The set $\mathcal{E}_m$ is formed by subsets $\mathcal{E}_m^t$, such that an edge $e_{ij}^t \in \mathcal{E}_m^t$ exists if RRH i and RRH j are neighbors and users can move directly from one to another. In this work, we consider that two RRHs are neighbors if they share a common border in the Voronoi diagram of the access network. Each edge e_{ij}^t is assigned a weight h_{ij}^t referring to the total number of active users moving between RRHs i and j during the time interval ΔT_r, i.e. between snapshots $t-1$ and t.

The set $\mathcal{E}_r$ is also formed by subsets $\mathcal{E}_r^{t,t+1}$, such that an edge $e_{ii}^{t,t+1} \in \mathcal{E}_r^{t,t+1}$ links node i^t to node i^{t+1}, with an assigned weight $h_{ii}^{t,t+1} = u_i^t$. Weight u_i^t refers to the total number of users connected to RRH i^t and thus represents the total number of potential RHO, which would be triggered in case RRH i would change its BBU mapping in snapshot $t+1$. In the following,

we consider that the u_i^t users associated with RRH i^t require d_i^t resources during the time interval covered by snapshot t. At each snapshot, each RRH is mapped to a BBU k, which we consider to have a fixed capacity of resources c_k. We note that the resources of a BBU are limited even when advanced joint transmission techniques, such as coordinated multipoint (CoMP), are used in CRAN [19].

B. RRH Clustering

We aim at dividing the set of nodes $\mathcal{R}$ into a set of clusters $\mathbb{C}$, each cluster representing a BBU. We use $\mathbb{C}_k^t$ to refer to RRHs mapped to BBU k over snapshot t and $\mathbb{C}_k$ to refer to all RRHs, over time, mapped to BBU k. We note that a CRAN architecture imposes constraints on the fronthaul connecting the RRHs and the BBU pool [7]. Because of delay requirements, the distance between an RRH and the BBU pool can not be larger than 15 km [20]. In the scenarios we study, one data center is sufficient to cover the entire network. Therefore, we do not introduce any BBU-RRH distance constraint in our system, although this can be easily integrated in our time-varying graph model.

Our goal is to find a mapping between RRHs and BBUs that minimizes the number of handovers in the system. Using the time-varying graph representation, this is equivalent to grouping the graph nodes (i.e. the RRHs) into a set of clusters (i.e. BBUs), such as to minimize the weight of the edges (i.e. the number of handovers) connecting different clusters. We also need to take into account that each RRH has a resource demand, related to the number of UEs it serves, and that the BBUs have a limited resource capacity. Therefore, we formally define our problem as follows:

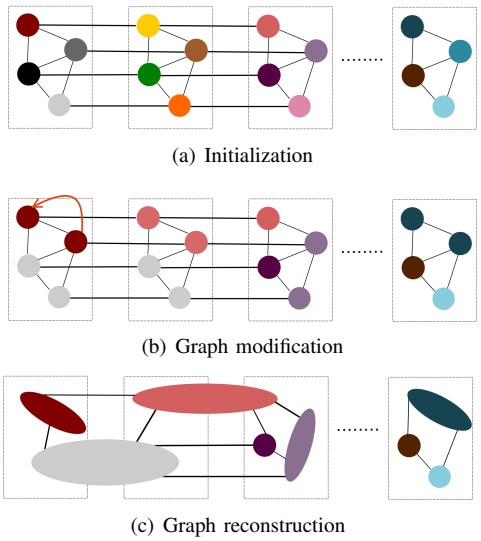
$$\min\Big(\sum_{i^t\in\mathbb{C}_k, j^t\in\mathbb{C}_l} h_{ij}^t + \sum_{i^t\in\mathbb{C}_k, j^t\in\mathbb{C}_l} h_{ii}^{t,t+1}\Big)$$

$$s.t. \sum_{i^t\in\mathbb{C}_k} d_i^t \le c_k;\ \forall k, \forall t \tag{1}$$

If we do not consider the supplementary capacity constraint in Eq.(1), our problem is a classical graph clustering, or community detection problem [21], where the goal is to find sets of nodes with similar properties.

The most popular techniques for community detection are based on the maximization of the modularity metric [21]. To define the modularity, let us assume the generic graph $H(\mathcal{N}, \mathcal{L})$, where nodes u and $v \in \mathcal{N}$ are linked by an edge l_{uv} with weight w_{uv}. Suppose that H is divided into a set of clusters, or partitions, $\mathbb{P}$. The modularity for the graph partitioning $\mathbb{P}$ compares the cohesion inside partitions to the case of a random distribution of edges over partitions. It takes a value ranging between -1 and 1. A high value of modularity indicates a high cohesion of links inside each partition, with respect to the links among them. The modularity can be evaluated as follows:

$$Q = \frac{1}{2W}\sum_{u\in\mathcal{N}, v\in\mathcal{N}}\Big(w_{uv} - \frac{w_u w_v}{2W}\Big)\delta(\mathbb{P}_m, \mathbb{P}_n), \tag{2}$$



(a) Initialization

(b) Graph modification

(c) Graph reconstruction

Fig. 4. Illustration of the steps of the oracle strategy over the time-varying graph. Each node takes a color based on the cluster to which it belongs.

where we consider that node u belongs to partition $\mathbb{P}_m$ and node v belongs to partition $\mathbb{P}_n$. The δ-function $\delta(x, y)$ is equal to 1 if $x = y$ and 0 otherwise. At the same time:

$$w_u = \sum_{v\in\mathcal{N}} w_{uv};\ \forall u \in \mathcal{N} \tag{3}$$

is the sum of weights of edges attached to a node u, and

$$W = \frac{1}{2}\sum_{u\in\mathcal{N}, v\in\mathcal{N}} w_{uv} \tag{4}$$

is the total sum of weights of edges, in the whole graph H.

Optimizing the modularity is known to be NP-hard [22]. Among the solutions proposed in the literature, the Louvain method [14] is the most popular heuristic and we rely on it in order to design our strategies, after some modifications required to take into account the capacity constraint in Eq.(1).

V. Mobility-aware RRH-BBU Mapping

We use the CRAN model described in Sec.IV to find an RRH-BBU mapping minimizing the number of handovers in the system. We first take an oracle approach in Sec.V-A, considering that we have a perfect knowledge of UE demand and mobility over the next $|\mathcal{T}|$ snapshots. An online algorithm, without a priori knowledge of the network load, is then discussed in Sec.V-B.

A. Oracle-type Algorithm

Assuming the knowledge of traffic demand, we build a time-varying graph on which we run a modified version of the Louvain algorithm. We outline in Alg.1 the steps that we follow to cluster the graph. In the algorithm, we mark in red the parts that we add to the original Louvain method. Fig. 4 highlights the three major steps of the algorithm: initialization, graph modification and graph reconstruction.

The algorithm operates over a temporary graph structure $G^{temp} = (\mathcal{N}^{temp}, \mathcal{L}^{temp})$, where $\mathcal{N}^{temp}$ forms a set of nodes, and $\mathcal{L}^{temp}$ is a set of weighted edges linking them. G^{temp} is updated at each iteration. The algorithm builds G^{temp} by considering

that, initially, $\mathcal{N}^{temp} = \mathcal{R}$ and $\mathcal{L}^{temp} = \mathcal{E}$. It also assigns the weight of each edge in $\mathcal{E}$, to its counterpart in $\mathcal{L}^{temp}$. Moreover, it associates to each RRH in $\mathcal{R}$ a cluster $\mathbb{C}_k$. This first step is summarized in Line 1 of Alg.1 and in Fig.4(a).

In the second step, the algorithm attempts to modify the structure G^{temp}, as follows. Each node r has a set of neighbors $\mathcal{N}_r^{temp}$ (i.e. the neighboring RRHs). A node r will join the community of a neighbor $s \in \mathcal{N}_r^{temp}$, which gives the maximum increase in modularity, as long as this does not violate the capacity constraint of the cluster to which belongs s. This step is repeated as long as positive modularity variations are obtained, as described in Fig.4(b) and lines 5-26 in Alg.1. There, function ExtractNeighbors(r) returns the set $\mathcal{N}_r^{temp}$ and CalculateΔQ(r, s) computes the modification in modularity if r joins the community of s. Function VerifyCapacity(r, s) checks if letting node r join the community of s results in surpassing the capacity limit of the corresponding BBU. If that is the case, the function returns False, and otherwise it returns True. Finally, function JoinCommunity(r, s) lets r join the community of s.

With respect to the original Louvain method [14], we added the capacity constraint in lines 13-19. This allows us to add a node to a community (i.e. to map an RRH to a BBU) only when the capacity of the cluster is sufficient (i.e. when the BBU has enough resources to handle the RRH load).

```
1  G^temp, C = Attribute(R, L);
2  modif = True;
3  rebuilt = True;
4  while rebuilt == True do
5      while modif == True do
6          modif = False;
7          for r ∈ N do
8              ΔQ_max = 0;
9              s_max = 0;
10             N_r^temp = ExtractNeighbors(r);
11             for s ∈ N_r^temp do
12                 ΔQ_rs = CalculateΔQ(r,s);
13                 if ΔQ_rs > ΔQ_max then
14                     verif = VerifyCapacity(r,s);
15                     if verif == True then
16                         ΔQ_max = ΔQ_rs ;
17                         s_max = s ;
18                     end
19                 end
20             end
21             if ΔQ_max > 0 then
22                 JoinCommunity(r,s_max);
23                 modif = True;
24             end
25         end
26     end
27     rebuilt,G^temp, C = RebuildGraph(G^temp, C);
28     modif = True;
29 end
```

Algorithm 1: Oracle BBU-RRH mapping algorithm.

Once there are no more possible modifications implying positive variations in the modularity, the algorithm rebuilds the graph, by taking communities as nodes and linking them with a new set of edges, weighted by the total weight of the inter-cluster links. This step is completed by function RebuildGraph($G^{temp}, \mathbb{C}$) in Line 27 and depicted in Fig.4(c).

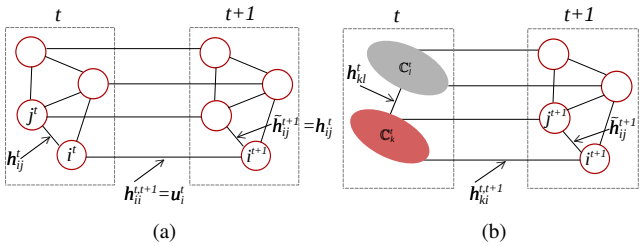


Fig. 5. Illustration of snapshot representation (a) and graph construction process (b) employed in the online BBU-RRH mapping algorithm.

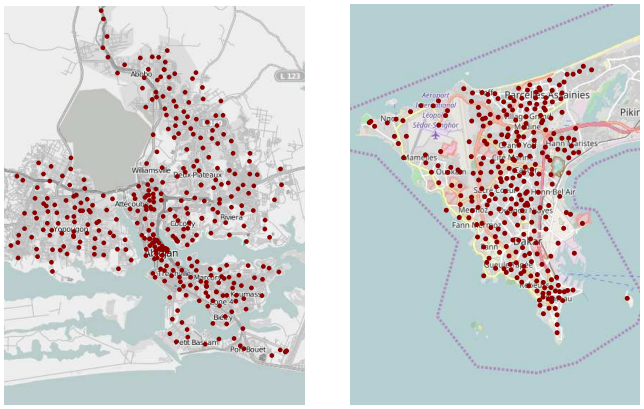
B. Online Algorithm

Our online BBU-RRH mapping strategy adopts a similar time-varying graph representation to the one used in the oracle algorithm, except that it runs in a real-time manner, without a priori knowledge of the network load.

On a periodic basis, the algorithm operates over a graph $G^{temp} = (\mathcal{N}^{temp}, \mathcal{L}^{temp})$, derived based on the current traffic snapshot t and a prediction of the future traffic snapshot $t+1$. The objective is to determine the BBU-RRH mapping over snapshot $t+1$ or, more precisely, the mapping to be used during the ΔT_r time interval separating snapshots t and $t+1$.

We present in Fig.5(a) an example of a couple of consecutive snapshots t and $t+1$ used to build G^{temp}. For this, we need to know h_{ij}^t, the number of active users moving between RRHs i and j during snapshot t, the set of clusters $\mathbb{C}^t$ representing the RRH-BBU mapping at t, and the number of users u_i^t covered by RRH i^t. All this information is easily available at the BBU pool. However, we also need to estimate $\widetilde{h}_{ij}^{t+1}$, the number of active UE movements between RRHs i and j over the next ΔT_r period, and $\widetilde{d}_i^{t+1}$, the required resources on each RRH during the next snapshot. Depending on the reconfiguration duration ΔT_r, obtaining this estimation can be more or less difficult. While rather accurate traffic estimation techniques exist in the literature [23], in the following we consider $\widetilde{h}_{ij}^{t+1} = h_{ij}^t$ and $\widetilde{d}_i^{t+1} = d_i^t$. We thus assume that the mobility of users and their traffic demand across the network does not encounter important variations over ΔT_r. As shown in Sec. VIII, this basic prediction technique already provides very good results.

Based on this information, G^{temp} is constructed by considering that, initially, $\mathcal{N}^{temp} = \mathcal{R}^{t+1} \bigcup \mathbb{C}^t$. This means that $\mathcal{N}^{temp}$ includes RRHs in $\mathcal{R}^{t+1}$, as well as current clusters $\mathbb{C}^t$. The set of edges $\mathcal{L}^{temp}$ linking nodes in $\mathcal{N}^{temp}$ is illustrated in Fig.5(b) and derived as follows. We place an edge $e_{ki}^{t,t+1}$, with weight $h_{k,i}^{t,t+1} = h_{ii}^{t,t+1}$, between cluster $\mathbb{C}_k^t$ and node i^{t+1} if $i^t \in \mathbb{C}_k^t$. We also add a link e_{kl}^t between cluster $\mathbb{C}_k^t$ and $\mathbb{C}_l^t$ for every $(i^t, j^t) \in \mathbb{C}_k^t \times \mathbb{C}_l^t$ where RRHs i and j are neighbors. We assign to edge e_{kl}^t a weight $h_{k,l}^t = \sum_{i^t \in \mathbb{C}_k^t, j^t \in \mathbb{C}_l^t} h_{i,j}^t$. In addition to that, neighboring RRHs i^{t+1} and j^{t+1} are linked through an edge e_{ij}^{t+1} whose weight is the number of UEs predicted to move between the two RRHs between snapshots t and $t+1$.



(a) (b)

Fig. 6. Geographical distribution of BSs in Abidjan (a) and (b) Dakar.

The operation of our online approach is similar to the oracle approach in Alg.1, but some adaptations are required for the online scenario. If the snapshot to consider is the first one, i.e. $t = 0$, the algorithm starts by initializing the set of current clusters with an empty set: $\mathbb{C}^t = \mathbb{C}^0 = \emptyset$. Then, for any snapshot t, the algorithm builds the graph G^{temp} using a modified function AttributeOnline($\mathcal{R}^t, \mathcal{E}^t, \mathbb{C}^t$). The graph modification and graph reconstruction steps operate the same way as in the offline strategy case, except that we do not allow to move nodes in $\mathcal{R}^t$ outside their current clusters $\mathbb{C}^t$. Practically, this forbids us to go back in the past and modify the configuration we decided at the previous snapshot, which is coherent with an online algorithm.

VI. Datasets and Benchmarks

We evaluate our strategies using two real-world traffic datasets, provided within the context of the Data for Development (D4D) challenges [24], [25]. The datasets represent call traffic activities of Orange customers, covering five months in 2012 over the city of Abidjan in Ivory Coast [24] and the whole 2013 year over the city of Dakar in Senegal [25]. For hourly time intervals, each dataset provides the number and duration of voice calls exchanged between each couple of BSs over the Orange GSM network. Fig.6 portrays the position of BSs over the maps of the cities: 364 BS are installed in Abidjan and 290 BS are installed in Dakar.

By assuming that RRHs will occupy the position of Orange BSs, user demand and mobility in a CRAN RRH correspond to that of the BS it will replace. As explained, we also assume that all BBUs are grouped in one data center for each city. We conduct our evaluation over typical working weekdays, Apr. 3rd, 2012 for Abidjan and Jan. 15th, 2013 for Dakar.

Some information required for our evaluation is missing in the original datasets. Hence, we need to derive some traffic and mobility information. We consider that the number of calls arriving during one frame time follows a Poisson distribution with parameter λ, computed from the average number of calls per frame given in the dataset. Since the dataset provides hourly information, λ also changes with an hourly basis.

We also consider that the duration of calls follows a Log-normal distribution [26]. The location μ and scale σ parameters of the distribution are derived based on the mean m and standard deviation s of the non-logarithmized hourly calls duration. We are only able to obtain m from the dataset, as we only have the information concerning the aggregate duration of calls and their number over each hour. Concerning the standard deviation s, we suppose its value is equal to 1 s.

Finally, the D4D dataset does not include information concerning the mobility of users. We derive it by assuming that x% of the total calls experience a handover. We then randomly pick these calls and choose the UE destination cell according to a uniform distribution over the set of neighboring cells. To test the impact of user mobility on our RRH-BBU mapping solutions, we test two use-cases, with the assumptions of 5% and 50% of total calls encountering a handover, representative of two extreme cases: low and high mobility scenarios.

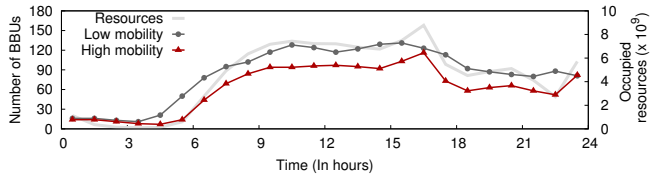
We compare our solutions to two BBU-RRH mapping strategies. The first one aims at optimizing the mapping between BBUs and RRHs from a frequency utilization perspective. More precisely, its objective is to efficiently utilize frequency resources, in order to minimize the interference in the system. Instead, the second strategy aims at optimizing BBU-RRH mapping from a capacity utilization perspective. In particular, it aims at minimizing the number of BBUs that are being used in the network. Both these strategies operate over separate individual traffic snapshots. We briefly describe them in the following:

Frequency-oriented strategy. We employ the frequency-oriented BBU-RRH mapping algorithm proposed by Wang *et al.* [8]. The algorithm determines the mapping based on a dynamic frequency reuse scheme. It applies a graph coloring method over separate snapshot graphs. In each graph, nodes represent RRHs and edges link neighboring RRHs. As such, the method does not allow to use the same color, i.e. the same frequency, for adjacent nodes.

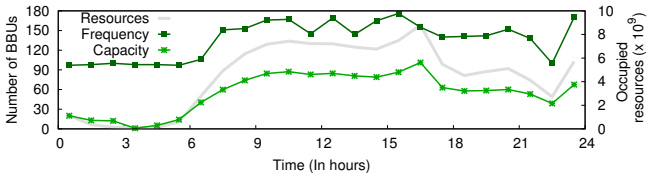
Capacity-oriented strategy. This algorithm determines the mapping between BBUs and RRHs with the objective of minimizing the number of BBUs. It employs a greedy approach over a traffic snapshot. The algorithm first selects the RRH with the highest user demand and places it on a BBU. It then goes through the list of its neighbors, by decreasing order of demand and places them over the same BBU, as long as its capacity is not violated. The algorithm then covers all nodes in the network by considering multi-hop neighborhoods, one after the other. Once a BBU reaches its capacity limit, another BBU is instantiated.

VII. Oracle strategy evaluation

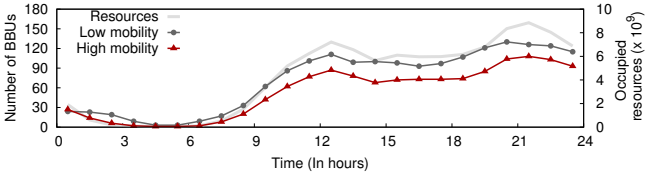
We apply our oracle BBU-RRH mapping strategy, detailed in Sec. V-A, to the processed datasets introduced in Sec. VI. We initially assume that a BBU is capable of handling no more than the demand of a high capacity GSM BS, with 18 carrier frequencies. We test our oracle-type BBU-RRH mapping algorithm, as well as the frequency- and capacity-oriented strategies, detailed above, over the Abidjan and Dakar



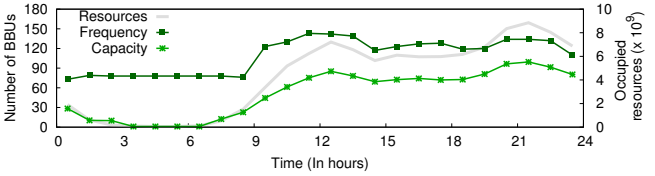
(a) Abidjan



(b) Abidjan



(c) Dakar



(d) Dakar

Fig. 7. (a,c): Average number of BBUs required over the day, for low and high mobility scenarios, using the oracle strategy with ΔT_r = 10 min, together with the total number of hourly occupied GSM resources. (b,d): Average number of BBUs required over the day, for the high mobility scenario, using the frequency-oriented and capacity-oriented state-of-the-art algorithms with ΔT_r = 10 min, together with the total number of hourly occupied GSM resources.

Mobility \ Strategy	Oracle	Frequency	Capacity	Online
Low	0.77	4.79	4.4	0.77
High	0.76	1.35	0.95	0.71

TABLE I

RATIO OF HANDOVERS R, FOR OUR ORACLE STRATEGY AS WELL AS STATE-OF-THE-ART FREQUENCY-ORIENTED AND CAPACITY-ORIENTED STRATEGIES IN ABIDJAN.

Mobility \ Strategy	Oracle	Frequency	Capacity	Online
Low	0.85	3.06	2.74	0.78
High	0.79	1.19	0.85	0.79

TABLE II

RATIO OF HANDOVERS R, FOR OUR ORACLE STRATEGY AS WELL AS STATE-OF-THE-ART FREQUENCY-ORIENTED AND CAPACITY-ORIENTED STRATEGIES IN DAKAR.

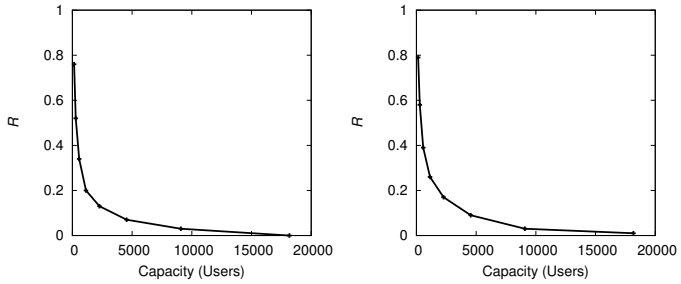
datasets. For each dataset, our tests are led over the two scenarios of low and high mobility (5% and 50% of the calls suffer a handover, respectively). In this part, we consider that two consecutive snapshots are separated by a time interval ΔT_r = 10 min, meaning that, once the decision taken, an RRH must remain mapped to a BBU for at least 10 minutes.

We compute the ratio $R = \frac{HO_{CRAN}}{HO_{RAN}}$, representing the ratio between the number of handovers in CRAN, HO_{CRAN}, and the number of handovers existing in a traditional RAN, HO_{RAN}. The corresponding results are presented in Tab.I and Tab.II, for each dataset, in the two mobility scenarios. Focusing first on our oracle strategy in the case of Abidjan, we note that R values equal to 0.77 and 0.76 are obtained in the low and high mobility scenarios, respectively. This indicates that the mobile network triggers 23% and 24% less handovers with respect to a traditional RAN, in the two mobility cases. Instead, both frequency- and capacity-oriented strategies lead to much higher ratios. More critical is the fact that these strategies result in R values higher than one, meaning that they increase the number of handovers with respect to the existing architecture (up to 4x more handovers, depending on the scenario). The same observations hold for the case of Dakar, with slight variations in the obtained values.

We complement these results by checking the overall evolution of the BBUs in the network. We plot in Fig. 7(a) and Fig. 7(c) the average number of BBUs required throughout the day, based on our oracle strategy, for the Abidjan and Dakar datasets respectively. In each case, we present the results for low and high mobility scenarios. The figures also show the hourly occupied resources in each case, an indication of the mobile network load. They correspond to the actual number of GSM resources, obtained from the datasets. We remark that, as expected, more BBUs are required during the day than during the night, when a few BBUs can handle the entire load in the city. However, even at peak hours, the number of BBUs is much lower than in a classical RAN (we recall that 364 BBUs are needed for Abidjan and 290 BBUs are needed for Dakar). We also note that, in the presence of low mobility, more BBUs are needed than in the case of high mobility. The reason for this is that, in the low mobility scenario, smaller weights are attributed to edges producing MHOs. The oracle algorithm will favor in this case clusters of individual or small groups of RRHs over the time dimension, leading to more clusters in the graph and a higher number of BBUs.

We plot the number of BBUs required for both frequency- and capacity-oriented strategies in Fig.7(b) and Fig.7(d). We notice that our oracle strategy results in slightly more BBUs than the capacity-oriented strategy and much less BBUs compared with the frequency-oriented strategy. This shows that the cost of accounting for handovers is not very important: only 10 supplementary BBUs are required at peak hour by our solution when compared with the capacity-oriented strategy, while reducing the handovers between 20% and 400%.

Finally, we are interested in the performance of the oracle strategy when coordinated transmissions are allowed between RRHs. Thanks to its centralized architecture, CRAN is expected to enable CoMP communication techniques, such as Coordinated Scheduling and Beamforming (CS/CB) [19]. By adjusting power levels and beamforming coefficients among different antennas, lower inter-carrier interference is obtained, leading to higher signal-to-interference-plus-noise ratios. By that, as more antennas, i.e. RRHs, are assigned to the same BBU, higher capacities can be achieved. Thus, to evaluate our strategy in CS/CB-enabled systems, we compute the ratio R



(a) Abidjan (b) Dakar

Fig. 8. Ratio of handovers R for different BBU capacity limits, when using the oracle strategy in high mobility scenario.

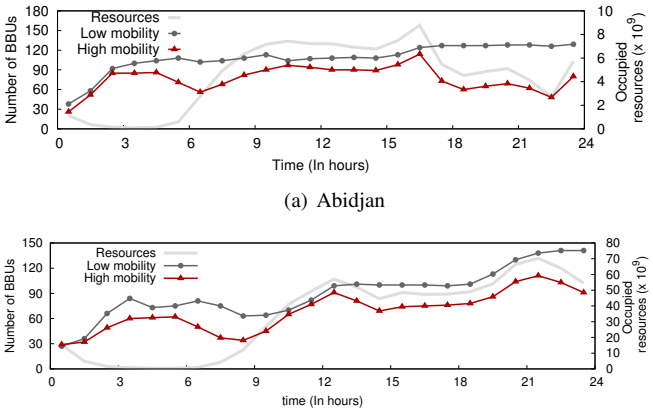
for different BBU capacity limits. The results are presented in Fig. 8, for the high mobility scenario. We notice that, as higher capacity limits can be reached in the system, the ratio R of handovers drops. Indeed, by increasing the BBU capacity, less BBUs are needed to handle traffic demands, resulting in a lower portion of handovers in the overall system.

VIII. Online algorithm evaluation

In this section, we focus on the evaluation of our online BBU-RRH mapping strategy introduced in Sec. V-B. As for the oracle algorithm, we consider the two scenarios of low and high mobility and we assume that two snapshots are separated by a time interval $\Delta T_r = 10$ minutes. We compare the number of handovers obtained in CRAN using our online strategy to the number of handovers in a traditional RAN in Tab.I and Tab.II. In the case of Abidjan, we observe that we have 23% and 29% less handovers in the low and high mobility scenarios respectively. In Dakar, the strategy results in 22% and 21% less handovers for the low and high mobility scenarios respectively.

This indicates that the online method can lead, in some cases, to a lower number of handovers with respect to the oracle strategy. This surprising behavior can be explained by the fact that the online strategy makes a prediction of future traffic in snapshot $t+1$, as described in Sec. V-B. As a result, when the UE demand is underestimated, the BBU capacity can be inferior to the number of required resources, producing a call drop. These call drops were not allowed in the oracle strategy and the apparent gain of the online strategy is the consequence of this extra degree of liberty. We also note that the online strategy leads to a higher number of BBUs in the network when compared to the oracle algorithm, as shown in Fig. 9. This is mainly due to fact that the online strategy is not allowed to change mapping decisions taken for past snapshots.

A parameter with an important impact on the quality of the estimation is the CRAN reconfiguration time ΔT_r: as this value increases, the time window that needs to be estimated increases as well, and the quality of the estimation decreases. This leads to blocking more calls as a result of higher errors in future traffic estimation. A second parameter with significant consequences on the online algorithm is the access control capacity threshold L. Practically, for BBU k with capacity c_k, the access control policy allows an RRH to map to a BBU only if the total demand of the BBU does not exceed $c_k L$.



(a) Abidjan

(b) Dakar

Fig. 9. Average number of BBUs required over the day for low and high mobility scenarios, using the online strategy with $\Delta T_r = 10$ min, together with the total number of hourly occupied GSM resources.

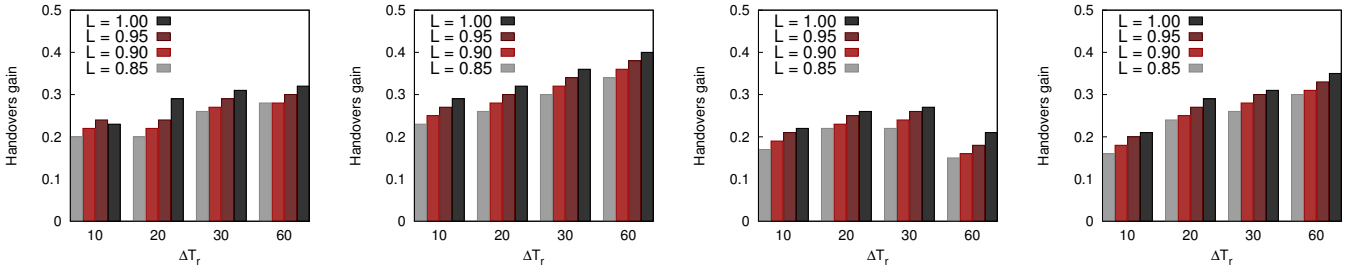
In Fig. 10, we show the handover ratio R for the online algorithm with different values for parameters ΔT_r and L. We plot the obtained gain for ΔT_r equal to 10, 20, 30 and 60 minutes, and different capacity thresholds L of 0.85, 0.9, 0.95, and 1, in the low and high mobility scenarios, for the two cities. The figure shows that, as higher values of ΔT_r and L are considered, we generally get higher gains in terms of handovers. The exception is the low mobility scenario in Dakar, which indicates the existence of a threshold value $\Delta T_r = 30$ minutes where the handover gain starts decreasing. In fact, higher values of ΔT_r translate into less reconfigurations in the system, meaning less RHOs and more MHOs.

We also evaluate, in Fig. 11, the call blocking probability obtained by our online strategy with the various values of ΔT_r and L. The call blocking probability is calculated at high traffic hours only, i.e. between 7:00 and 21:00, and represents the percentage of calls lost because of exceeding the capacity limit of a BBU. We can notice that, for higher values of ΔT_r and L, the call blocking probability increases. In the extreme case of $\Delta T_r = 60$ minutes and $L = 1$, the percentage of blocked calls is as high as 1.5%, which is not acceptable for mobile operators. Therefore, the price to pay for saving 40% of handovers might be too important. However, even a small tolerance for call blocking probability, in the order of 10^{-4}, can result in handover gains of more than 30%, for example when $\Delta T_r = 60$ minutes and $L = 0.85$.

In summary, these results indicate that using the online strategy, with adequate parameters, leads to important savings in terms of handovers in the network, offers a more efficient management of BBUs, and grants a lower system reconfiguration frequency. Yet, this comes at the price of a slight increase in the probability of blocking communications. By introducing a capacity threshold that prevents BBUs from becoming overloaded, one can balance the various performance metrics.

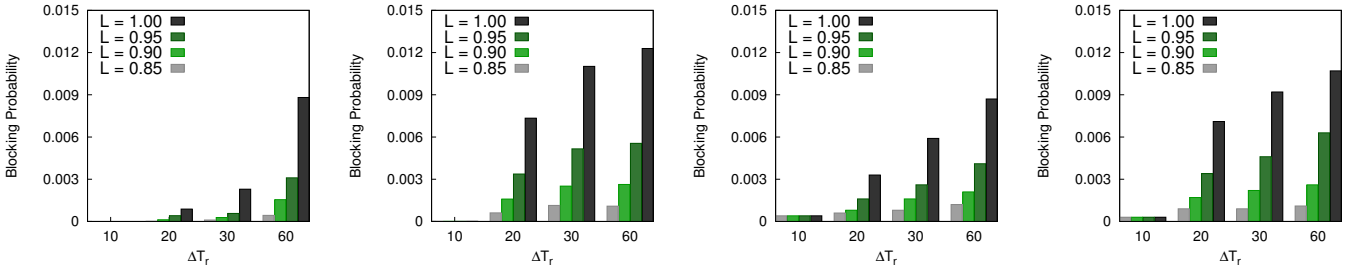
IX. Conclusion

In this paper, we study the problem of BBU-RRH mapping in CRAN, focusing on user mobility management mechanisms. After unveiling the emergence of a new type of



(a) Abidjan, low mobility (b) Abidjan, high mobility (c) Dakar, low mobility (d) Dakar, high mobility

Fig. 10. Handovers gain for different ΔT_r values and capacity thresholds, when using the online strategy.



(a) Abidjan, low mobility (b) Abidjan, high mobility (c) Dakar, low mobility (d) Dakar, high mobility

Fig. 11. Blocking probability for different values of ΔT_r and L, when using the online strategy.

handovers, as part of the virtualized CRAN, we propose dynamic BBU-RRH mapping methods that allow to efficiently manage the overall number of handovers in the network. The proposed algorithms rely on a time varying-graph model of user behavior over the access network. They allow to dynamically reshape the mapping between BBUs and RRHs by adapting it to the mobility and traffic consumption patterns of users. We show that important savings, of more than 30% when compared with the number of handovers in a classical RAN, can be obtained based on our strategies in a real-world environment. In our future work, we plan to extend our study by covering various cellular network technologies, using more accurate predictions and datasets reflecting user behavior.

References

[1] Cisco, "Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2015–2020", White Paper, Feb. 2016.

[2] I. C.-Lin *et al.*, "Recent Progress on C-RAN Centralization and Cloud-ification", *IEEE Access*, vol.2, 1030–1039, Sep. 2014.

[3] Z. Zhu *et al.*, "Virtual Base Station Pool: Towards a Wireless Network Cloud for Radio Access Networks", *Proc. ACM CF*, Ischia, Italy, May 2011.

[4] A. Checko *et al.*, "Cloud RAN for Mobile Networks - A Technology Overview", *IEEE Com. Surveys & Tutorials*, 17(1): 405–426, Mar. 2015.

[5] C. Liu *et al.*, "The Case for Re-Configurable Backhaul in Cloud-RAN based Small Cell Networks", *Proc. IEEE Infocom*, Turin, Italy, Apr. 2013.

[6] S. Bhaumik *et al.*, "CloudIQ: A Framework for Processing Base Stations in a Data Center", *Proc. ACM MobiCom*, Istanbul, Turkey, Aug. 2012.

[7] J. Tang *et al.*, "System Cost Minimization in Cloud RAN with Limited Fronthaul Capacity", *IEEE Transactions on Wireless Communications*, 16(5):3371–3384, May 2017.

[8] K. Wang *et al.*, "Graph-based dynamic frequency reuse in Cloud-RAN", *Proc. IEEE WCNC*, Istanbul, Turkey, Apr. 2014.

[9] S. Namba *et al.*, "Colony-RAN Architecture for Future Cellular Network." *Proc. FutureNetw*, Berlin, Germany, Jul. 2012.

[10] A. Checko *et al.*, "Evaluation of Energy and Cost Savings in Mobile Cloud RAN." *Proc. OpNetwork*, Washington, DC, USA, Nov. 2013.

[11] M.Y. Lyazidi *et al.*, "Dynamic Resource Allocation for Cloud-RAN in LTE with Real-Time BBU/RRH Assignment." *Proc. IEEE ICC*, Kuala Lumpur, Malaysia, May 2016.

[12] L. Liu *et al.*, "Analysis of Handover Performance Improvement in Cloud-RAN Architecture." *Proc. ICST ChinaCom*, Kunming, China, Aug. 2012.

[13] K.Sundaresan *et al.*, "FluidNet: A Flexible Cloud-based Radio Access Network for Small Cells", *IEEE/ACM Trans. on Networking*, 24(2): 915–928, Apr. 2016.

[14] V. D. Blondel *et al.*, "Fast Unfolding of Communities in Large Networks", *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10): P10008, Oct. 2008.

[15] M. Z. Shafiq *et al.*, "Understanding the Impact of Network Dynamics on Mobile Video User Engagement", ACM SIGMETRICS, 42(1), 367-379, Jun. 2014.

[16] A. Balachandran *et al.*, "Modeling Web Quality-of-Experience on Cellular Networks", *Proc. ACM Mobicom*, Maui, Hawaii, Sep. 2014.

[17] C. Gomez *et al.*, "Impact of Handover between UMTS and GPRS on TCP/IP: An Empirical Approach", *Proc. IEEE VTC*, Montreal, Canada, Sep. 2006.

[18] V. Kostakos, "Temporal Graphs", *Physica A: Statistical Mechanics and its Applications*, 388(6): 1007–1023, Mar. 2009.

[19] V.N. Ha *et al.*, "Computation Capacity Constrained Joint Transmission Design for C-RANs", *Proc. IEEE WCNC*, Doha, Qatar, Apr. 2016.

[20] K. Murphy, "Centralized RAN and Fronthaul", Ericsson White Paper, May 2015.

[21] A. Lancichinetti *et al.*, "Community Detection Algorithms: A Comparative Analysis", *Physical Review E*, 80(5), 056117, Nov. 2009.

[22] U. Brandes *et al.*, "On Modularity-NP-Completeness and Beyond", *IEEE Trans. on Knowledge and Data Engineering*, 20(2): 172–188, Feb. 2008.

[23] D. Naboulsi *et al.*, "Classifying Call Profiles in Large-scale Mobile Traffic Datasets", *Proc. IEEE Infocom*, Toronto, Canada, Apr. 2014.

[24] V. Blondel *et al.*, "Data for Development: The D4D Challenge on Mobile Phone Data", *arXiv preprint arXiv*, 1210.0137 (2012).

[25] Y. Montjoye *et al.*, "D4D-Senegal: The Second Mobile Phone Data for Development Challenge", *arXiv preprint arXiv*, 1407.4885 (2014).

[26] J. Guo *et al.*, "Estimate the Call Duration Distribution Parameters in GSM System Based on K-L Divergence Method", *Proc. IEEE WiCom*, Shanghai, China, Sep. 2007.