



**HAL**  
open science

# Hyperbolic K-means for traffic-aware clustering in cloud and virtualized RANs

Hanane Djeddal, Liticia Touzari, Anastasios Giovanidis, Chi-Dung Phung,  
Stefano Secci

► **To cite this version:**

Hanane Djeddal, Liticia Touzari, Anastasios Giovanidis, Chi-Dung Phung, Stefano Secci. Hyperbolic K-means for traffic-aware clustering in cloud and virtualized RANs. *Computer Communications*, 2021, 176, pp.258-271. 10.1016/j.comcom.2021.06.021 . hal-03109662v3

**HAL Id: hal-03109662**

**<https://hal.science/hal-03109662v3>**

Submitted on 13 Jul 2021

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - NonCommercial - ShareAlike 4.0 International License

# Hyperbolic K-means for traffic-aware clustering in cloud and virtualized RANs

Hanane Djeddal<sup>a,\*</sup>, Leticia Touzari<sup>a,\*</sup>, Anastasios Giovanidis<sup>1,a</sup>, Chi-Dung Phung<sup>b</sup> and Stefano Secci<sup>b</sup>

<sup>a</sup>Sorbonne University, <sup>1</sup>CNRS-LIP6, 4 Place Jussieu, Paris CEDEX 05, France, 75252

<sup>b</sup>Cnam, Cedric Lab, 292 Rue Saint-Martin, Paris, France, 75003

## ARTICLE INFO

### Keywords:

C-RAN  
vRAN  
O-RAN  
SD-RAN  
K-means  
Clustering  
Hyperbolic geometry  
Poincaré half-plane

## ABSTRACT

As the internet and connected objects gain more and more in popularity, serving the ever-increasing data traffic becomes a challenge for the mobile operators. The traditional cellular radio access network (RAN), where each base station is co-located with its own processing unit and is responsible for a specific geographic area, has evolved first with the so-called Cloud RAN (C-RAN), and is currently undergoing further architectural evolution under the virtualized RAN (vRAN), Software-Defined RAN (SD-RAN), and Open RAN (O-RAN) architectures. In all these versions, the data processing units can be dynamically centralized into a pool and shared between several base stations, enlarging the geographical view for scheduling and resource allocation algorithms. For instance, resource utilisation is improved by avoiding resource idling during off-peak hours. C-RAN and vRAN gains depend strongly on the clustering scheme of radio units (RRHs and RUs). In this paper, we propose a novel radio clustering algorithm that takes into account both the traffic demand and the position of stations, by using the hyperbolic distance in 3-dimensions. We introduce a modified K-means clustering algorithm, called Hyperbolic K-means, and show that this generates geographically compact RU clusters with traffic charge equally shared among them. Application of our algorithm on real-world mobile data traffic, collected from the cities of Nantes and Lille in France, shows an increase in resource utilisation by 25%, and a reduction in deployment cost by 15%, compared to the standard RAN. Furthermore, the performance of our Hyperbolic K-means algorithm is compared extensively against alternative C-RAN clustering proposals from the literature and is shown to outperform them, in resource utilisation as well as in cost reduction.

## 1. Introduction

With the rapid growth in consumer electronics, boosted by the Internet-of-Things, more and more wireless products are entering the market adding to the total number of devices accessing mobile networks. According to Cisco 2017 [10], the share of smart devices and connections from the total of connected objects will increase from 53% in 2017 to 73% by 2022. Furthermore, new services emerge related to Internet-of-Vehicles, remote control, and video monitoring [9], which are expected to generate intensive demand for network edge-computing resources. Satisfying such increase in traffic is a big challenge for mobile operators; a naive installation of more base stations or addition of more data processing units to existing sites, has high deployment and energy expenses, without guaranteed service improvement or revenue increase for the mobile operators.

As an alternative solution for low cost, and low energy-consumption, around 2010 the Cloud Radio Access Network (C-RAN) architecture was conceived by IBM [20] and developed by the China Mobile Research Institute [8], later adopted also by other operators. This concept is currently enlarged to accommodate various other possible configurations with the so-called virtualized RAN (vRAN) and open RAN (O-RAN), which offers flexibility in functional splitting for 5G architectures. [13], [22], [11].

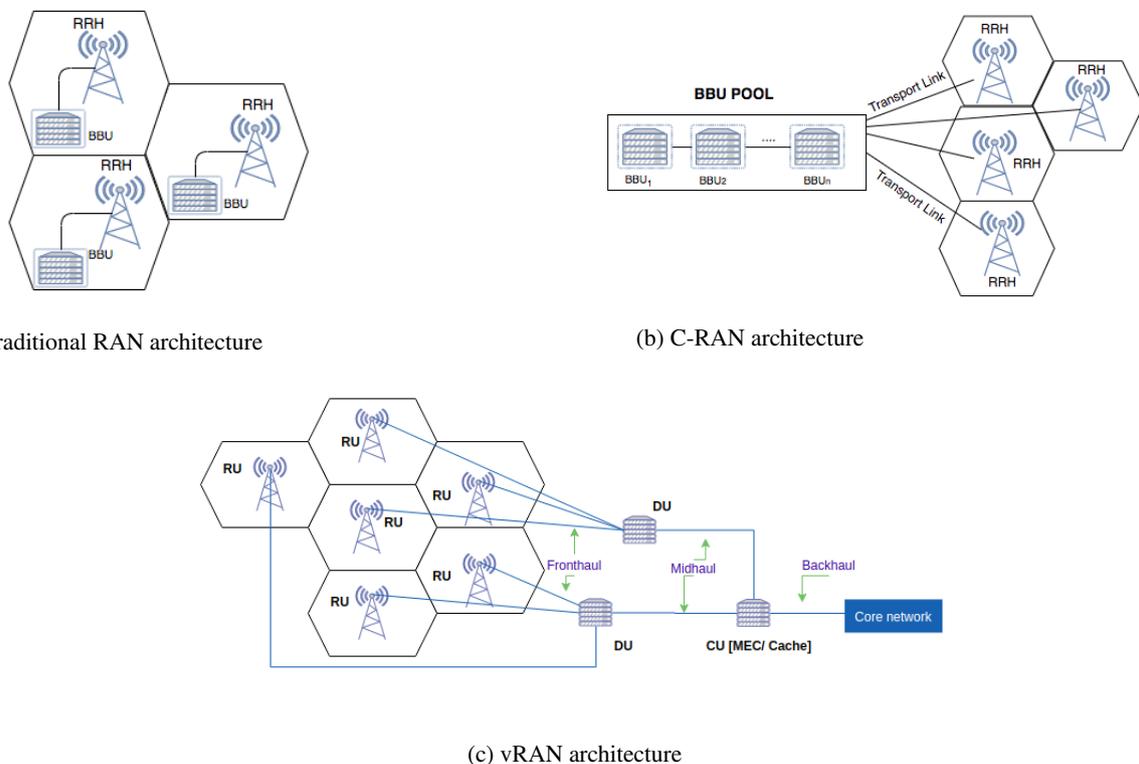
While in the traditional RAN, illustrated in Fig. 1a, each site has its own Remote Radio Head (RRH) and Base-Band Unit (BBU), in the C-RAN architecture base stations (BS)

host only RRHs whereas the BBUs are centralized in a pool allowing resources to be dynamically allocated to the RRH cluster [5], [16] see Fig. 1b. By removing BBUs from cell sites, operating expenses as well as energy consumption can be reduced, while providing the same coverage and a better quality of service [28], [3].

To allow for more flexibility in the resource allocation of the BBUs, the virtualised-RAN (vRAN) was introduced [26], [29]. Inspired by Network Function Virtualisation (NFV) it allows to run the baseband unit controls as software on generic hardware platforms. This way resource allocation and load balancing can be adapted on demand by software updates, without having to replace hardware throughout the entire infrastructure. C-RAN and vRAN architectures are not alternative ones. Even if C-RAN was deployed earlier, a vRAN infrastructure is broader in scope in the sense that it encompasses the softwarization of a number of RAN functions, and not only the BBU function. In this scope, C-RAN is viewed as one application running on top of a vRAN/SD-RAN infrastructure.

Current vRAN systems are split into three parts [26], [27], [29]: the Radio Units (RU) are geographically scattered to achieve coverage; these are connected to Distributed Units (DU), where some layer-1 and layer-2 functions can be performed; the latter are in turn connected to Centralized Units (CU) where base-band operations can be run, along with other RAN orchestration and Mobile-Edge Computing subsystems [4], [13], see Fig. 1c. The C-RAN system can be seen as a special case or just an application of a vRAN, where vBBUs are hosted at CUs, and RRH correspond to the

\*Equal contributions from both authors.



**Figure 1:** Illustration of the traditional RAN architecture and comparison with C-RAN and vRAN.

RU and DU combined. Generally, RU and DU are expected to be collocated or geographically very close to each other in practice, and transport links provide connections between the DUs and the CU hosting the BBU assigned to the RU-DU subsystem. In 5G and beyond 5G systems, it is envisioned that CU, BBU and MEC servers can be co-located at different edge network sites, typically the so-called Central Office (CO), legacy network points-of-presence being rearchitected nowadays as mini-datacenters [4]. Usually, the distance between an RRH and its BBU-pool (resp. DU and CU) can reach several kilometers and fiber lines are used for reliable high bandwidth data transfer.

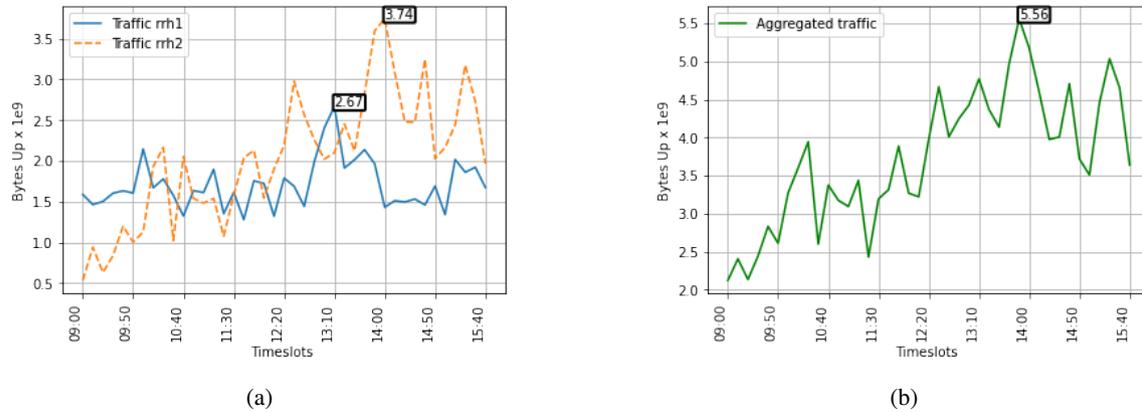
In the following, we will mostly focus on the C-RAN application but the algorithm we propose can be adapted to vRAN/MEC as well, where base stations are clustered at the CU level. We will use the abbreviation RU to refer to both RRH (for C-RAN) and RUs (for vRAN).

Aggregating resources to a pool level allows them to be managed more efficiently and optimize their utilisation. Each BBU pool is in charge of several RUs, hence the installed capacity of the pool should be enough to serve the maximal ensemble traffic generated/received by these RUs, and the total capacity can be further shared among them according to their real-time demand. Performance is optimal when the BBU-pool utilisation is as flat as possible through the day, with a peak-to-average ratio close to one. To achieve this, clustering based on the traffic pattern of the stations (RUs) can be exploited. In fact, grouping RUs with complementary traffic patterns to the same BBU-pool can reduce the total capacity necessary to support the aggregate traffic of the

cluster and maximize its utilisation over different periods of the day [7].

A meaningful clustering scheme should then (i) reduce the overall processing (BBU) capacity necessary and maximize its utilisation. (ii) The load should be fairly balanced among clusters. The scheme should also (iii) keep a limit on the optical fiber distance between the BBU-pool and its assigned RUs in order to respect delay and propagation constraints. Furthermore, (iv) the clusters should be geographically as compact as possible, with neighbouring RUs assigned to the same cluster, in order to avoid frequent handovers between clusters when users are mobile.

In this paper, we propose a BBU-RUs clustering solution that offers improved capacity utilization and reduced deployment cost, while respecting geographic requirements and load fairness among clusters. Our starting point is the standard K-means algorithm, but we introduce a non-trivial modification of the distance metric. The proposed modification involves the distance of the Poincaré half-plane model in 3-dimensions (“hyperbolic  $\mathbb{H}^3$ ”), which allows us to appropriately combine the different natures of traffic and positional features. This approach was motivated by recent advances in complex networks [23] and data science [19], which show that embedding complex data into hyperbolic spaces is very advantageous for problems of clustering and community detection [14]. Our method, being center-based, groups RUs around a fictional center (so-called “centroid”) which determines the average charge of the cluster. The goal is to end up with a fair clustering scheme, which satisfies the above design requirements (i)-(iv). The solution is done per



**Figure 2:** (a) Data traffic patterns of two RUs in Lille from 2019-03-19 (09:00) to 2019-03-19 (15:40) showing their traffic peak hour and volume. (b) Their aggregated traffic.

time-slot (e.g. hour), and is extended by proposing a robust clustering over multiple consecutive time-slots.

The paper is structured as follows: We first state the general BBU-RU clustering problem for the C-RAN architecture, and discuss standard C-RAN clustering performance metrics, in Section 2. We then present existing clustering methods from the literature in Section 3. In Section 4 we introduce our clustering solution based on the hyperbolic  $\mathbb{H}^3$  distance and explain how it leads to the desired properties for the formed clusters; we also include its robust extension over multiple time-slots. In Section 5 we evaluate our algorithm using real data from Orange mobile France for two cities (Nantes, Lille) and compare its performance with existing approaches. Finally, Section 6 concludes our work.

## 2. Problem statement

In traditional RANs, each base station should be equipped with enough processing capacity in order to meet traffic demand at all times for the coverage area it is assigned to. However, since the traffic demand varies over different periods of the day, the processing capacity of base stations is not always used to its fullest [7]. Moreover, different areas have different traffic patterns, for example business districts can have maximal charge during morning and afternoon hours in week days, while residential areas witness their rush hours during evenings and weekends. This means a certain BS can be overcharged during its peak hour while another BS in a different area can have minimal traffic at the same time, as shown in the example of Fig. 2a. In such scenarios, which occur very often in practice, the capacity of the low-charged BS is idling, whereas it could support traffic service for the overcharged one [7].

C-RAN and vRAN architectures group BSs together and allow them to cooperate by sharing common resources. In the example of Fig. 2a if the two BSs are mapped to the same BBU pool, their traffic is aggregated as shown in Fig. 2b, hence the capacity needed to serve both BSs simultaneously should at most cover the peak of their aggregated traffic (equal to 5.56 GB), which is smaller than the sum of their indi-

vidual peak traffic (equal to  $3.75+2.67= 6.42$  GB). As a consequence, less processing units are necessary to be installed for the service of the group of these two BSs. What is more, during off-peak hours the pool resources are more intensively utilised, and do not idle.

The objective in this paper is to propose clustering (BS grouping) schemes that minimize the BBU resources necessary to meet the traffic demand, under some distance limitations. These limitations have to do with the propagation time between RU and BBU, so that clustering very distant RUs to the same BBU-pool is not recommended due to unacceptable delays. Hence, the solution of a fully centralised cloud architecture with all RUs in a city served by the same BBU-pool need not be optimal or even feasible, and we need to search for a semi-centralised solution that determines how many clusters should be optimally introduced per area (city), and which are their RU members.

To further elaborate on the appropriate number of clusters in a city, note here that the C-RAN (resp. vRAN) implementation is very flexible and can vary depending on the needs: there is the option to consider each macro-cell as a cluster center and include several add-on cells, or aggregate pairs of macro-cells together for smooth handover and load balancing [18]; the authors in [25] suggest that a typical macro-cell can support up to 15 RUs; an extreme solution is to aggregate all RUs to a unique BBU pool, most suitable for small towns [13]; other authors [29] try to find the optimal number of DUs or CUs from the solution of an optimisation problem. In any case, the total Round-Trip-Time (RTT) from RU to BBU limits the size and diameter of clusters. The typical RTT budget is  $3msec$  due to Hybrid-ARQ delay requirements [2], [29]. This delay budget includes the processing time as well as fronthaul transmission delay between RU and BBU. In an optical fiber network, it corresponds to approximately  $20km$  fronthaul length. But for time-critical ultra Reliable Low Latency Communication (uRLLC) applications, which are expected to be supported in 5G and future cellular networks, the end-to-end delay requirement is limited to  $1msec$ , [29], [24]. Such tight delays can only be guaranteed by shortening the distance between RUs and the BBU pool

**Table 1**  
Table of Notations and Variables

Variable	Description	Variable	Description
$P$	Partition	$m$	Centroid
$C$	Cluster	$(\bar{x}, \bar{y}, \bar{w})$	Coordinates of a centroid
$K$	Number of clusters with index $k$	$(\tilde{x}, \tilde{y}, \tilde{w})$	normalised coordinates
$T$	Number of time-slots with index $t$	$\sigma$	Threshold for DCCA ( $km$ )
$N$	Number of BSs with index $n$	$\tau_{in}$	Number of iterations in <b>Algo. 2</b>
$E2$	Euclidean 2D distance ( $m$ or $km$ )	$\tau_{out}$	Number of repetitions of <b>Algo. 2</b>
$E3$	Euclidean 3D distance	$\gamma$	Scaling parameter in $H3$
$H3$	Hyperbolic distance	$U$	Average-to-Peak Traffic Ratio
$\varphi$	A set of centroids	$Diam$	Cluster Diameter ( $m$ )
$(x, y)$	Geographical coordinates of a BS	$Var$	Within-cluster variance
$z$	Traffic dimension of a BS (Bytes)	$Util$	Utilization for a partition $P$
$w$	Available resources of a BS (Bytes)	$Cost$	Cost for a partition $P$
$Z$	Aggregated traffic per cluster	$W$	Aggregated resources per cluster

considerably. Hence, the appropriate number of clusters is a function of delay limitations and the city size; it further depends on the mix of supported services, as well as the fronthaul and midhaul link capacity. It is also worth mentioning that, in a forthcoming SD-RAN framework, the availability of virtualization facilities at central-offices independently of the C-RAN needs is expected to favor decentralization of BBUs. Our work here presents a general method to cluster RUs together, for any given number of clusters  $K \geq 1$ , but we do not explicitly provide a rule on how to choose this number. We rather show how metrics such as utilisation, cost, aggregated traffic per cluster and cluster diameter vary as a function of selected  $K$ .

To evaluate the quality of our proposed clustering solution, and compare it with existing ones, we will mainly use two performance metrics, already introduced in [7]. We use these because they elegantly quantify C-RAN clustering performance, and also in order to have coherence and a common vocabulary for the performance comparisons between various schemes. We will show that the clustering scheme proposed in this paper outperforms existing schemes, as evaluated from the metrics that follow. The main notation used throughout the paper is summarised in Table 1.

## 2.1. Notation and Evaluation Metrics

Let  $z(n, t)$  be the traffic of station  $n$  at time-slot  $t$  for  $n = 1, \dots, N$  and  $t = 1, \dots, T$ , where  $N$  is the total number of stations in an area (city) and  $T$  is the number of time-slots. Here, we will use as time-slot duration 1 hour, but other choice is possible depending on the available data. Requested traffic per station is normalised to 1, so that the request per slot refers to the percentage of installed processing capacity utilised to satisfy traffic demand. We assume that all individual stations have the same amount of pre-installed processing capacity, so that  $z(n)$ 's are comparable. After normalisation, we introduce a transformation of the traffic demand, called the *remaining (or available) resources per station  $n$*  and time-slot  $t$ . This is equal to

$$w(n, t) = 1 - z(n, t). \quad (1)$$

It quantifies the percentage of resources remaining idle per station per time-slot, and will be used in the proposed clustering algorithm.

Consider a partition  $P = \{C_1, \dots, C_K\}$  of the set of  $N$  stations into  $K \in \mathbb{Z}_+$  clusters. Denote by  $Z(C, t)$  the aggregated traffic at time  $t$  of the stations grouped in cluster  $C$ , i.e.,

$$Z(C, t) = \sum_{n \in C} z(n, t). \quad (2)$$

In the degenerate case of a cluster with a single station it holds,  $Z(\{n\}, t) = z(n, t)$ . Similarly, the aggregate remaining resources are defined as  $W(C, t) = \text{card}(C) - Z(C, t)$ , where  $\text{card}(C)$  is the number of RUs forming the cluster.

**Average-to-Peak Traffic Ratio (AtPTR)** for cluster  $C$ . This is a measure of capacity utilisation per cluster. It is defined as the ratio of the average aggregate traffic request (i.e. average capacity used) at the BSs of the cluster for several time-slots, over the peak aggregate traffic,

$$U(C) = \frac{\frac{1}{T} \sum_{t=1}^T Z(C, t)}{\max_t Z(C, t)}. \quad (3)$$

Obviously,  $U(C) \in (0, 1]$ . When  $U(C)$  is close to 1 the cluster makes good use of the pool resources over time, and resource idling is low. The AtPTR measure can be defined for just one station, i.e.  $C = \{n\}$ , in which case  $U(n) = \frac{1}{T} \sum_{t=1}^T z(n, t) / \max_t z(n, t)$ .

**Metric I: Utilization** for partition  $P$ . It is a measure of the improvement of the average AtPTR (i.e. capacity utilization per cluster) due to the partition  $P$ , compared to the average AtPTR over all single base stations in a traditional simple RAN,

$$Util(P) = \frac{\frac{1}{K} \sum_{k=1}^K U(C_k)}{\frac{1}{N} \sum_{n=1}^N U(n)}. \quad (4)$$

It holds  $Util(P) \geq 1$ . The equality holds when  $K = N$ , i.e. one cluster per BS. When traffic from several BSs in a cluster is aggregated, the pool resources are better utilized and

the metric increases. The  $Util(P)$  is maximum for  $K = 1$ .

**Metric II: Cost** for partition  $P$ . It is a measure of the decrease in total installed BBU-pool resources in the clustered C-RAN, compared to the total BBU resources installed in individual base stations (simple RAN).

$$Cost(P) = \frac{\sum_{k=1}^K \max_t Z(C_k, t)}{\sum_{i=1}^N \max_t z(n, t)}, \quad (5)$$

where the maximum is taken over all time-slots  $t = 1, \dots, T$ . It holds  $Cost(P) \leq 1$ . The equality holds again here when  $K = N$ . The  $Cost(P)$  is minimum when  $K = 1$ .

The two performance metrics for a clustering  $P$  quantify two different things. Metric I measures how much the utilisation of installed resources is improved by clustering  $P$ , whereas metric II measures the economies in installed resources due to clustering  $P$ , compared to the simple RAN standard scenario. Both are optimal for  $K = 1$ . However, due to distance limitations, the solution to create a single global pool per city is not feasible.

### 3. Clustering methods in the literature

#### 3.1. Linear Programming

Many research works in the literature try to formulate and solve the clustering problem as a Mixed Integer Linear Program (MILP), see [13], [26], [11], [29]. Although this approach is definitely valid, these problems are NP-hard, so high complexity issues arise when the number of stations is city-wide large, and especially when considering multi-period scheduling. Since it is computationally expensive to solve them to optimality, certain authors propose heuristic solutions without giving implementation details, or optimality guarantees. Another issue is that most of these algorithms make arbitrary assumptions about the input parameter values and costs that will determine the clustering solution. The later can strongly be affected by small imprecisions or change over time in the measurement (sensitivity of the solution). Finally, an important weak point is that the objective is maximised with binary (0-1) association and routing variables, that decide on which BBU-pool (or CU) to associate each RU with, without taking into account the relative geographic positions of the stations, and the potential interference these may introduce (or avoid) because of the specific partition as solution.

More specifically, the work in [26] formulates a variation of the 1D bin-packing problem with numerous additional realistic restrictions, in order to find the optimal number of DU and the appropriate associations. This is an NP-hard problem and the authors resort to heuristic methods for its solution without any optimality guarantees. Their problem is formed based on several given input parameters related to capacity, bandwidth and traffic load, so the solution will vary considerably depending on how these parameters change over time. In [13] the authors consider a single-CU architecture and form an MILP with MEC/network function

placement and routing variables. This is also an NP-hard problem as shown by reduction from the multi-dimensional multiple-choice Knapsack problem. To solve it efficiently the authors propose a decomposition using Benders' decomposition. The solution is computationally very demanding (NP-hard), especially for large instances with reasonable required precision. The work in [11] is an MILP network slice design and function placement problem, which is also NP-hard. The authors solve an instance using a commercial integer solver. Similarly, the authors in [29] minimise the active number of central offices (COs) housing DUs and CUs. They form an MILP with bandwidth, latency and processing requirements that are included in the constraints. The NP-hard problem in the evaluation is solved by a commercial integer solver for some given set of values for the parameters.

#### 3.2. K-means clustering

An appropriate method that involves the 2D-geometry and relative positions is by using data analysis. K-means is one of the most standard clustering algorithms, which can be used to partition the set of BSs of the city into  $K$  disjoint clusters. Each BS is associated to a unique cluster, resulting in a partition  $P$  that respects the following properties [17]:

1.  $\bigcup_{k=1}^K C_k = \{1, \dots, N\}$ , i.e. the union of clusters includes all present BSs, where  $C_1, C_2, \dots, C_K$  are the  $K$  clusters.
2.  $C_k \cap C_{k'} = \emptyset$  for all  $k \neq k'$ , i.e. clusters are disjoint sets.

K-means clustering is designed to minimize the within-cluster variance by regrouping the most similar BSs in the same cluster. In this case similarity is relevant to the 2D-distance

$$\min_{C_1, C_2, \dots, C_K} \sum_{k=1}^K Var(C_k), \quad (6)$$

with  $Var(C_k)$  the within-cluster variance of the cluster  $k$ ,

$$Var(C_k) = \frac{1}{card(C_k)} \sum_{i \in C_k} d^2(i, m_k). \quad (7)$$

The  $m_k$  is the centroid of the cluster  $k$ , and in euclidean space it is the point having as coordinates the average values of all RUs in the cluster. In 2D it is defined as  $m_k = (\bar{x}_k, \bar{y}_k)$ .

The  $d^2(i, j)$  is the square of the distance between station  $i$  and the cluster centroid  $m_k$ . If we assume that the RUs are embedded on the 2D euclidean space, the distance  $d_{E2}(x_i, y_i, x_j, y_j) = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$  can be used, which makes a lot of sense in the C-RAN and vRAN problem, where base stations are positioned on the 2D plane; it can be favorable to group stations together that are closer to each other, thus avoiding inter-clustering hand-overs and bringing all RUs closer to the BBU-pool, positioned at the cluster centroid. Note, however, that this approach makes use only of the two geographical coordinates  $(x_n, y_n)$  per BS  $n = 1, \dots, N$ , and does not include the traffic demand feature.

A naive extension to the 3D, is to include all dimensions  $(x, y, z)$ , (or  $(x, y, w)$  for the resources), where the variables  $z$  and  $w$  refer to a specific time-slot  $t$ . As we will show later, the euclidean 3D distance ( $E3$ ) is not the best choice, because the clusters formed do not exhibit the desired properties. The pseudo-algorithm for 2D euclidean K-means is described as follows (it extends to 3D or higher dimensions in a straightforward way):

---

**Algorithm 1:** Euclidean K-means Clustering in 2D

---

1. Initialise: Fix a number  $K$  of clusters and randomly assign each RU to one of the  $K$  clusters.
  2. Iterate until the cluster assignments stop changing:
    - (a) Compute the centroid of each cluster. The  $k$ th cluster centroid is the  $m_k = (\bar{x}_k, \bar{y}_k)$  average position of the RUs in the  $k$ th cluster.
    - (b) Assign each BS to the cluster whose centroid is closest according to the 2D-Euclidean distance.
- 

### 3.3. Distance-Constrained Clustering Algorithm

The authors in [6] and [7] propose the Distance-Constrained Clustering Algorithm (DCCA) for RU clustering. This algorithm takes into consideration not only the position, but also the traffic-demand and its temporal evolution through the day, while respecting some distance constraint. The method introduces an entropy-based weight to evaluate the complementarity of traffic between pairs of RUs over a determined time-interval. RUs are placed as nodes of a weighted-graph, whose edges can exist or not and allow clusters to form. Each link has a binary activation variable controlled by the algorithm; when this becomes 1, an edge appears between the two RUs, which can now share their resources and belong to the same cluster. The authors further introduce a distance constraint, so that only RUs within some distance from each other are allowed to collaborate.

Although this approach sets the problem in a correct framework, it is sub-optimal because the complementarity of a large group of RUs is calculated as the sum of complementarities between pairs of these RUs. This is of course not correct: in a scenario of three stations, both  $RU_2$  and  $RU_3$  may have large traffic demand, which renders them complementary in traffic with  $RU_1$  which has low charge; but placing them all three in the same cluster should normally be avoided, because  $RU_2$  and  $RU_3$  are not mutually complementary. The pairwise comparison is thus suboptimal, and other approaches which consider the joint-complementarity for the whole set of RUs in a cluster should be investigated.

## 4. Hyperbolic K-means

In this section our novel suggestion for RAN clustering is introduced.

### 4.1. Traffic-aware RU clusters

The main challenge is how to incorporate the traffic demand inside the clustering algorithm, additionally to the positional features. To achieve this, we first present three Qualitative criteria that should be satisfied by any traffic-aware clustering:

- Q.1 When two RUs have complementary traffic (or remaining resources), i.e. one has high traffic load (resp. low resources) and the other low traffic load (resp. high resources), they should be grouped together, in order to cooperate, within certain geographical limitation of distance.
- Q.2 When two base stations both have low traffic volume (resp. high resources), it is irrelevant whether they cooperate or not, because of low benefit.
- Q.3 When two base stations both have high traffic volume (resp. low resources), their cooperation leads to no benefit and they should not be clustered together. Ideally, highly loaded stations should be distributed evenly among several clusters.

We saw in Section 3.2 that the K-means approach for 2D has the advantage to summarise well the information from all members of the cluster at its centroid. Furthermore, the method generates clusters that are compact, thus avoiding to include RUs that spread over large distances from the centroid of each cluster. It has two drawbacks, however: the vanilla K-means in 2D does not include the traffic dimension; it also generates one partition per time-slot, hence does not easily generalise to longer time-intervals.

A naive way to incorporate the resource dimension  $w$  (or the traffic dimension  $z$ ) in the K-means, as additional feature to the position  $(x, y)$  of the RU, is to consider the 3D euclidean metric space and its corresponding euclidean distance. From now on we will use  $w$  as the third feature. The 3D euclidean distance between  $RU_i = (x_i, y_i, w_i)$  and  $RU_j = (x_j, y_j, w_j)$  is

$$d_{E3}(x_i, y_i, w_i, x_j, y_j, w_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (w_i - w_j)^2}. \quad (8)$$

The above expression can be re-written as

$$d_{E3} = \sqrt{d_{E2}^2(x_i, y_i, x_j, y_j) + (w_i - w_j)^2}, \quad (9)$$

which relates the 3D with the 2D euclidean distance, and the difference of the remaining resources. This specific choice, although natural, *does not achieve the desired effect, since it treats the traffic and position in a homogeneous way*, although the units and range of geography and traffic are completely different. The  $(x, y)$  is positioned in the real 2D-space  $((-\infty, +\infty) \times (-\infty, +\infty))$  and the features  $x, y$  are measured in meters, or kilometers, whereas the resource dimension is in BBU units, so that  $w \in [0, 1]$  (the BBU capacity is normalised to 1). Another issue with the choice of 3D euclidean distance, is that two RUs have minimum distance when  $w_i = w_j$ , i.e. when both stations have the same remaining resources, irrespective of whether this quantity is

high or low. Moreover, given a fixed difference  $(w_i - w_j)^2$ , the square of the 3D-distance is proportional to the square of the 2D-distance; as a result, RUs in a small geographical distance from each other are favored to collaborate, whereas large 2D-distance prohibits any collaboration.

Given the above observations, the 3D euclidean distance is not the best candidate to apply K-means, when aiming to achieve a clustering with balanced load (i.e. inter-cluster complementarity), and we need to look for other candidates, that treat the traffic dimension in a way different than the RU position.

## 4.2. Hyperbolic Distance

An interesting idea is to embed the 3D features of each RU  $(x_n, y_n, w_n)$  into a hyperbolic space instead, and use the distance induced by the Poincaré metric. The idea of embedding complex data into hyperbolic space is recent, but is gaining momentum. It has already been successfully applied for the analysis of complex networks in [19], as well for developing new methods to learn symbolic data which exhibit hierarchy and similarity from Facebook's research team in [23]. These highly cited works show that for problems of clustering and community detection, the hyperbolic space is more appropriate than the Euclidean. Very recently, the authors in [15] have formulated the K-means clustering algorithm in the hyperbolic setting, where they use the Poincaré ball model [21, Chapter 8.1]. Further contributions include the introduction of hyperbolic space algorithms for community detection [14].

In the majority of the aforementioned methods data is embedded into the Poincaré ball model, which in 3D is the manifold  $\mathbb{B}^3 = \{(x, y, w) \in \mathbb{R}^3 : \|(x, y, w)\| < 1\}$ . In our case, the positional features  $(x, y) \in \mathbb{R}^2$  are real numbers, but the feature of remaining resources is in fact real positive  $w \in \mathbb{R}_+$ . So, we choose to embed the data on the Poincaré half-plane model instead [21, Chapter 8.2], which in 3D is the manifold  $\hat{\mathbb{B}}^3 = \{(x, y) \in \mathbb{R}^2, w \in \mathbb{R}_+ : \|(x, y, w)\| < 1\}$ , with induced distance between  $RU_i = (x_i, y_i, w_i)$  and  $RU_j = (x_j, y_j, w_j)$

$$d_{H3}(x_i, y_i, w_i, x_j, y_j, w_j) = \operatorname{arccosh}\left(1 + \frac{d_{E3}^2(x_i, y_i, w_i, x_j, y_j, w_j)}{2w_i w_j}\right) \quad (10)$$

In the above  $\operatorname{arccosh}(s) = \ln(s + \sqrt{s^2 - 1})$ , an increasing function of the argument  $s$ , with domain  $s \in [1, \infty)$ . Note that the expression in (10) satisfies the formal properties of a distance: Take any three points  $A, B, E \in \hat{\mathbb{B}}^3$ . Then the following properties can be shown to hold,

- (a)  $d_{H3}(A, B) > 0$  for  $A \neq B$  (Positivity),
- (b)  $d_{H3}(A, B) = 0 \Leftrightarrow A = B$  (Identity of indiscernibles),
- (c)  $d_{H3}(A, B) = d_{H3}(B, A)$  (Symmetry), and
- (d)  $d_{H3}(A, B) \leq d_{H3}(A, E) + d_{H3}(E, B)$  (Subadditivity).

It is clear from (10) that the hyperbolic distance (we refer to it from now on in short as  $H3$ ) is monotone increasing in

$d_{E3}$ . However, the third resource dimension  $w$  plays a special role as it divides the  $d_{E3}$ . As a consequence, when the absolute available resources  $w_i, w_j$  are small, their product is small, and the distance between  $RU_i$  and  $RU_j$  is large [12], [1]. This is important because it generates the tendency to place RUs with low resources (high load) in separate clusters (following requirement Q.3 from Section 4.1). Furthermore, RUs that are distant in the 2D-euclidean sense are perceived closer through the hyperbolic lens. To understand the properties of  $H3$ , how these compare to  $E3$  and  $E2$ , and how they are more appropriate for our traffic-aware clustering problem, we refer the reader to Appendix A.

With a bit of calculus, the new distance takes another interesting form

$$d_{H3}(x_i, y_i, w_i, x_j, y_j, w_j; \gamma) = \operatorname{arccosh}\left(\gamma \frac{d_{E2}^2(x_i, y_i, x_j, y_j)}{2w_i w_j} + \frac{1}{2} \left(\frac{w_i}{w_j} + \frac{w_j}{w_i}\right)\right). \quad (11)$$

Now, it is more clear how  $H3$  incorporates in its expression both the Euclidean 2D distance (involving the positional pair  $(x, y)$ ), as well as the impact of resource imbalance from  $w_i$  and  $w_j$ . The argument of  $\operatorname{arccosh}$  has two summands. The first is described by the 2D-euclidean distance with  $(x, y)$  and the product of the resource feature  $w$ , whereas the second depends only on the resource feature  $w$ . The second summand quantifies the resource imbalance between  $w_i$  and  $w_j$ . It is 1 when  $w_i = w_j$  and larger than 1 when  $\|w_i - w_j\| > 0$ . In fact, it is symmetric regarding an imbalance towards  $w_i$  or  $w_j$ . The hyperbolic distance depends on these two summands, and trades-off between geographical distance and resource imbalance. The issue is however, that 2D-distance and resources *do not have the same units!* If the  $d_{E2}$  is measured in meters, the geographical distance between two RUs can be of the order of thousands, and can dominate the summation. If we use  $d_{E2}$  measured in kilometers instead, the resource balance summand will dominate, thus giving more emphasis on whether two RUs have the same amount of resources or not. There are a lot of other options, including yards, feet, centimeters, etc. Although the units depend on the convention we make, each choice will lead to a different value of the distance and consequently a different clustering result.

This is why we introduce a new parameter  $\gamma > 0$  inserted above. This parameter simply scales the units of distance in 2D, e.g. in meters, centimeters, kilometers, or other. For  $\gamma = 1$ , the expression in (11) is equal to (10). If  $\gamma = 1$  calibrates  $d_{E2}^2$  in  $m^2$ , then we can shift to  $km^2$  by setting  $\gamma = 10^{-6}$ . More generally,  $\gamma$  can be left as a tunable parameter, because it can impact the value of distance and the final clustering solution. We investigate its influence in the evaluation Section 5. Since there are so many options, we can eventually choose an appropriate  $\gamma$  value that maximises our metrics.

### 4.3. Hyperbolic K-means algorithm for RU clustering

We are now ready to incorporate the  $H3$  distance for K-means clustering in C-RAN. As observed in Fig. 15, the 3D-hyperbolic distance (and also the euclidean) between two points  $(x_i, y_i, w_i)$  and  $(x_j, y_j, w_j)$  is minimal when the resource ratio  $\rho = 1$ , meaning that the resources are balanced  $w_i = w_j$ . This property can be exploited for clustering, by introducing the centroid of each cluster  $C_k$ . Let us define the centroid  $m_k = (\bar{x}_k, \bar{y}_k, \bar{w}_k)$  coordinates, again as the averages

$$(\bar{x}_k, \bar{y}_k, \bar{w}_k) = \frac{1}{\text{card}(C_k)} \sum_{j \in C_k} (x_j, y_j, w_j). \quad (12)$$

In every update loop, each  $RU_n$ ,  $n = 1, \dots, N$  will be associated to the cluster with minimum hyperbolic distance  $d_{H3}(n, m_k)$  between the specific  $RU_n$  and the centroid. Ignoring for a moment the effect of the positional pair dimensions  $(x, y)$  (the 2D-euclidean distance), such rule will try to associate stations to clusters in a way that changes as little as possible the average available resources  $\bar{w}_k$ . A further consequence is that the resulting clusters will have balanced average traffic in a fair way (see Example below).

In this case, the within-cluster variance of a given cluster  $C_k$  having as centroid  $m_k(\bar{x}_k, \bar{y}_k, \bar{w}_k)$  is defined as follows

$$\text{Var}_{H3}(C_k) = \frac{1}{\text{card}(C_k)} \sum_{i \in C_k} d_{H3}^2(i, m_k), \quad (13)$$

and the global distortion is the sum of variances over all clusters

$$V_{H3}(P) = \sum_{k=1}^K \text{Var}_{H3}(C_k). \quad (14)$$

---

#### Algorithm 2: Hyperbolic K-means Clustering in 3D

---

1. Initialise: Fix a number  $K$  of clusters. Set the iteration counter  $\tau_{in} = 0$  and chose randomly the initial cluster centroids  $(\bar{x}_k(0), \bar{y}_k(0), \bar{w}_k(0))$ . Also, fix the tolerated distortion error  $\epsilon > 0$ .
  2. Assign each RU to the cluster whose centroid is the closest according to the hyperbolic distance  $H3$ . The resulting initial partition is  $P(0) = \{C_1(0), \dots, C_K(0)\}$ .
  3. Calculate the initial global distortion  $V_{H3}(P(0))$ .
  4. Repeat:
    - (a) Update  $\tau_{in} \leftarrow \tau_{in} + 1$ .
    - (b) Calculate the new centroid of each cluster by (12).
    - (c) Assign each observation to the cluster whose centroid is closest according to the distance  $H3$ , to derive the new partition  $P(\tau_{in})$ .
    - (d) Calculate the new global distortion  $V_{H3}(P(\tau_{in}))$ .
- Until the difference between the global distortions of two consecutive iterations is less than  $\epsilon$  :
- $$|V_{H3}(P(\tau_{in})) - V_{H3}(P(\tau_{in} - 1))| < \epsilon$$

---

**Implementation:** In Step 1 (initialisation), it is practically better to place the  $K$  initial centroids at the positions of  $K$  RUs randomly among the present  $N$  RUs. This choice aims to guarantee that no cluster will start empty, and in practice leads to  $K$  non-empty clusters after convergence. Since the centroids are initialised randomly, and since the K-means algorithm is known not to converge to a global optimum, we repeat the above Algorithm several times (use a counter  $\tau_{out}$ ) and then we pick the solution with maximum performance, among the  $\tau_{out}$  available ones.

**Example:** To see how the hyperbolic K-means works, with the resulting fairness among clusters, we make a thought experiment. Suppose  $RU_1 = (-\delta, 0, \alpha w)$ ,  $RU_2 = (0, 0, w)$  and  $RU_3 = (+\delta, 0, w/\alpha)$  are co-linear on the x-axis, as their coordinates indicate. Let  $\alpha > 0$  in general. The  $RU_2$  has the same 2D-euclidean distance  $\delta$  from both  $RU_1$  and  $RU_3$ . Suppose we want to find  $K = 2$  clusters. Will  $RU_2$  be clustered with  $RU_1$  or with  $RU_3$ ?

The cluster centroids are initialised as  $m_1 = RU_1$  and  $m_3 = RU_3$ . The association of  $RU_2$  is with  $RU_1$  if  $d_{H3}(RU_2, RU_1) < d_{H3}(RU_2, RU_3)$ . Using the expression from (11) with  $\gamma = 1$ , we see that this inequality is valid when  $\alpha > 1$ . So, when  $\alpha > 1$ , the clustering  $C_1 = \{RU_1, RU_2\}$  and  $C_2 = \{RU_3\}$  will update the centroid coordinates as  $m'_1 = (-\delta/2, 0, w(1 + \alpha)/2)$  and  $m'_2 = (+\delta, 0, w/\alpha)$ . As a result, the difference of average resources between cluster  $C_1$  and  $C_2$  equals to  $\Delta w = ||w(1 + \alpha)/2 - w/\alpha||$  is smaller (more fair) compared to the situation when the  $RU_2$  would be clustered with  $RU_3$ , in which case  $\tilde{\Delta} w = ||w\alpha - w(1 + 1/\alpha)/2|| > \Delta w$ .

The association will not change after the update of centroids, because for  $\alpha > 1$  it can be verified that it still holds  $d_{H3}(RU_2, m'_1) < d_{H3}(RU_2, m'_2)$ .

### 4.4. Robustness

Up to this point, we have considered a single traffic value per station per time-slot  $t$ . As explained at the beginning, the traffic varies over time, and suppose we have several measurements per station through the day. We thus need to obtain a robust clustering over multiple consecutive time-slots.

As a first naive idea, we can consider averaging the traffic per RU over all the  $T > 1$  time-slots, and then perform our Hyperbolic K-means using average data as 3rd dimension. This idea however simple, is not appropriate, because it flattens the trace and loses all information over RU complementarity in different time-slots. Imagine for example that in a certain slot two stations  $i$  and  $j$  are complementary with  $w_i > w_j$ , whereas a couple of slots later the situation is inverted with  $w_i < w_j$ ; the two stations still are complementary as pair. If we average over these two time slots, the complementarity is lost, because the two RUs have similar values of average traffic. To avoid such loss of information, we introduce **Algorithm 3**, which extends our Hyperbolic K-means over multiple time-slots. Let  $T$  be the number of consecutive time-slots to consider. For each time slot  $t$ , we apply

Algorithm 3 using the resource values  $w(n, t)$  measured per RU  $n$  and for the specific time slot  $t$ . Obviously the RU 2D-euclidean positions  $(x_n, y_n)$  do not change over time, but the cluster centroids can change because of the third dimension. Let  $P(t) = \{C_k(t), k = 1, \dots, K\}$  be the obtained clustering per time slot  $t$ , and let  $\varphi(t) = \{m_k(t), k = 1, \dots, K\}$  be the set of cluster centroids of  $P(t)$ . It is important to note that every partition has exactly  $K$  centroids (in the degenerate case two or more centroids may be overlapping).

The robust algorithm works as follows: The centroids of the first time-slot  $\varphi(1)$  are considered as reference. For each centroid  $m_k(1)$  in the set, find the  $(T-1)$  centroids  $E2$  closest to it from the centroid sets of the next  $T-1$  partitions  $\varphi(t)$ . Then calculate their average on each of the three dimensions  $(x, y, w)$ . In the special case with  $T = 2$ , for  $m_k(1) = (\bar{x}_k(1), \bar{y}_k(1), \bar{w}_k(1))$  the  $k$ -th centroid obtained at time-slot  $t = 1$ , we search for  $m_{k'}(2) = (\bar{x}_{k'}(2), \bar{y}_{k'}(2), \bar{w}_{k'}(2))$ , the  $E2$  closest centroid to  $m_{k'}(2)$  from an execution at  $t = 1$ . The resulting robust centroid is the average of each dimension of these two centroids  $m_k(1)$  and  $m_{k'}(2)$ . We finally remap the RRHs according to the new centroids using the  $H3$  distance.

---

**Algorithm 3:** Robust Hyperbolic K-means for RU clustering

---

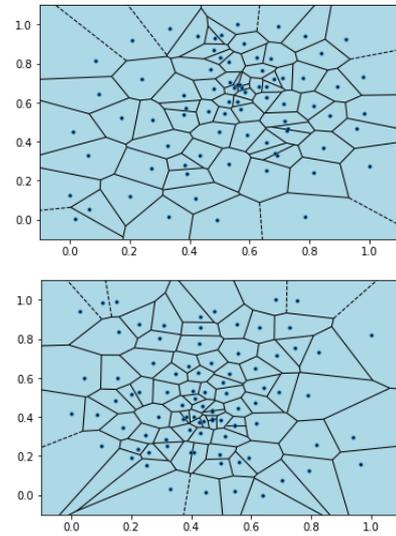
1. For each time slot  $t = 1, \dots, T$ :
    - (a) Execute **Algorithm 2** for all  $t = 1, \dots, T$  and obtain the set of centroids  $\varphi(t) = [m_1(t), \dots, m_K(t)]$ .
  2. Given the first set of centroids  $\varphi(1)$ , for each  $m_k(1) \in \varphi(1), k = 1, \dots, K$ :
    - (a) Get the closest centroid ( $E2$ ) to  $m_k(1)$  from each partition  $\varphi(t), t = 1, \dots, T$ .
    - (b) Compute the average  $k$ -th centroid over all time realisations.
  3. Assign each RU to the clusters according to their  $H3$  distance to the new average (robust) centroids.
- 

## 5. Evaluation

In this section, we use real-world mobile data provided by Orange, France for two cities (Lille and Nantes). First, we study the effect of parameters on the performance of the Hyperbolic K-means (we consider the robust extension of the algorithm for the rest of this section, and so we use the notation *Hyperbolic K-means* and *Robust Hyperbolic K-means* interchangeably). Then, we apply the metrics mentioned in Section 2.1 to compare the utilization and cost of our algorithm (**Algorithm 3**) against other existing solutions: (i) the K-means clustering using the 2D-euclidean distance (considering just the RU position without the traffic / resources), (ii) the 3D-euclidean distance (with remaining resources as third feature), (iii) the DCCA clustering.

### 5.1. Datasets description

We use for our evaluation mobile data provided by Orange, France that contains four months of traffic data from



**Figure 3:** Voronoi partitions of Lille (up) and Nantes (down)

2019-03-19 to 2019-06-16 for Lille and from 2019-03-16 to 2019-06-16 for Nantes. In addition, we use a dataset of the RUs positions for each city. The details of these datasets are listed in Table 2.

Lille and Nantes contain respectively  $N = 88$  and  $N = 97$  RUs. An RU position consists of its geographical coordinates  $(x_n, y_n)$  in the Lambert II Carto projection system. Fig. 3 shows how the areas of the two cities are partitioned in a Voronoi diagram.

The traffic dataset contains per antenna, its traffic in Bytes-Up and Bytes-Down considered in 10-minute-long time-slots. Fig. 4 shows an example of a traffic volume of two antennas, we observe that the Bytes-Down traffic volume is more important than Bytes-Up, and thus is more representative. Therefore, we will only consider the Bytes-Down traffic for evaluation.

### 5.2. Evaluation Plan

In order to evaluate the algorithms, we construct a typical week traffic profile according to the *Base Stations Traffic Profile Generation* method proposed in [6].

Given our traffic dataset, for each RU, we aggregate and average the traffic of each timeslot of each day of the week over the four months of traffic data to construct the typical traffic of the week, as shown in Fig. 5.

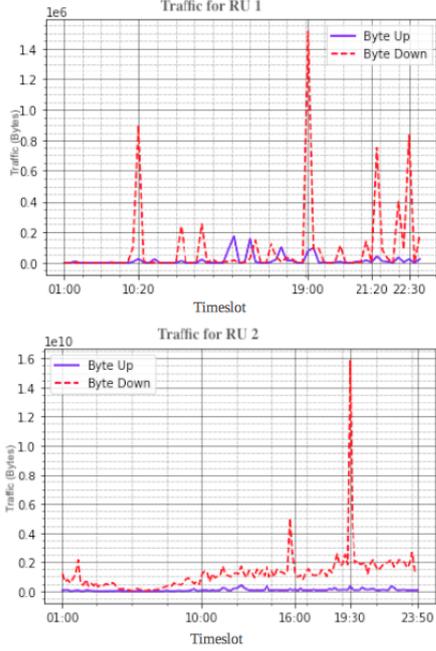
The traffic and positions are normalized for evaluation by the following method: for each  $(x_n, y_n, z_n)$  from the dataset, where  $x_n$  is the first geographical coordinate,  $y_n$  is the second geographical coordinate and  $z_n$  is the traffic, the corresponding normalized values are :

$$\begin{aligned} \tilde{x}_n &= \frac{x_n - \min \mathcal{X}}{\max \mathcal{X} - \min \mathcal{X}}, \tilde{y}_n = \frac{y_n - \min \mathcal{Y}}{\max \mathcal{Y} - \min \mathcal{Y}}, \\ \tilde{z}_n &= \frac{z_n - \min \mathcal{Z}}{\max \mathcal{Z} - \min \mathcal{Z}} \end{aligned} \quad (15)$$

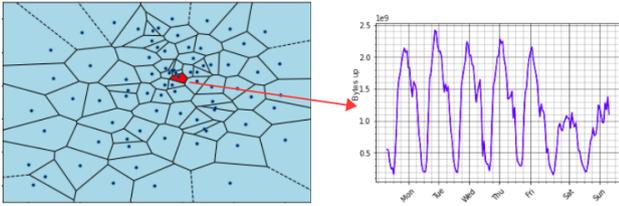
In the above  $(\mathcal{X}, \mathcal{Y})$  is the dataset over the positions, and  $\mathcal{Z}$  is the dataset of traffic for the considered period of clus-

**Table 2**  
Data Description

Dataset	Lille	Nantes
Number of antennas	1394	1413
Number of RU positions	88	97
Data collection period	2019-03-19 to 2019-06-16	2019-03-16 to 2019-06-16
Time-slot duration	10-minute	10-minute
Maximal traffic Bytes-Up	$7.25 \cdot 10^9$	$3.04 \cdot 10^9$
Minimal traffic Bytes-Up	0.0	0.0
Maximal traffic Bytes-down	$15.82 \cdot 10^9$	$18.15 \cdot 10^9$
Minimal traffic Bytes-down	0.0	0.0



**Figure 4:** Example of Bytes-Up and Bytes-Down patterns of two RUs



**Figure 5:** An example of the traffic pattern from a chosen RU.

tering. The normalised values  $(\tilde{x}_n, \tilde{y}_n, \tilde{z}_n)$  will be used in the evaluation metrics of utilisation (4) and cost (5). Furthermore, we define the normalised resources  $\tilde{w}_n = 1 - \tilde{z}_n$  for all stations and time-slots, which will be used in the Robust Hyperbolic K-means algorithm.

We set the duration  $T$  of consecutive time slots to  $T = 24h$  for all of our evaluations. The specific choice of  $T$  is motivated by the observations in 5, which shows that traffic pattern is different on a daily basis. In fact the most pronounced

difference is observed between weekdays and weekend, so we could have taken a more conservative choice. We would like to comment here, that selection of shorter periods, e.g. cluster per 1 hour or per 6 hours results in frequent changes in partitions, without significant global improvement in performance compared to our current selection  $T = 24h$ .

For our experiments, we repeat each algorithm  $\tau_{out} = 50$  times for each value of the parameter being studied and compute the performance metrics for each execution, we then use the average value of the metrics of interest from the 50 executions to plot our figures.

### 5.3. Parameter Effect Study

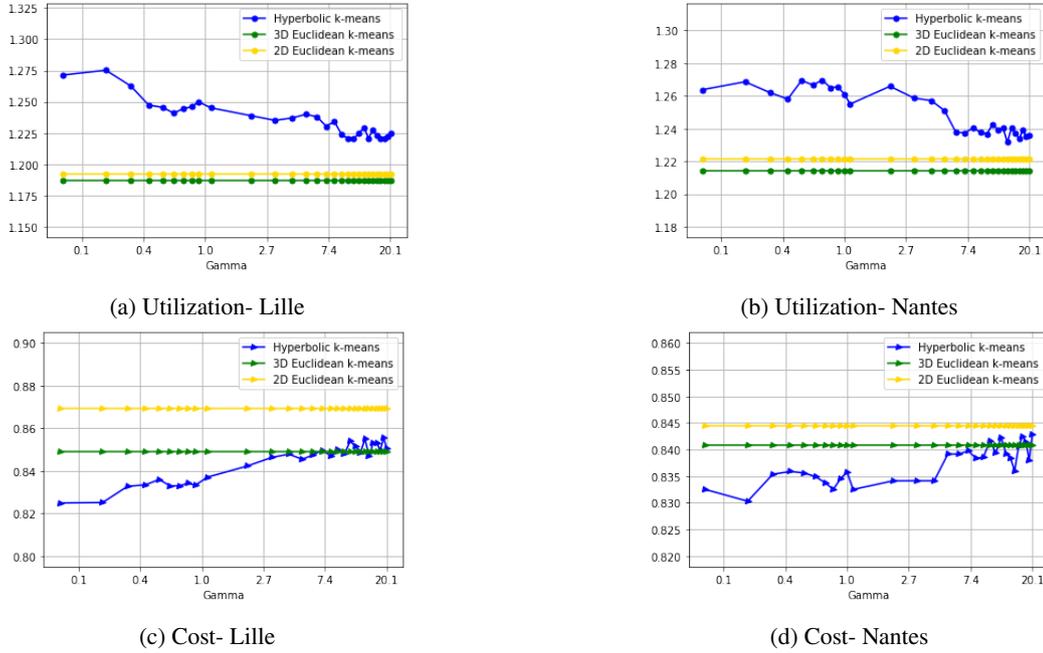
We start by evaluating the effect of certain design parameters in the Robust Hyperbolic K-means: the  $\gamma$  scaling parameter and the number  $K$  of clusters. We evaluate the cost and utilization for different values of these parameters and determine an optimal set that we use later for comparison against the existing clustering algorithms.

#### 5.3.1. The scaling parameter $\gamma$

The Hyperbolic  $H3$  distance used in the K-means has been defined in (11). In this expression, the scaling parameter  $\gamma$  was introduced to determine through scaling, whether the focus should be on the RU positions ( $\gamma$  large) or their traffic ( $\gamma$  small). For very large values of  $\gamma$ , only the first summand in the argument is important, putting the focus almost entirely on the RUs positions. In this case the behaviour of Hyperbolic K-means should be similar to that of 2D-Euclidean K-means.

We run the Robust Hyperbolic K-means (**Algorithm 3**) on the normalized traffic data for a number of clusters  $K = 6$  and various values of  $\gamma \in (0, 20)$  to evaluate its impact on the utilization and cost of the obtained partitions. We also run for comparison the two Euclidean K-means algorithms (2D- and 3D-), which do not depend on  $\gamma$ , for the same number of clusters  $K = 6$ .

**Take-away 1:** We observe in Fig. 6a and Fig. 6b that the Robust Hyperbolic K-means shows significant improvement in utilization performance compared to the euclidean K-means algorithms for values of  $\gamma \leq 7$ , for values around  $\gamma = 1$ , we notice a 5.1% improvement for Lille (resp. 5.88% for Nantes) in utilization compared to  $E3$  and a 4.2% improvement for Lille (resp. 3.27% for Nantes) compared to  $E2$ . For higher values of  $\gamma$  all three algorithms tend to con-



**Figure 6:** Utilization and Cost of the k-means clustering using Hyperbolic, Euclidean 2D and Euclidean 3D distances according to  $\gamma$ .

verge to equal utilization performance. Fig. 6c and Fig. 6c show that for, small values of  $\gamma$  we get less cost. Small  $\gamma$  leads to clusters with wide geographical spread (diameter) as Fig. 7 indicates. In this figure we evaluate the diameter of a partition,<sup>1</sup> defined here as the max distance between two RUs in the same cluster (for all clusters in the partition)

$$Diam(P) = \max_k \max_{i,j \in C_k} d_{E2}^2(i, j). \quad (16)$$

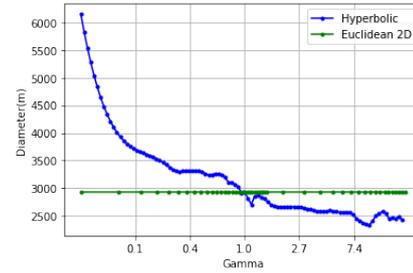
In the above,  $d_{E2}^2(i, j)$  applies the 2D-euclidean distance between the positions of  $RU_i$  and  $RU_j$ . The figure shows that for  $\gamma < 0.2$  the cluster diameter increases considerably.

Since the main role of  $\gamma$  is to appropriately scale the 2D-distance and the resource dimension to achieve homogeneity in the  $H3$  expression, we choose for the rest of the evaluation tests  $\gamma=1$  since our data-sets are normalized in  $[0, 1]$  for both the traffic and the RU positions.

### 5.3.2. The number of clusters $K$

We run all three K-means algorithms (with  $H3$ ,  $E2$  and  $E3$ ) for various values of  $K$  ( $\gamma=1$ ) according to the evaluation protocol mentioned previously : for each value of  $K$ , we repeat the algorithms 50 times and compute the performance metrics for each execution. We plot the mean of the 50 execution per value of  $K$  in Fig. 8 . We observe, in Fig. 8a and Fig. 8b that the utilization decreases as  $K$  increases, whereas, in Fig. 8c and Fig. 8d, the cost increases with  $K$ . Both metrics tend to 1 for large  $K$ . This is explained by the fact that the higher the number of clusters, the more the clustering performance resembles that of the traditional RAN

<sup>1</sup>We plot the diameter in meters by de-normalizing the values according to the reverse process of equation (15).

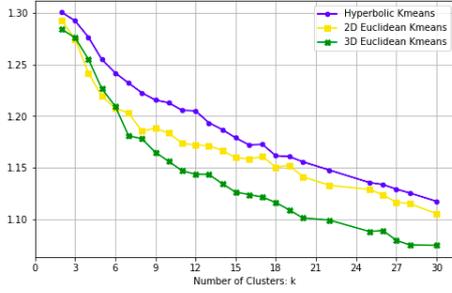


**Figure 7:** Variation of the clusters diameters according to parameter  $\gamma$ .

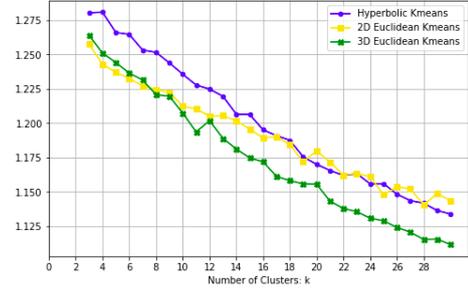
architecture. On the other hand, smaller numbers of clusters  $K$  have better performance, but their diameter explodes (non-feasible in practice). For values around  $K = 7$  the benefits are still considerable: for Lille city, the utilization is improved by 4.31% (resp. 2.28% for Nantes) compared to  $E3$  and by 2.41% (resp. 2.59% for Nantes) compared to  $E2$ . The cost is also 2.07% less than  $E2$  (resp. 0.60% for Nantes) and 0.54% less than  $E3$  (resp. 0.39% for Nantes)

Fig. 9 plots the maximal cluster diameter per partition for each of the three clustering schemes, obtained for different values of  $K$ . We notice that for  $K > 5$ , the maximal cluster diameter (Lille) is less than 3km. We choose, for the rest of the evaluations,  $K \in [5, 10]$  which is a good compromise between the distance threshold and the traffic Utilization/deployment Cost metrics.

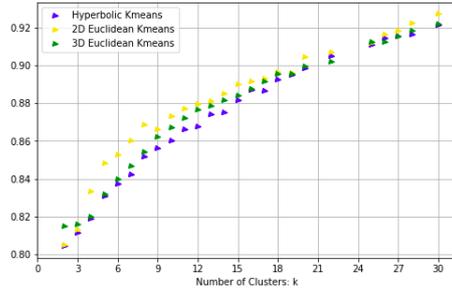
**Take-away 2:** The Robust Hyperbolic  $K$ -means outperforms both euclidean schemes in the metrics of utilization and cost in the whole range of  $K$  values, i.e., for any num-



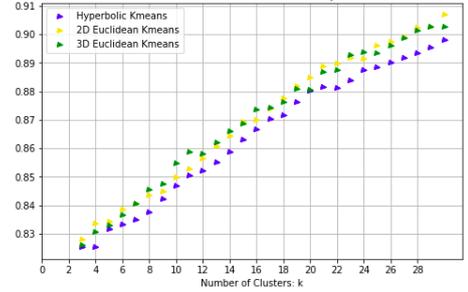
(a) Utilization- Lille



(b) Utilization- Nantes

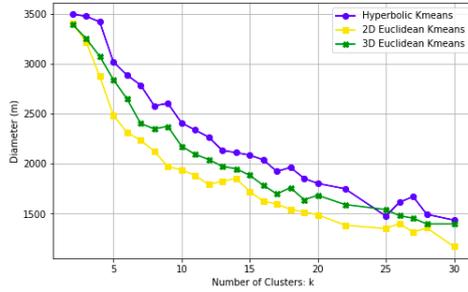


(c) Cost- Lille

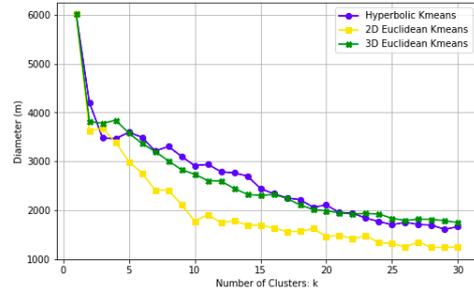


(d) Cost- Nantes

**Figure 8:** Utilization and Cost of the K-means clustering using Hyperbolic, 2D-Euclidean and 3D-Euclidean distances with varying  $K$ .



(a) Lille



(b) Nantes

**Figure 9:** Maximal cluster diameter for k-means clustering using Hyperbolic, Euclidean 2D and Euclidean 3D distances according to  $K$ .

ber of clusters, as Fig. 8 shows (For Nantes city, the 2D-euclidean scheme has closer results to the Robust Hyperbolic K-means in utilization metric but in the cost metric, they are less so). This improvement in performance is achieved by forming clusters having a diameter comparable to that of 3D-euclidean, and a bit higher than 2D-euclidean, as Fig. 9a and Fig. 9b illustrate. The difference with 2D-euclidean in diameter is of the order of 500m for the city of Lille and 1km for the city of Nantes. Hence, the Robust Hyperbolic K-means spreads a bit more the cluster sizes to achieve higher performance, while keeping them as compact as possible. Another interesting observation from Fig. 9 is that the diameter for  $H3$  and  $E3$  K-means becomes comparable for larger number  $K$ . This is because for small 2D-euclidean distance, the hyperbolic and  $E3$  distance tend to behave more similarly, as Fig. 13 shows in the Appendix.

Note, here, that our aim is not to determine the optimal number  $K$  or the appropriate cluster size. As explained in

Section 2, this is a complex architectural question, that also depends on the services that need to be supported. For this reason, the above plots show clustering performance for a large range of values  $K$ . We observe that our method is profitable for the whole range of values  $K$ .

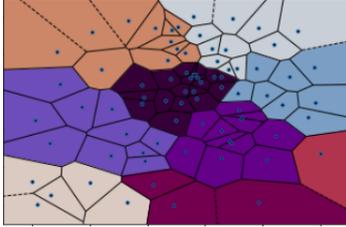
#### 5.4. Comparison with DCCA

Having studied the choice parameters for the multiple variations of K-means, we now compare the clustering schemes with the selected set of parameters. We evaluate the Robust Hyperbolic ( $H3$ ) K-means (**Algorithm 3**) against the 2D-Euclidean ( $E2$ ) K-means that considers only RU positions, and especially the DCCA algorithm from [7]. We further provide a detailed comparison with DCCA over the number of clusters and the maximal diameter.

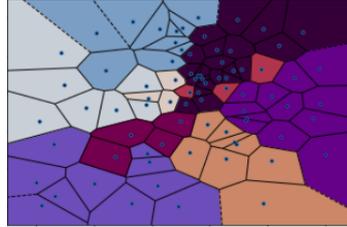
We choose  $K = 9$  clusters and Table 3 summarises the results for both cities of Lille and Nantes. We remind that the utilization and cost metrics for the traditional RAN archi-

**Table 3**  
Comparative Results

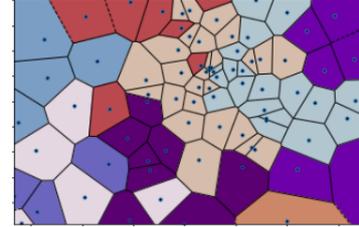
Method	Lille City			Nantes City		
	Inputs	Util.	Cost	Inputs	Util.	Cost
<b>Hyperbolic <math>H3</math></b>	$K = 9, \gamma=1, T = 24h$	1.23	0.85	$K = 9, \gamma=1, T = 24h$	1.25	0.83
<b>DCCA</b>	Threshold $\sigma = 3.5km$	1.17	0.86	Threshold $\sigma = 4.7km$	1.19	0.84
<b>Euclidean <math>E2</math></b>	$K = 9$	1.13	0.86	$K = 9$	1.17	0.88



(a) 2D Euclidean K-means.  
Utilization: 1.13, Cost: 0.86



(b) 3D Hyperbolic K-means.  
Utilization: 1.23, Cost: 0.85



(c) DCCA clustering.  
Utilization: 1.17, Cost: 0.86

**Figure 10:** Clusters for Lille City with: 2D-Euclidean K-means, 3D-Hyperbolic K-means, and DCCA.

ecture are both equal to 1. We observe that the Robust Hyperbolic K-means effectively improves the utilization metric by 23% for Lille and by 25% for Nantes compared to standard RAN, while DCCA improves it by 17% and 19%, respectively. Both methods reduce the cost by around 15%. The 2D Euclidean K-means achieves an improvement in utilization of 13% for Lille and 17% for Nantes compared to standard RAN, lower than the other two.

We need to explain here that, contrary to the K-means variations, the DCCA method uses as input a threshold  $\sigma$  that limits the acceptable diameter and it outputs the DCCA partition with some number of clusters  $K$ . Each threshold leads to a different  $K$ . We selected a DCCA diameter threshold which led to a partition with 9 clusters. We then set  $K = 9$  as input for the hyperbolic and euclidean K-means variations, to guarantee a fair comparison between the methods.

To better understand the differences between schemes, we visualize in Fig. 10 the clusters that each method generates for the city of Lille.

**Take-away 3:** The Robust Hyperbolic K-means produces clusters that are geographically compact, in a similar fashion to the 2D-Euclidean K-means. To better balance the clusters over traffic, the method spreads geographically these clusters while trying to keep them in a compact form, in contrast to the DCCA algorithm.

We now vary the number of clusters  $K$  for the K-means and compare the performance between DCCA and the Robust Hyperbolic K-means. More precisely, we run DCCA using various values of the maximum acceptable diameter. Each value generates the number  $K$  of formed clusters as output. We use the resulting number  $K$  of clusters per realization as entry to the Hyperbolic K-means. The performance results of the two methods for the two cities and in a time-window of 24 hours are illustrated in Fig. 11.

**Take-away 4:** We observe that in the entire range of  $K$

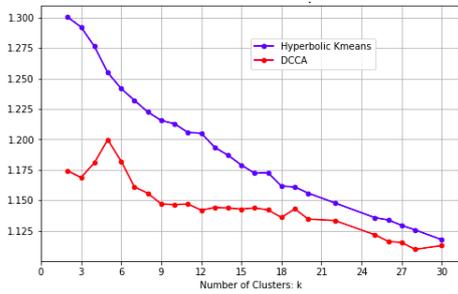
(and thus for any equivalent input threshold of the DCCA), the Robust Hyperbolic K-means offers much higher utilization over time, with a cost similar to the DCCA. The improvement is more pronounced for a smaller number of clusters (as expected): Using  $K = 3$ , for Lille city, the improvement in utilization can go up to 10.56% (resp. 6.42% for Nantes). For moderate number of clusters  $K = 7$  the utilization gain is 6.11% (resp. 5.50% for Nantes).

We have discussed throughout the work, that each cluster needs to respect a distance constraint related to the length of the optical fibre. This was used as hard constraint in the input of DCCA. The Robust Hyperbolic K-means achieves to produce the same number of clusters with a geographic spread - measured by the diameter  $Diam$  - much smaller compared to the DCCA. The comparison of the two methods over  $Diam$  is illustrated in Fig. 12. We can read the figure as follows:

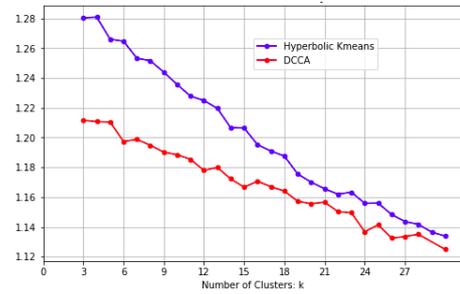
**Take-away 5:** Given some distance constraint ( $y$ -axis) the number of clusters  $K$  that respect this is considerably smaller in the (robust) Hyperbolic K-means, than in the DCCA. Hence, the  $H3$  method can produce a small number of compact clusters with balanced traffic (high utilization), using the same number of BBU resources (cost) as the DCCA method.

## 6. Conclusion

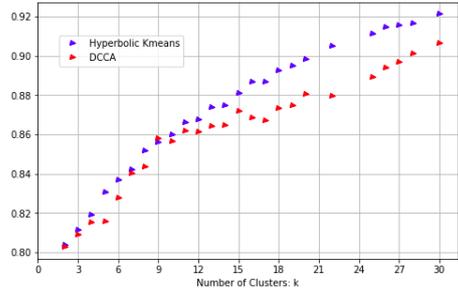
In this work, we have addressed one of the challenges in the C-RAN and vRAN architecture, the RU clustering into locally centralised BBU-pools (or RU-CU in the vRAN case). Our objective was to propose a clustering scheme that maximises the resource utilization, that uses reduced number of resources, and respects some distance limitations related to the cluster diameter. To achieve this, we have been based on the vanilla K-means algorithm and proposed a variation that considers the traffic (or available resources) as third dimension. Instead of embedding the features in eu-



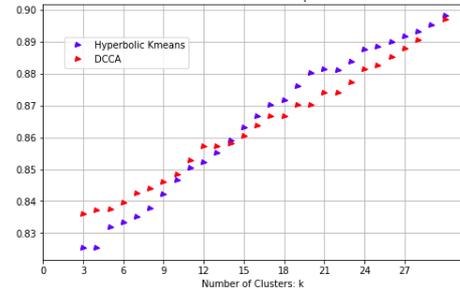
(a) Utilization- Lille



(b) Utilization- Nantes

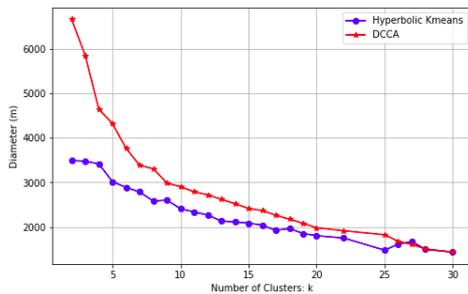


(c) Cost- Lille

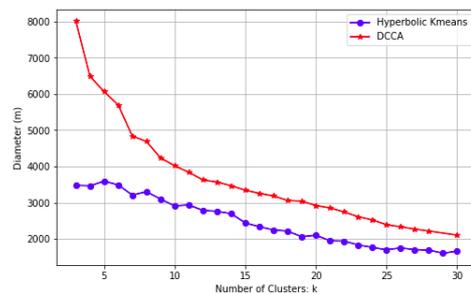


(d) Cost- Nantes

**Figure 11:** Utilization and Cost for Robust Hyperbolic K-means and DCCA over varying number of clusters.



(a) Lille



(b) Nantes

**Figure 12:** Cluster diameters of Robust Hyperbolic K-means and DCCA over varying numbers of clusters.

clidean space, we used the hyperbolic space, because the induced distance has a behaviour that can facilitate the formation of clusters with the desired properties. The resulting clusters from this novel approach further exhibit fairness in the aggregate load. We further made the algorithm robust over multiple time-slots.

To evaluate performance, we have made use of real mobile data from Orange Mobile France, for the cities of Lille and Nantes. Our method effectively reduces deployment cost by 15% and improves resources utilization by 23 – 25%. It outperforms both euclidean K-means as well as state-of-the-art methods from the literature (DCCA).

Rather noteworthy is the fact that the hyperbolic K-means algorithm proposed in this work has generality. It need not be restricted to problems of RU-BBU association, but could be applied to any clustering problem that deals with various features which need not necessarily be treated homogeneously. It is very useful for all clustering problems that require to group together nodes with complementarity over

some feature. The design parameters  $\gamma$  (related to units) and the temporal window of clustering are flexible tuning parameters that allow for performance improvement. The hyperbolic embedding has been proven very useful in this setting, adding further to the arguments in favor of this novel viewpoint.

## Acknowledgement

This work was partially supported by the ANR CAN-CAN project (ANR-18-CE25-0011).

## A. Appendix: $H3$ distance properties

To get a better understanding on the properties of the hyperbolic distance  $H3$  how this compares to the 3D-euclidean  $E3$ , and how it can be useful for our specific clustering problem, we introduce here a transformation of the pair of resource variables  $(w_i, w_j)$ . The imbalance of the two variables will be described by their ratio  $\rho > 0$ . Furthermore, let us fix their sum to  $\lambda > 0$  resource units. Then, the tuple  $(w_i, w_j)$  can be uniquely expressed as a function of  $(\rho, \lambda)$  and vice versa,

$$\left. \begin{array}{l} \lambda = w_i + w_j \\ \rho = w_i/w_j \end{array} \right\} \Leftrightarrow \left. \begin{array}{l} w_i = \rho \cdot \lambda / (1 + \rho) \\ w_j = \lambda / (1 + \rho) \end{array} \right\}. \quad (17)$$

Let us also denote the 2D euclidean distance of the RU positions by  $\delta = d_{E2}(x_i, y_i, x_j, y_j)$ . The hyperbolic distance  $H3$  in (11) can be expressed as a function of  $(\delta, \rho, \lambda)$  (we assume  $\gamma = 1$  w.l.o.g.)

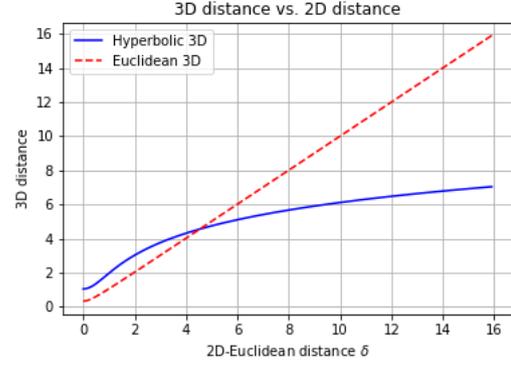
$$d_{H3}(\delta, \rho, \lambda) = \operatorname{arccosh} \left( \frac{\delta^2}{2\rho \frac{\lambda^2}{(1+\rho)^2}} + \frac{1}{2} \left( \rho + \frac{1}{\rho} \right) \right). \quad (18)$$

In a similar way, the 3D euclidean distance ( $E3$ ) can also be expressed by the same arguments

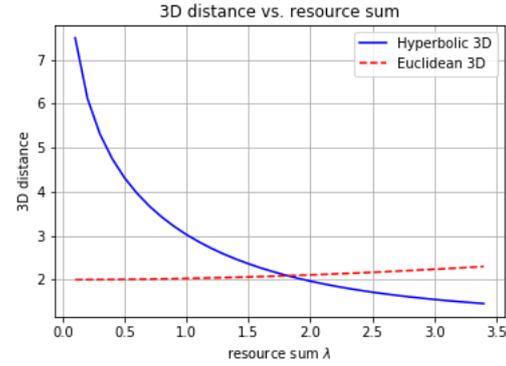
$$d_{E3}(\delta, \rho, \lambda) = \sqrt{\delta^2 + \lambda^2 \frac{(1-\rho)^2}{(1+\rho)^2}}. \quad (19)$$

We already observe a difference in the expressions (19) and (18). The 3D-euclidean distance is increasing in the sum of resources  $\lambda$ , whereas the hyperbolic distance is decreasing (remember  $\operatorname{arccosh}(s)$  is an increasing function of  $s$ ). Hence, the 3D-euclidean distance will perceive two RUs having low remaining resources as being close to each other, something which leads to inefficient clustering for C-RANs. To further understand why the hyperbolic distance is more appropriate in our scenario, we will keep two parameters fix in the above expressions and plot its response when increasing the third parameter.

- Fix  $\lambda$  and  $\rho$ , vary the 2D-euclidean distance  $\delta$ . The plot is in Fig. 13 for both distances. We observe that the 3D-euclidean increases proportionally to the 2D-euclidean with unit slope. The hyperbolic distance, on the other hand, deforms the influence of the 2D-distance. The hyperbolic is a concave function of the 2D-euclidean and increases with a slope less than 1. As a result, RUs that are distant in the 2D-sense are perceived closer through the hyperbolic lens.
- Fix  $\delta$  and  $\rho$ , vary the sum of resources  $\lambda$ . The plot is in Fig. 14 for both distances and we can see the difference in their behaviour. In contrast to the 3D-euclidean, the hyperbolic is monotone decreasing in  $\lambda$ , hence two RUs with small remaining resource sum are perceived far from each other and should not be placed in the same cluster. What is also striking is that the downward slope of  $H3$  is very large in the whole



**Figure 13:** Hyperbolic and euclidean 3D-distance when varying the 2D-euclidean distance  $\delta$ , for fixed  $\lambda = 1$  and  $\rho = 2$ .



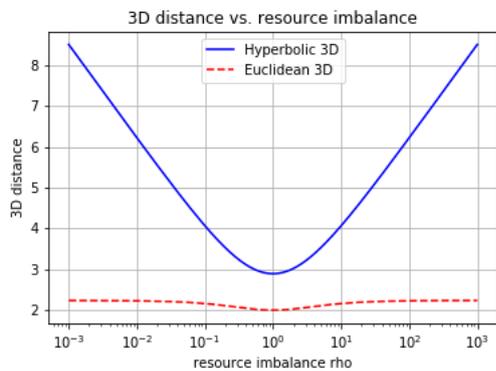
**Figure 14:** Hyperbolic and euclidean 3D-distance when varying the resource sum  $\lambda$ , for fixed  $\delta = 2$  and  $\rho = 2$ .

range of  $\lambda$  values (diminishing for very large  $\lambda$ ), which indicates that this distance is very sensitive to a small change in the resource sum. This comes in striking contrast with  $E3$  which is increasing in  $\lambda$  with a very small slope, hence quite insensitive.

- Fix  $\delta$  and  $\lambda$ , vary the imbalance ratio  $\rho = w_i/w_j$ . The plot is in Fig. 15 for both distances. The range of values is chosen in  $[0.001, 1000]$ , so for  $\rho < 1$  the resources are imbalanced in favor of  $w_j$  and for  $\rho > 1$  it is the other way round. We see that both the hyperbolic and the 3D-euclidean are symmetric, with axis of symmetry  $\rho = 1$ . But the  $H3$  is much more sensitive to resource imbalance than  $E3$ . As the plot illustrates,  $H3$  increases fast due to imbalance, in a symmetric fashion around  $\rho = 1$ , with a minimum when the two RUs are completely balanced, i.e. have the same number of remaining resources.

## References

- [1] Álvarez-Corrales, L., Giovanidis, A., Martins, P., Decreusefond, L., 2017. Wireless node cooperation with resource availability constraints, in: 2017 15th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt), pp. 1–8. doi:10.23919/WIOPT.2017.7959946.



**Figure 15:** Hyperbolic and euclidean 3D-distance when varying the imbalance ratio  $\rho$ , for fixed  $\lambda = 1$  and  $\delta = 2$ .

[2] Bhaumik, S., Chandrabose, S.P., Jataprolu, M.K., Kumar, G., Muraidhar, A., Polakos, P., Srinivasan, V., Woo, T., 2012. CloudIQ: A framework for processing base stations in a data center, in: Proceedings of the 18th Annual International Conference on Mobile Computing and Networking, Association for Computing Machinery, New York, NY, USA. p. 125–136. URL: <https://doi.org/10.1145/2348543.2348561>, doi:10.1145/2348543.2348561.

[3] Boulou, K., El Helou, M., Lahoud, S., 2015. RRH clustering in cloud radio access networks, in: 2015 International Conference on Applied Research in Computer Science and Engineering (ICAR), IEEE, Lebanon. pp. 1–6. URL: <http://ieeexplore.ieee.org/document/7338135/>, doi:10.1109/ARCSE.2015.7338135.

[4] Ceselli, A., Fiore, M., Furno, A., Premoli, M., Secci, S., Stanica, R., 2018. Prescriptive analytics for mec orchestration, in: 2018 IFIP Networking Conference (IFIP Networking) and Workshops, pp. 1–9. doi:10.23919/IFIPNetworking.2018.8696508.

[5] Checko, A., Christiansen, H.L., Yan, Y., Scolari, L., Kardaras, G., Berger, M.S., Dittmann, L., 2015. Cloud ran for mobile networks—a technology overview. *IEEE Communications Surveys Tutorials* 17, 405–426.

[6] Chen, L., Liu, L., Fan, X., Li, J., Wang, C., Pan, G., Jakubowicz, J., Nguyen, T.M.T., 2017. Complementary base station clustering for cost-effective and energy-efficient cloud-RAN, in: 2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCOM/IOP/SCI), IEEE, San Francisco, CA. pp. 1–7. URL: <https://ieeexplore.ieee.org/document/8397526/>, doi:10.1109/UIC-ATC.2017.8397526.

[7] Chen, L., Yang, D., Zhang, D., Wang, C., Li, J., Nguyen, T.M.T., 2018. Deep mobile traffic forecast and complementary base station clustering for C-RAN optimization. *Journal of Network and Computer Applications* 121, 59–69. URL: <https://linkinghub.elsevier.com/retrieve/pii/S1084804518302455>, doi:10.1016/j.jnca.2018.07.015.

[8] China-Mobile, 2011. White paper: C-RAN The Road Towards Green RAN. Technical Report Version 2.5. China Mobile Research Institute.

[9] China-Unicom, 2017. China Unicom Edge Computing Technology White Paper. Technical Report 2017-06. China Unicom Network Technology Research Institute.

[10] Cisco, 2017. White paper: Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2017-2022. Technical Report. Cisco, San Jose, CA, USA.

[11] da Silva Coelho, W., Benhamiche, A., Perrot, N., Secci, S., 2020. On the impact of novel function mappings, sharing policies, and split settings in network slice design, in: 2020 16th International Conference on Network and Service Management (CNSM), pp. 1–9. doi:10.23919/CNSM50824.2020.9269106.

[12] Decreusefond, L., Ferraz, E., Martins, P., Vu, T.T., 2012. Robust

methods for lte and wimax dimensioning, in: 6th International ICST Conference on Performance Evaluation Methodologies and Tools, pp. 74–82. doi:10.4108/valuetools.2012.250209.

[13] Garcia-Saavedra, A., Iosifidis, G., Costa-Perez, X., Leith, D.J., 2018. Joint optimization of edge computing architectures and radio access networks. *IEEE Journal on Selected Areas in Communications* 36, 2433–2443. doi:10.1109/JSAC.2018.2874142.

[14] Gerald, T., Zaatiti, H., Hajri, H., Baskiotis, N., Schwander, O., 2020. From node embedding to community embedding : A hyperbolic approach.

[15] Hajri, H., Zaatiti, H., Hébrail, G., Aknin, P., 2019. Apprentissage automatique sur des données de type graphe utilisant le plongement de poincaré et les algorithmes stochastiques riemanniens., in: Conférence Nationale d’Intelligence Artificielle Année 2019.

[16] I, C.L., Huang, J., Duan, R., Cui, C., Jiang, J., Li, L., 2014. Recent Progress on C-RAN Centralization and Cloudification. *IEEE Access* 2, 1030–1039. URL: <https://ieeexplore.ieee.org/document/6882182/>, doi:10.1109/ACCESS.2014.2351411.

[17] James, G., Witten, D., Hastie, T., Tibshirani, R., 2013. An Introduction to Statistical Learning. volume 103 of *Springer Texts in Statistics*. Springer New York, New York, NY. URL: <http://link.springer.com/10.1007/978-1-4614-7138-7>, doi:10.1007/978-1-4614-7138-7.

[18] Kiyoshima, K., Takiguchi, T., Kawabe, Y., Sasaki, Y., 2015. Commercial development of lte-advanced: Applying advanced c-ran. *NTT Docomo Techn. Journ.* 17, 3098–3130.

[19] Krioukov, D., Papadopoulos, F., Kitsak, M., Vahdat, A., Boguñá, M., 2010. Hyperbolic geometry of complex networks. *Phys. Rev. E* 82, 036106. URL: <https://link.aps.org/doi/10.1103/PhysRevE.82.036106>, doi:10.1103/PhysRevE.82.036106.

[20] Lin, Y., Shao, L., Zhu, Z., Wang, Q., Sabhikhi, R.K., 2010. Wireless network cloud: Architecture and system requirements. *IBM Journal of Research and Development* 54, 4:1–4:12.

[21] Loustau, B., 2020. Hyperbolic geometry.

[22] Maeder, A., Lalam, M., De Domenico, A., Pateromichelakis, E., Wäijbén, D., Bartel, J., Fritzsche, R., Rost, P., 2014. Towards a flexible functional split for cloud-ran networks, in: 2014 European Conference on Networks and Communications (EuCNC), pp. 1–5. doi:10.1109/EuCNC.2014.6882691.

[23] Nickel, M., Kiela, D., 2017. Poincaré embeddings for learning hierarchical representations, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, Curran Associates Inc., Red Hook, NY, USA. p. 6341–6350.

[24] Parvez, I., Rahmati, A., Guvenc, I., Sarwat, A.I., Dai, H., 2018. A survey on low latency towards 5g: Ran, core network and caching solutions. *IEEE Communications Surveys Tutorials* 20, 3098–3130. doi:10.1109/COMST.2018.2841349.

[25] Pizzinat, A., Chanclou, P., Saliou, F., Diallo, T., 2015. Things you should know about fronthaul. *Journal of Lightwave Technology* 33, 1077–1083. doi:10.1109/JLT.2014.2382872.

[26] Wang, X., Thota, S., Tornatore, M., Chung, H.S., Lee, H.H., Park, S., Mukherjee, B., 2016. Energy-efficient virtual base station formation in optical-access-enabled cloud-ran. *IEEE Journal on Selected Areas in Communications* 34, 1130–1139. doi:10.1109/JSAC.2016.2520247.

[27] Wang, X., Wang, L., Elayoubi, S.E., Conte, A., Mukherjee, B., Cavdar, C., 2017. Centralize or distribute? a techno-economic study to design a low-cost cloud radio access network, in: 2017 IEEE International Conference on Communications (ICC), pp. 1–7. doi:10.1109/ICC.2017.7996771.

[28] Wu, J., Zhang, Z., Hong, Y., Wen, Y., 2015. Cloud radio access network (C-RAN): a primer. *IEEE Network* 29, 35–41. URL: <http://ieeexplore.ieee.org/document/7018201/>, doi:10.1109/MNET.2015.7018201.

[29] Yu, H., Musumeci, F., Zhang, J., Xiao, Y., Tornatore, M., Ji, Y., 2020. Du/cu placement for c-ran over optical metro-aggregation networks, in: Tzanakaki, A., Varvarigos, M., Muñoz, R., Nejabati, R., Yoshikane, N., Anastasopoulos, M., Marquez-Barja, J. (Eds.), *Optical Network Design and Modeling*, Springer International Publishing, Cham. pp. 82–93.