Distributed Resource Management in Downlink Cache-Enabled Multi-Cloud Radio Access Networks

Robert-Jeron Reifert, Student Member, IEEE, Alaa Alameer Ahmad, Member, IEEE, Hayssam Dahrouj, Senior Member, IEEE, Anas Chaaban, Senior Member, IEEE, Aydin Sezgin, Senior Member, IEEE, Tareq Y. Al-Naffouri, Senior Member, IEEE, and Mohamed-Slim Alouini, Fellow, IEEE

Abstract—A compound of several clouds, jointly managing large-scale inter-cloud and intra-cloud interference, promises to be a practical solution to account for the ambitious premises of beyond fifth generation networks. This paper considers a multi-cloud radio access network (MC-RAN), where each cloud is connected to a distinct set of cache-enabled base stations (BSs) via limited capacity fronthaul links. The BSs are equipped with local cache storage and baseband processing capabilities, as a means to alleviate the fronthaul congestion problem. The paper then investigates the problem of jointly assigning users to clouds and determining their beamforming vectors so as to maximize the network-wide energy efficiency subject to fronthaul capacity and transmit power constraints. This paper solves such a mixed discrete-continuous, non-convex optimization problem using fractional programming and successive inner-convex approximation techniques to deal with the non-convexity of the continuous part of the problem, and $l_0$-norm approximation to account for the binary association part. A highlight of the proposed algorithm is its capability of being implemented in a distributed fashion across the multiple clouds through a reasonable amount of information exchange. The numerical simulations illustrate the pronounced role the proposed algorithm plays in improving the energy efficiency of large-scale cache-enabled MC-RANs, especially at the high interference regime.

Index Terms—Multi-cloud radio access network, energy efficiency, successive inner convex approximation, fractional programming, Dinkelbach algorithm, distributed implementation.

I. INTRODUCTION

A. Overview

Beyond the fifth generation (B5G) wireless communication networks are expected to enable ultra-connectivity through the empowerment of Internet of Things (IoT) systems [3]. IoT systems introduce unprecedented amounts of data traffic, thanks to the tremendous increase in the number of efficient mobile communication devices such as smartphones and tablets, and the extreme popularity of content-provider social media platforms such as YouTube and Netflix [3]–[5]. Video data traffic leads to an exponential increase in mobile data traffic. The video data usage is anticipated to increase from 63% of the total data traffic of 38 exabytes (EB) per month in 2019 to 76% of the total data traffic of 160 EB per month in 2025 [6]. While the data traffic exponentially increases and the requirements for modern communication systems introduce new challenges, restraining the network’s total energy consumption is vital.

In recent years, cloud radio access networks (C-RANs) have emerged as promising network architecture to accommodate the requirements of B5G wireless networks. In C-RAN, a set of geographically distributed base stations (BSs) are connected to a central processor (CP) at the cloud via high-speed digital fronthaul links, which helps managing the exacerbating large-scale wireless interference [7]. In conjunction with C-RANs development, edge caching in wireless networks promises nowadays to be an efficient technique to reduce the backhaul congestion in the network, and consequently to minimize the content delivery time during peak-traffic communication [8]. This is achieved by storing the popular content at the BSs closer to edge-users. Such caching approach can further improve the content delivery rate and reduce the communication latency via alleviating the communication load on the fronthaul links, which represents the bottleneck in achieving high data-rates in C-RANs. As such, edge caching introduces further challenges, as it impacts the network architecture, e.g., the split of processing user data at the CP or the BSs.

The majority of works on C-RAN consider a single-cloud scenario, where a single CP in the cloud is responsible for coordinating the operation of the well-spread multi-cell networks, containing a large number of BSs and users (see [9], [10] and references therein). However, the plurality and widespread of devices in next-generation systems would necessitate the deployment of multiple CPs, each responsible for managing a distinct set of BSs [11]–[14]. Each CP at the cloud coordinates the data processing and beamforming vectors of
the set of BSs associated with it. Such coordination between CPs, however, needs not to exacerbate the communication backbones and is rather limited to message passing among the different clouds; hence the need to distributively manage their underlying infrastructures on a message-passing level, which this paper tackles in details. We hereafter refer to a C-RAN with multiple CPs as a multi-cloud radio access network (MC-RAN) to distinguish it from the classical single CP C-RAN. In MC-RAN, the inter-cloud interference becomes an additional performance barrier metric, especially given the limited communication between the distributed CPs; thus the need to jointly managing both the inter-cloud and the intra-cloud interference. Even more pronounced than in single CP C-RAN, backhaul congestion is a major problem in MC-RAN, especially during high-traffic times, due to the accen-tuated number of constrained cloud-BS connections. Edge caching in MC-RANs becomes, therefore, vital for tackling the congestion of the backhaul link and, consequently, for minimizing delivery delay times. To this end, our paper addresses several challenges in the context of future multi-cloud networks, namely the fronthaul congestion, the influence of edge caching on the considered network architecture, the interference between clouds, the need for a distributed solution, and constraining the network’s energy consumption. It is the first work of its kind that proposes, and evaluates the benefit of caching in the context of multi-cloud systems. It is one step forward towards designing decentralized, energy-efficient, cloud-enabled systems assisted with wireless edge caching capabilities.

In this paper, we consider a content-based MC-RAN, where each cloud coordinates the operation of a set of BSs. Each BS is equipped with a local memory that has a certain storage capacity to cache the most popular files. The paper adopts the problem of maximizing the energy efficiency (EE) metric, which is defined as the rate-to-power ratio, so as to strike a good trade-off between the achievable rate and transmit power. In the MC-RAN model addressed in this paper, the system performance becomes a function of the user-to-cloud association and caching strategies, as well as the beamforming vector of each user. The paper tackles such a challenging non-convex mixed-integer optimization problem through an efficient algorithm that can be implemented in a distributed fashion across the multiple CPs.

B. Related Works

The MC-RAN resource management problem considered in this paper is related to recent works on wireless edge caching, C-RANs, and distributed resource allocation. The C-RAN architecture has been studied by several recent works, e.g., [15]–[17]. All these works deal with the efficient design and optimization of the resource allocation strategy in the considered C-RAN system architecture. For instance, [15] utilizes the concept of rate-splitting and common message decoding to enhance the network-wide sum-rate. Work [16] additionally enhances the C-RAN’s performance by exploiting the characteristics of reflecting surfaces. However, with most related works on C-RAN, large-scale optimization is often computationally infeasible [18], also the fronthaul conges-

tion becomes highly problematic, which necessitates utilizing caching at edge capabilities [19]. Wireless edge caching has received substantial attention in research communities recently. The seminal paper [8] highlights the benefits of caching in reducing the end-to-end transmission delay and alleviating the bottleneck of fronthaul congestion in wireless communication. In [20], the authors investigate both coded and uncoded caching strategies and analyze their impact on the EE of the system. Cooperative caching and optimizing transmission schemes jointly in small cell networks are studied in [21]. Concerned with the EE in cache-enabled networks, the work [22] considers scalable video coding-based random and fractional caching in a single user and multiple BSs setting. To reduce energy consumption, latency, and enhance the cache hits, work [23] considers an information centric network under a central control caching scheme in a network with multiple autonomous systems. In [24], the authors consider a multi-cell multi-antenna fog-RAN with multiple cache instances and a recommendation scheme, which further boosts cache hits. A connection between single CP C-RAN and edge caching is drawn in the seminal work [25]. The authors investigate the dynamic content-centric BS clustering and multicast beamforming design and formulate the problem of minimizing a weighted sum of fronthaul cost and transmit power under the quality-of-service (QoS) constraints. The authors in [26] study a coded-caching strategy in C-RAN and use semi-definite programming relaxations (SDP) to optimize the beamforming vectors from BSs to users. Other works have studied edge caching in C-RAN with different objectives. In [27], the authors study the impact of caching on balancing the outage probability against fronthaul usage in a single CP C-RAN. The paper [27] suggests a caching strategy that jointly optimizes the cell average outage probability and fronthaul usage. The paper [28] studies the joint design of cloud and edge processing, where the edge nodes, i.e., the BSs, are equipped with local caches. All these works [25]–[28], however, consider a single CP C-RAN model, which again may be computational prohibitive in ultra-dense networks. References [25]–[27], in particular, consider a conventional C-RAN model in which the CP is responsible for performing most tasks using the baseband processing protocol, while radio transmission is done by the BSs. Such functional split may be prohibitive in a distributed setup, which is often required in future networks. Therefore, reference [28], on the other hand, illustrates the necessity of the existence of baseband processing capabilities at the BSs. Such capabilities are necessary when the BSs are equipped with a local cache that enables them to send the content directly to the end-user without the need for CP interaction; thereby reducing the usage of fronthaul capacity. Promising results show that edge computing architecture has strong merits for meeting B5G system requirements, mainly, those related to enhanced mobile broadband (eMBB), massive machine-type communications (mMTC), and ultra-reliable low latency communications (URLLC) services [29]. For example, in [30], the authors propose a computational cost model, which directly links the resource blocks reserved for a certain service with the computational capacity required for performing the processing tasks. Moreover, it is shown in [30], [31] that the
required resource blocks mainly depend on the type of service requested by the users. Departing from previous works, the consideration of such computational costs in the considered cache-assisted C-RAN model is vital.

Extending the concept of C-RAN, works [18], [32] propose the embedding of cloud computing into heterogeneous networks (HetNets), where cells of different size, e.g., macrocells, microcells, are jointly employed. Such paradigm is referred to as H-CRAN. Motivated by increased throughput, coverage, and EE performance, the centralized physical layer large-scale cooperative signal processing is a promising technique for current and future networks. However, due to the distance of cloud and BSs in large-scale networks and the computational and memory-related burden at the cloud, even in moderate dense BS deployments, both works point out that such networks are unable to achieve optimized performance metrics with a reasonable computational complexity. Thereby, [32] motivates the need of introducing the multi-cloud paradigm in future networks, as further illustrated in our paper. Recently, multi-cloud systems are studied in references [11], [12], under the assumption that each CP adopts a compression-based transmission strategy. In the current article, however, we focus on the data-sharing strategy, which can achieve better performance in terms of sum-rate [33]. In the same direction, references [13], [34] consider the user-to-CP association problem in a multi-cloud setup. While reference [13] assumes fixed beamforming and an infinite fronthaul capacity, reference [34] partially overcomes this issue by assuming a discrete set of fixed resources associated with each cluster of BSs connected to a specific CP. While the works related to multi-cloud setups consider distributed solutions, it is important to incorporate the multi-cloud architecture into the ideas of cache-assisted C-RANs and develop novel distributed solutions to the highly complex problem related to such networks.

Contrary to the herein considered works on resource management, which mostly consider solving complex optimization problems numerically using sophisticated algorithms, deep learning is emerging as a promising technique to tackle such problems differently. While an in-depth analysis of deep learning for solving such complex optimization problems is out of this work’s scope, references such as [35], [36] can be useful in this future research direction.

In summary, most of the previous works focus on either a single-cloud architecture, or consider the edge caching problem or the local processing power, but not the connection of both facets. In this article, however, we consider the downlink of an MC-RAN in which the BSs are equipped with local caches and baseband processing capabilities. The performance of such a system is a function of the user-to-cloud association and the baseband functional split between the CPs and the local BSs. As the caches require processing power, additional energy consumption at the BSs has to be considered [37]. We propose a transmission scheme where the content requested by each user can be served directly from the BS, if it is stored in the cache, or can be retrieved from the CP in case local processing is not affordable. To the best of our knowledge, this is the first work that investigates the connection between edge caching and functional split in MC-RAN.

C. Contributions

Unlike the aforementioned references, this paper focuses on an MC-RAN setup and considers the problem of jointly determining the user-to-cloud association and the users’ beamforming vectors by maximizing the EE subject to exclusive local or global processing constraints, per-BS power constraints, and per-BS fronthaul constraints. To tackle such a difficult mixed discrete-continuous, non-convex optimization problem, we propose a solution that is based on fractional programming and successive inner-convex approximations (SICA) framework to determine the continuous variables, and an $l_0$-norm heuristic approximation to cope with the discrete (binary) variables. A highlight of the proposed algorithm is its ability to determine the user-to-cloud association and beamforming vectors in a distributed fashion across the multiple clouds, which makes it amenable to practical implementation. Unlike our conference version [1] which excludes the edge caching capabilities and focuses on the sum-rate maximization problem, this paper rather considers a cache-enabled MC-RAN and tackles the EE objective, which strikes a trade-off between achieving a reasonably high sum-rate for a relatively low power consumption. We herein consider a practical system model in which multiple CPs are responsible to manage a dense set of BSs, each equipped with local cache storage and baseband processing capabilities, as a means to alleviate the congestion of the fronthaul links across the multiple clouds of the network. The major contributions of this paper can be summarized as follows:

1) **Hybrid Transmission Scheme:** In the studied system model, we propose a flexible functional split between the CPs at clouds and the BSs. That is, if a BS caches the requested content, the baseband processing functions can be performed either locally at the BS, bypassing the interaction with the CP and the corresponding load on the fronthaul links, or centrally at the CP. Each functional split option determines a trade-off between the computation and fronthaul communication costs and results with different EE values. Hereby, the fronthaul congestion problem and the connection between edge caching and functional split challenges are tackled.

2) **Optimization Framework:** We formulate an EE maximization problem subject to exclusive local or global processing constraints, per-BS power constraints, and per-BS fronthaul constraints. We then develop a general solution to the formulated non-convex problem, by first solving for the user-to-cloud association, then relaxing the binary variables using $l_0$-norm approximation, and finally solving for the continuous non-convex optimization problem using Dinkelbach-transform [38], with an SICA framework. Thus, we tackle the energy consumption challenge with respect to the interference between clouds.

3) **Distributed Approach:** To best illustrate how the proposed solution is amenable for practical implementation in MC-RANs, the paper shows how the algorithm can be executed in a distributed fashion across the multiple CPs, while requiring minimal inter-cloud information exchange. Numerical simulations present that, dependent
on the system parameters, the distributed strategy closely approaches the centralized implementation in terms of achieved EE, while exhibiting superior convergence behavior.

D. Notations

Throughout the paper, boldface lower-case and capital letters (e.g. \( \mathbf{h}, \mathbf{H} \)) denote vectors and matrices, respectively. Calligraphic letters (e.g. \( \mathcal{H} \)) represent sets and a column vector consisting of all the elements in set \( \mathcal{H} \) is defined as \( \text{vec}(\mathcal{H}) \).

If \( \mathcal{H} = \{h_1, \cdots, h_N\} \), then \( \text{vec}(\mathcal{H}) \equiv [h_1, \cdots, h_N]^T \). If \( \mathcal{H} = \{h_1, \cdots, h_N\} \), then \( \text{vec}(\mathcal{H}) \equiv [h_1^T, \cdots, h_N^T]^T \). \( O_N \) is a vector of length \( N \) with all elements set to zero. The real and complex field are noted as \( \mathbb{R} \) and \( \mathbb{C} \), respectively, while the real part of complex numbers is given by \( \mathbb{R} \{\cdot\} \). Finally \( (\cdot)^H \) denotes the hermitian transpose and \( (\cdot)^T \) the transpose operator, also \( |\cdot| \) is the absolute value and \( \|\cdot\|_p \) the \( L_p \) norm.

II. SYSTEM MODEL

A. Received Signal Model

Consider the downlink of an MC-RAN consisting of \( C \) CPs, where each CP \( c \) coordinates a set of BSs of size \( B_c \). Each BS is assumed to have \( L \) antennas, connected to one (and only one) CP via a digital fronthaul link with finite capacity, and equipped with a local cache memory of a total of \( F_b \leq F \) local files, where \( F \) denotes the total number of the library files. The network consists of \( K \) single-antenna users. Fig. 1 illustrates an example of the considered model, with an MC-RAN of 3 clouds, and a total of 5 BSs and 8 users.

Let \( C = \{1, \cdots, C\} \) be the set of CPs, and \( B = \{1, \cdots, B\} \) be the set of BSs in the network, where \( B = \sum_{c \in C} B_c \). Furthermore, let \( K = \{1, \cdots, K\} \) be the set of users, and \( \mathcal{F} = \{1, \cdots, F\} \) be the set of all files. Since not every cloud stores all files from the library, let \( Q \in \{0, 1\}^{F \times C} \) be the file availability matrix. It contains the entry 1 if and only if a cloud \( c \) stores the file \( f \in \mathcal{F} \). In this context, we denote the \( f \)-th row and the \( c \)-th column as \( Q_{f,c} = q_{f,c} \). Then, \( q_{f,c} = 1 \) when \( c \) has access to the file \( f \), which is requested by user \( k \), and zero otherwise. Also, we assume that each user \( k \in K \) can be assigned to one and only one CP \( c \). For such assignment the availability of \( k \)'s requested file at \( c \), i.e., \( q_{f,c} = 1 \). Furthermore, we assume that each CP \( c \in C \) is connected to a cluster of BSs denoted by \( B_c = \{1, \cdots, B_c\} \).

The networks clusters \( B_c, c \in C \) are assumed to be disjoint, i.e., \( \cup_{c \in C} B_c = B \) and \( B_c \cap B_{c'} = \emptyset, \forall c \neq c' \). Note that we assume the BS-to-cloud association to be fixed in this work. This is especially reasonable, since we herein consider the more dynamic resource management problem. In contrast, the BS-to-cloud association is part of the network planning step, which operates on a different time-scale and is less dynamic.

Let \( \mathbf{h}_{b\cdot k} \in \mathbb{C}^{L \times 1} \) be the channel vector from the \( b \)-th BS of the \( c \)-th cloud to the \( k \)-th user, and let \( \mathbf{h}_{c\cdot k} \in \mathbb{C}^{F_b \times 1} \) be the aggregated channel vector from the \( c \)-th cloud to the \( k \)-th user. This can be expressed as \( \mathbf{h}_{c\cdot k} \equiv \text{vec}((\mathbf{h}_{c\cdot b} \forall b \in B_c)). \) To simplify our discussion and make the problem mathematically tractable, we assume that each CP has access to the full channel state information (CSI) and to the cached content of BSs in \( B \). Each CP is also aware of the demands (requested files) of all the users in the network. To deliver the requested files, we adopt a time-slotted block-based transmission model where each transmission block consists of several time slots. The channel fading coefficients remain constant within one block and may vary independently from one block to another. We focus on optimizing the EE of the cache-aided MC-RAN within one transmission block. Without loss of generality, we consider that the CPs divide each requested file into several data chunks, so that the transmission of each file may take place on several consecutive transmission blocks and the number of required transmission blocks to transmit each file may be different from other files. We note that as the requests have to be exchanged only after several blocks, the assumption of full user demand knowledge at the CPs is reasonable.

B. Cache Model

In content delivery networks, edge caching is employed to bring the content closer to users. In general, we can distinguish between two phases in content delivery process to mobile users, namely cache placement and cache delivery phases [39], [40]. Therefore, the recent works studying cache-aided wireless networks can be divided into two main categories: 1) optimizing the cached content delivery process for a given cache placement to get the best possible performance [41]; 2) improve the content delivery process through efficient design of cache placement strategies [27]. Essentially, in the cache placement phase, the popular content is stored at edge-network nodes, i.e., at BSs, with the sole purpose of improving the cache delivery phase, especially in peak-traffic times. Hence, the cache placement phase takes place over a much longer time-scale than that required in the cache delivery phase, since the popularity of the content changes much slower than the time required to deliver the requested content to the users.

In this article, we focus on the optimization of the cache delivery phase, while the cache placement phase is considered to be performed a priori, which is reasonable due to the different time-scales of cache placement and cache delivery phases. The cache content at BSs and the user’s requests are assumed to be known at the clouds. Again, each cloud only stores some part of the whole library of files, i.e., cloud \( b \) stores files \( f \in \{f \in \mathcal{F} | q_{f,b} = 1\} \). Let \( C \in \{0, 1\}^{F \times B} \) be the binary cache placement matrix where \( C_{f,b} = c \) is the element in the \( f \)-th row and the \( b \)-th column. Now, let \( f_k \in \mathcal{F} \)
be the requested file of user $k$, then $c_{f_k,b} = 1$, a cache hit, means that $f_k$ is cached at BS $b$ and $c_{f_k,b} = 0$, a cache miss, means it is not. Define the set of cache hit users as $K_1 = \{ k \in K | \exists c, b \in C \times B : c_{f_k,b} = 1 \}$. Hence, the set $K_1$ contains all users whose requested files are cached locally at the BSs. On the other hand, we define the set of cache-miss users $K_2$ as the set of users whose requested files are not cached at the BSs, i.e., $K_2 = \{ k \in K | \forall c, b \in C \times B : c_{f_k,b} = 0 \}$. Note that in the special case where $C = 0_{F \times B}$, no files are stored in BSs caches, i.e., $K_1 = \emptyset$ and $K_2 = K$. On the other hand, when $\sum_{b \in B} c_{f,b} \geq 1 \forall f \in F$, each file is cached at least at one BS, i.e., $K_2 = \emptyset$ and $K_1 = K$.

### C. Baseband Processing and Fronthaul Communication Cost

The majority of works on wireless caching consider the fronthaul communication and transmit costs, but ignore the computation cost required to process the requested file before transmitting it to the users. Unlike previous works, in this paper, we account for such factors while optimizing the delivery phase strategy such that the EE of the MC-RAN is maximized. The paper considers a computational cost model, in which the processing cost associated with each requested content is fixed and depends on the service requested by the user and the number of resource blocks served for delivering the requested content. We denote this cost as $\sum_{c,b,k} \mathbb{I}\{ \exists n \in C_s, n \neq 0 \} \lambda_{c,b,k}$. Such cost metric is justified by the presence of multiple virtual network functions, which need to be executed before the transmission of the raw data [30]. Next, we describe different transmitting strategies considered in the cache-aided MC-RAN (i.e., at CPs and BSs). Each uses the processing resources, fronthaul links, and transmit resources differently.

### III. TRANSMIT SCHEME AND FUNCTIONAL SPLIT

#### A. Design Transmit Signals at the CP

In this paper, we focus on the data-sharing transmission strategy. In this strategy, the CP performs joint encoding of users’ data. In more details, CP $c$ encodes $v_k$, the data chunk of the file $f_k$ requested by user $k$, into $s_k$. Here $s_k$ denotes the symbol of the encoded data at CP $c$ to be transmitted to user $k$ at the current time slot. We assume that $s_k$ is chosen independently from a complex Gaussian distribution with zero-mean and unit variance. After that, the CP forwards $s_k$, the encoded data chunks, through limited capacity fronthaul links to the cluster of BSs serving user $k$. The BSs then cooperate to transmit the signal to user $k$ using a joint beamforming vector. Although the beamforming vector coefficients are designed at the CP, the modulation and precoding tasks are performed at the BSs. We assume that the rate required to transmit beamforming vector coefficients over the fronthaul links is negligible compared to that required for transmitting the coded symbols of the users [9]. Let $w_{c,b,k} \in \mathbb{C}^{L \times 1}$ be the beamforming vector of BS $b$ in cluster $c$, for user $k$ designed at CP $c$. Further, let $w_{c,b,k} = \text{vec}(\{w_{c,b,k} \forall b \in B_c\}) \in \mathbb{C}^{B_c \times L \times 1}$ be the aggregate beamforming vector of user $k$ when associated with CP $c$. Note that if BS $b \in B_c$ is not in the BSs’ cluster serving user $k$, then $w_{c,b,k} = 0_{L}$, and the CP in this case does not share any data of user $k$ with BS $b$. Thus, the aggregate beamforming vector $w_{c,k}$ is a group-sparse vector by construction.

#### B. Design Transmit Signals Locally at BSs

Caching the most popular files locally at the BSs overcomes the disadvantages of processing the data at the cloud and significantly reduces the load on the fronthaul links. Hence, the usage of the fronthaul link boils down to the exchange of essential control information between CPs and BSs (e.g., beamforming coefficients and scheduling information). However, despite the advantages of caching the content locally at BSs in terms of reducing latency and saving the fronthaul link bandwidth, this comes at the cost of increasing the processing cost at the BSs. Hence, we consider that the BSs cache the uncoded data locally, and so encoding the data before transmission is done at the local processing unit at the BSs. That is, we assume the baseband processing tasks can be done at the BSs serving user $k$, whenever the BSs cache the required file of user $k$. To this end, let $\tilde{w}_{c,b,k} \in \mathbb{C}^{L \times 1}$ be the beamforming vector explicitly used at BS $b \in B_c$ for user $k$ when the baseband processing tasks are performed at the BS. Let $w_{c,b,k} = \text{vec}(\{\tilde{w}_{c,b,k} \forall b \in B_c\}) \in \mathbb{C}^{B_c \times L \times 1}$ be the aggregate beamforming vector at BSs in cluster $B_c$ which cache the requested file of user $k$. Note that the BS $b$ can encode the data locally and independently of the CP connected to it when it caches the requested file from user $k$. The control information needed for transmitting the signal is, however, assumed to be provided from the cloud. Hence, partial cooperation between CPs on the control level becomes possible.

#### C. Hybrid Transmit Strategy

This paper considers a hybrid transmit strategy, where each BS $b$ can serve a user $k$ (for each user $k \in K$) according to one of three possibilities: 1) The BS can participate in transmitting data to $k$ following the CP processing strategy, i.e., based on the CP encoding, 2) Or, the BS processes the data locally, i.e., in caching scenarios. 3) Or, the BS does not transmit to user $k$. The transmit signal at BS $b$ from cluster $B_c$, $x_{c,b} \in \mathbb{C}^{L \times 1}$, can thus be written as follows:

$$
x_{c,b} = \sum_{k \in K} (w_{c,b,k} + \tilde{w}_{c,b,k}) s_k.
$$

The encoding process can be either done at the cloud or locally at the BS, but not at both at the same time. Also, the BS can perform the processing locally only in case it caches the requested file. Therefore, equation (1) is accompanied by the following two conditions:

$$
\begin{align*}
&\mathbb{I}\{ ||w_{c,b,k}||_2^2 \geq 1, \forall k \in K, \forall b \in B_c\}, \\
&\mathbb{I}\{ \tilde{w}_{c,b,k} = 0_L, \forall k \in K_2, \forall b \in B_c, \}
\end{align*}
$$

where $\mathbb{I}\{ \cdot \}$ is the indicator function defined as: $\mathbb{I}\{ x \} = 1$ if $x > 0$, and 0 otherwise. Equations (1), (2), and (3) can be interpreted as follows: If file $f_k$ is not cached at BS $b$ then $b \in B_c$ transmits to user $k$ only if $\mathbb{I}\{ ||w_{c,b,k}||_2^2 \geq 1, \forall k \in K\}$, since in this case $\tilde{w}_{c,b,k} = 0_L$. In case file $f_k$ is cached at BS $b$, the data would be encoded at the CP or locally at the...
BS depending on equation (2). Moreover, by construction, if user $k$ is not associated with CP $c$ then $\mathbf{w}_{c,b,k} = 0, \forall b \in \mathcal{B}_c$. The specific design of beamforming vectors $\mathbf{w}_{c,b,k}$ or $\tilde{\mathbf{w}}_{c,b,k}$ in this case is based on solving our optimization problem, as discussed in details in section IV. After forming the transmit signal as in (1), BS $b$ sends $x_{c,b}$ subject to the following maximum transmit power constraint:

$$\mathbb{E}\left\{\left|\mathbf{x}_{c,b}\right|^2\right\} \leq P_{b}\text{Max}. \quad (4)$$

### D. Achievable Rates and Fronthaul Constraints

User $k$ can be served by one (and only one) BS-cluster connected to CP $c$ with an aggregate beamforming vector $\mathbf{w}_{c,k}$. Define the user-to-cloud association as a binary variable $z_{c,k}$, i.e., $z_{c,k} = 1$ if user $k$ is associated to cloud $c$, and 0 otherwise. We further assume that each user can be associated to one and only one CP since, otherwise, a signal-level coordination would be required between the clouds, rather than a controller-level coordination. We can now write the signal to interference plus noise ratio (SINR) of user $k$ when associated with CP $c$ as follows

$$\text{SINR}_{c,k} = \frac{\sum_{(c',k') \neq (c,k)} \|h_{c',k'}^{\mathrm{h}} (\mathbf{w}_{c',k'} + \tilde{\mathbf{w}}_{c',k'})\|^2}{\|\tilde{\mathbf{c}}_{c,k} + \tilde{\mathbf{w}}_{c,k}\|^2 + \sigma^2}, \quad (5)$$

where $\sigma^2$ is the noise power. Let $\tau$ be the bandwidth allocation of user $k$. The achievable rate of user $k$ associated to cloud $c$ becomes bounded as

$$r_{c,k} \leq \tau \log_2(1 + \text{SINR}_{c,k}). \quad (6)$$

The transmit power per-BS can be expressed as

$$P_b(\mathbf{w}, \tilde{\mathbf{w}}) = \frac{1}{\eta_b} \sum_{k \in \mathcal{K}} \left(\|\mathbf{w}_{c,b,k}\|^2 + \|\tilde{\mathbf{w}}_{c,b,k}\|^2\right), \quad (7)$$

where $\eta_b < 1$ is the efficiency of the transmit amplifier at BS $b$. The required fronthaul capacity at BS $b$ is given as

$$C_b(\mathbf{w}, \tilde{\mathbf{w}}) = \sum_{k \in \mathcal{K}} \left(\|\mathbf{w}_{c,b,k}\|^2 + (1 - c_{f_b,k}) \|\tilde{\mathbf{w}}_{c,k}\|^2\right) r_{c,k}, \quad (8)$$

where $\mathbf{w} \triangleq \text{vec}\{(\mathbf{w}_{c,k})_{c \in \mathcal{C} \times \mathcal{K}}\}$, $\tilde{\mathbf{w}} \triangleq \text{vec}\{(\tilde{\mathbf{w}}_{c,k})_{c \in \mathcal{C} \times \mathcal{K}}\}$. It is obvious from (8) that if BS $b$ caches file $f_k$ requested by user $k$, i.e., $c_{f_b,k} = 1$, then user $k$ does not add to the burden of the fronthaul link of BS $b$ when $\mathbf{w}_{c,b,k} = 0$. $L_b$

### E. Energy Efficiency at the Cloud

In the context of our paper, the individual EE metric of each cloud $c$ is defined as the sum-rate of all users associated with $c$ divided by the power consumption required to serve these users. This work takes into account the transmit power, processing power, fronthaul power consumption, and operational fixed power consumption. The latter, i.e., the operational fixed power, does not depend on the number of users and rather accounts for the required cooling and circuitry power resources for the functionality of the C-RAN. Mathematically, we define the EE at the cloud $c$ as follows

$$f_{\text{EE}}(c) \triangleq \frac{\sum_{k \in \mathcal{K}} r_{c,k}}{P_{\text{TX}}^c + g_{\text{EE}}(c) + P_{\text{CP}}^c}, \quad (9)$$

where $P_{\text{TX}}^c$ is the total transmit power consumed by the BSs of cluster $\mathcal{B}_c$; defined as

$$P_{\text{TX}}^c = \sum_{b \in \mathcal{B}_c} P_b(\mathbf{w}, \tilde{\mathbf{w}}), \quad (10)$$

where the processing power of the fronthaul, the BS, and the CP can be written as

$$g_{\text{EE}}(c) = \sum_{(b,k) \in \mathcal{B}_c \times \mathcal{K}} \mathbb{I}\left\{\|\mathbf{w}_{c,b,k}\|^2\right\} P_{\text{fhl}}^b + \sum_{(b,k) \in \mathcal{B}_c \times \mathcal{K}} \mathbb{I}\left\{\|\tilde{\mathbf{w}}_{c,b,k}\|^2\right\} P_{\text{proc}}^b, \quad (11)$$

and where $P_{\text{fhl}}^b$ is the operational fixed power consumption. Note that the fronthaul processing power is consumed by the clouds. Interestingly, (9) captures the trade-off between the local processing of cached files at the BSs and the fronthaul usage when the files are processed at the CP. The next section addresses the paper’s main optimization problem, which aims at maximizing the sum of EE of all the clouds, so as to determine the user-to-cloud assignment, the processing power decision variables, user-to-BS association, and the joint transmit beamforming vectors for all users across the network.

### IV. MC-RANs EE Maximization and Algorithms

This section first formulates the EE maximization problem as a mixed-integer optimization problem. To best tackle the intricacies of the problem at hand, the paper then presents some well-chosen mathematical reformulations, so as to derive an efficient iterative algorithm, the highlight of which is that it can be implemented in a distributed fashion across the network CPs.

#### A. General Problem

In the context of distributed EE across the MC-RAN, we seek to jointly optimize the functional split mode for each BS, the beamforming vectors and user-to-cloud association of all users in the network subject to per BS maximum transmit power and maximum fronthaul capacity constraints. The optimization problem under consideration can be mathematically written as:

$$\begin{align*}
\text{maximize} & \quad w_{c,k}, z_{c,k} \sum_{c \in \mathcal{C}} f_{\text{EE}}(c) \\
\text{subject to} & \quad (2), (3), \\
& \quad P_b(\mathbf{w}, \tilde{\mathbf{w}}) \leq P_{b}\text{Max} \quad \forall b \in \mathcal{B}_c, \forall c \in \mathcal{C}, \quad (12a) \\
& \quad C_b(\mathbf{w}, \tilde{\mathbf{w}}) \leq F_{b,c} \quad \forall b \in \mathcal{B}_c, \forall c \in \mathcal{C}, \quad (12b) \\
& \quad \text{SINR}_{c,k} \geq 2^{r_{c,k}/7} - 1 \quad \forall k \in \mathcal{K}, \forall c \in \mathcal{C}, \quad (12c) \\
& \quad \sum_{c \in \mathcal{C}} z_{c,k} = 1 \quad \forall k \in \mathcal{K}, \quad (12d) \\
& \quad z_{c,k} \in \{0, 1\} \quad \forall k \in \mathcal{K}, \forall c \in \mathcal{C}, \quad (12e) \\
& \quad \|\mathbf{w}_{c,b,k}\|^2 \leq M z_{c,k} \quad \forall k \in \mathcal{K}, \forall b \in \mathcal{B}_c, \forall c \in \mathcal{C}, \quad (12f) \\
& \quad \|\tilde{\mathbf{w}}_{c,b,k}\|^2 \leq M z_{c,k} \quad \forall k \in \mathcal{K}, \forall b \in \mathcal{B}_c, \forall c \in \mathcal{C}, \quad (12g) \\
& \quad \sum_{k \in \mathcal{K}} z_{c,k} \leq K_{c}\text{Max} \quad \forall c \in \mathcal{C}, \quad (12h) \\
& \quad z_{c,k} = 0 \quad \forall k \in \{i \in \mathcal{K} | q_{f_b,c} = 0\}, \forall c \in \mathcal{C}, \quad (12i)
\end{align*}$$

where $z = \text{vec}\{(z_{c,k})_{c \in \mathcal{C} \times \mathcal{K}}\}$, $r = \text{vec}\{(r_{c,k})_{c \in \mathcal{C} \times \mathcal{K}}\}$, and $M$ is a large positive integer $M \in \mathbb{R}_{++}$. This is related to the big-$M$ constraint [42]. $P_{b}\text{Max}$ and $F_{b,c}$ are the maximum transmit power and the fronthaul capacity of BS $b$ in cloud $c$, respectively. $K_{c}\text{Max}$ is the maximum number of users that can connect to cloud...
c. Constraint (12b) represents the maximum transmit power available to BS $b$, and constraint (12c) represents the available fronthaul capacity of BS $b$ connected to CP $c$. Constraint (12d) gives an upper bound on the maximum achievable rate of user $k$ when assigned to cloud $c$. Constraints (12e)-(12f) assure that each user can be associated with one and only one CP, note that assigning users to more than one cloud is out of this work’s scope. Constraints (12g)-(12h) represent the big-$M$ constraints and can be read as follows: if the CP $c$ is associated with user $k$, then constraints (12g)+(12h) are deactivated [42]. Otherwise, if $k$ is not associated with $c$, constraint (12g) forces the corresponding beamforming coefficients in $w_{c,b,k}$ to zero, also (12h) forces $\tilde{w}_{c,b,k}$ to zero. The number of associated users to cloud $c$ does not exceed a given maximum number of users, which is ensured by (12i).

At last, (12) restricts the user-to-cloud association in such a way that users may not be associated to clouds that do not store their requested contents, i.e., $z_{c,k} = 0$ if $q_{f,c} = 0$.

The above optimization (12) is over the binary variables $z_c$, the continuous beamforming vectors $w$ and $\tilde{w}$, and the rates $r$. Problem (12) is challenging to solve due to the non-convexity of the objective function and constraints (12c)-(12f), besides the discrete nature of variables $z_c$.

B. Overall Algorithmic Framework

The optimization of the association variables and beamforming vectors in (12) is hard to tackle jointly and may be computationally prohibitive to solve globally. Therefore, our paper proposes adopting a two-step optimization approach. In the first step, we adopt an auxiliary, local utility function that represents the benefit of associating a user $k$ to cloud $c$. Then we formulate a generalized assignment problem to find the user-to-cloud association $z_c$ for that given utility function. Afterwards, in the second stage, given the user-to-cloud assignment solution, we solve the optimization problem (12) using a $l_0$-norm relaxation followed by a successive inner-convex approximation approach to find the beamforming vectors $w$ and $\tilde{w}$, and the rates $r$. We next present the generalized assignment formulation to solve the user-to-cloud association problem.

C. User-to-Cloud Association

Since problem (12) is too complicated to solve in full generality, we first propose a heuristic solution to find a feasible user-to-cloud association solution, which serves as a network planning step. More precisely, let $U(c,k)$ be an auxiliary, local utility function that measures the benefit of associating user $k$ with cloud $c$. A reasonable choice of $U(c,k)$ is the following EE-like function:

$$U(c,k) = \frac{r_{c,k}}{\sum_{b \in B} \frac{1}{\eta_b} \left( \| w_{c,b,k} \|^2 + \| \tilde{w}_{c,b,k} \|^2 \right) + \gamma_{c,k} P_{f,c,k}}.$$ (13)

The intuition behind such choice is two-fold. Firstly, the utility function in (13) defines the benefit of associating user $k$ with cloud $c$ as the ratio of the achievable rate for such an association and the processing and transmit power costs. Such a utility helps mimicking a reasonable, local EE expression at each cloud $c$, given that the utility function (13) depends mainly on the aggregate beamforming vector from cloud $c$ to user $k$. Secondly, such choice helps formulating a generalized assigned problem, which allows us to derive efficient algorithms to find the association variables $z_c$; thereby alleviating the complexity of the solution of the complex problem (12). The simulation results of the paper further validate the numerical gain of such heuristic approach. At this step, we fix the beamforming vectors from cloud $c$ to user $k$, e.g., $w_{c,k}$ can have maximum ratio combining (MRC) structures defined as: $w_{c,k} \in \mathbb{C}^{h_{k,c} \times 1} = \frac{h_{k,c}}{\| h_{k,c} \|}$, $\forall k \in K$. Given the above utility $U(c,k)$ and the fixed beamforming strategy, problem (12) boils down to a user-to-cloud association problem, which can be written as:

$$\text{maximize} \quad \sum_{c \in C} \sum_{k \in K} z_{c,k} q_{f,c} U(c,k)$$

subject to

$$\sum_{c \in C} z_{c,k} \leq K_{\text{Max}}^c \quad \forall c \in C,$$ (14b)

$$z_{c,k} \in \{0, 1\} \quad \forall k \in K, \forall c \in C,$$ (14d)

$$z_{c,k} = 0 \quad \forall k \in \{ i \in K | q_{f,c} = 0 \}, \forall c \in C.$$ (14e)

Problem (14) follows a generalized assignment problem form [43], and is carried over the binary variables $z_c$. Note that constraint (12c) is incorporated in (14a) directly. When file $f_k$ requested by user $k$, is unavailable at cloud $c$, $q_{f,c}$ is set to zero, and thus there is no benefit from associating $k$ with $c$. This implicitly ensures (14e). While using global centralized optimization methods (e.g., the branch and cut algorithm) is possible, solving (14a) in a distributed manner is rather adopted in the context of our MC-RAN setup, which is done using an auction-based iterative algorithm, similar to [13] and [43], where only reasonable information exchange between the clouds is required. The algorithm guarantees convergence with a finite amount of iterations to a solution which is within a gap of $(1+\chi)$ to the global optimal solution of (14) [43, Theorems 1 and 2], where $\chi \in [1, +\infty)$ is the approximation ratio of a subroutine knapsack algorithm.

D. Problem Reformulation

Fixing the user-to-cloud association solution $z_c$ found above, problem (12) can now be reformulated as the following joint beamforming optimization problem:

$$\text{maximize} \quad \sum_{w,\tilde{w},r} \sum_{c \in C} f_{\text{EE}}(c)$$

subject to

$$r_{c,k} \leq \log_2 (1 + \gamma_{c,k}) \quad \forall k \in K, \forall c \in C,$$ (16b)

$$\sigma^2 + \sum_{(c',k') \neq (c,k)} \frac{\| h_{c',k'} \|^2}{\gamma_{c,k}} \leq 0 \quad \forall k \in K, \forall c \in C.$$ (16c)
where the introduction of the variables \( \gamma = \text{vec}\{\gamma_{c,k} \| \psi(c,k) \in C \times K\} \) to reformulate the maximum achievable rate constraint \((12d)\) into constraints \((16b)-(16c)\). Constraint \((16c)\) is now in the form of difference of convex (DC) functions which can be tackled using an efficient SICA approach. Please note that the inter-cloud and intra-cloud interference terms (middle terms) in \((16c)\) do not contain the user-to-cloud association \(z_{c,k}^r\), anymore. Ensured by the big-\(M\) constraints \((12g)-(12h)\), the beamforming vectors \(w_{c,k}^r\) and \(\tilde{w}_{c,k}^r\) are forced to zero if a user \(k^t\) is not associated with cloud \(c^t\). That is, the beamforming vectors implicitly include the (now fixed) user-to-cloud association variables, which explains the current form of constraint \((16c)\).

Moreover, \(1\{\|\tilde{w}_{c,b,k}\|_2^2\} \) indicates if the data of user \(k\) is processed locally at BS \(b\), \(1\{\|w_{c,b,k}\|_2^2\} \) denotes if CP \(c\) processes the data of user \(k\), and \(1\{\|w_{c,b,k}\|_2^2\} \) infers whether BS \(b\) is in the serving cluster of user \(k\) or not. These indicator functions present additional hurdles within the framework of the challenging problem \((16)\). We note that the benefit of using indicator functions is to determine the decision variables exclusively based on beamforming vectors. To deal with the challenging non-convex discrete indicator functions, we next use the \(l_0\)-norm relaxation technique.

First, we note that the indicator function in the objective \((16a)\) and the fronthaul constraint \((12c)\) can be equivalently expressed as an \(l_0\)-norm of the beamforming vectors. We can write \(1\{\|w_{c,b,k}\|_2^2\} \leq 1\) \(\|w_{c,b,k}\|_2^2\), \(1\{\|w_{c,b,k}\|_2^2\} \leq 1\) \(\|w_{c,b,k}\|_2^2\), and \(1\{\|w_{c,b,k}\|_2^2\} \leq 1\) \(\|w_{c,b,k}\|_2^2\). This equivalence is of importance since the \(l_0\)-norm function can be approximated with a weighted \(l_1\)-norm convex function \([9]\).

To enable the use of such approximation in the context of our paper, we write the function \(\|w_{c,b,k}\|_2^0\) as a reweighed \(l_1\)-norm as follows:

\[
\|w_{c,b,k}\|_2^0 = \beta_{c,b,k} \|w_{c,b,k}\|_2^2, \quad (17)
\]

where \(\beta_{c,b,k}\) is a constant weight associated with BS \(b\) in \(B_c\) and user \(k\) and is defined in this work as

\[
\beta_{c,b,k} = \frac{1}{\delta + \|w_{c,b,k}\|_2^2}, \quad (18)
\]

where \(\delta > 0\) is a regularization constant. In a similar manner, we define \(\tilde{\beta}_{c,b,k}\) and \(\tilde{\beta}_{c,k}\). When BS \(b\) assigns low transmit power to user \(k\), \(\beta_{c,b,k}\) increases and thus user \(k\) adds a non-desired burden to the fronthaul link and the energy consumption. The algorithm would then likely exclude \(k\) from being served by \(b\), thereby ensuring that the network only activates links with reasonable transmit powers. Since the \(l_1\)-norm is applied to a quadratic function of the beamforming vectors, the resulting approximation is a smooth continuous function which is easier to optimize as compared to a non-smooth \(l_0\)-norm.

The reformulated objective of \((16)\) now reads as

\[
f_{2,\text{EE}}(c) \triangleq \frac{\sum_{k \in C} r_{c,b,k}}{\rho_{\text{tx}} + p_{2,\text{EE}}(c)} + p_{\text{c}}, \quad (19)
\]

where

\[
p_{2,\text{EE}}(c) = \sum_{(b,k) \in B_c \times K} \beta_{c,b,k} \|w_{c,b,k}\|_2^2 r_{\text{fthl}}^b
\]

+ \(\sum_{(b,k) \in B_c \times K} \tilde{\beta}_{c,b,k} \|w_{c,b,k}\|_2^2 r_{\text{proc}}^b + \sum_{k \in K} \beta_{c,k} \|w_{c,k}\|_2^2 r_{\text{proc}}^c.
\]

Note that the function \(p_{2,\text{EE}}(c)\) is a \(l_0\)-norm relaxed formulation of \((11)\). The reformulated problem \((16)\) can now be written as

\[
\begin{align*}
\text{maximize} & \quad \sum_{c,b,k} f_{2,\text{EE}}(c) \\
\text{subject to} & \quad (3), (12b), (12c), (12g), (12h), (16b), (16c), \\
\beta_{c,b,k} \|w_{c,b,k}\|_2^2 + \tilde{\beta}_{c,b,k} \|w_{c,b,k}\|_2^2 \leq 1 \\
& \quad \forall k \in K, \forall b \in B_c, \forall c \in C, \quad (21b) \\
C_{2,b}(w) & \leq F_{c,b} \quad \forall b \in B_c, \forall c \in C. \quad (21c)
\end{align*}
\]

Note that constraints \((2)\) and \((12c)\) are also replaced in \((21b)\) and \((21c)\), respectively. Before reformulating the fronthaul capacity constraint \((12c)\), we first elaborate on the second term in \((8)\), namely \(1 - c_{f,k}, b\) \(\{\|w_{c,b,k}\|_2^2\}\). If a BS \(b\) caches file \(f_k\) then \(c_{f,k,b} = 1\), which means the fronthaul link of BS \(b\) is not affected by user \(k\). Otherwise, if the BS \(b\) does not cache the requested file by user \(k\), i.e., \(k\) is a cache-miss user from BS \(b\) perspective, we know from previous definitions and \((3)\) that \(w_{c,b,k} = 0\). Based on these observations, we can conclude that the second term in \((8)\) does not influence the fronthaul link and can thus be ignored. The reformulated fronthaul term is now

\[
C_{2,b}(w) = \sum_{k \in K} \beta_{c,b,k} \|w_{c,b,k}\|_2^2 r_{c,k}. \quad (22)
\]

The above reformulations help overcoming the discrete nature of the original problem \((16)\). However, the reformulated problem \((21)\) remains difficult, non-convex, and so it is tackled next using fractional programming and SICA.

E. Fractional Programming and Successive Inner-Convex Approximations

Note that the objective function in \((21a)\) is a sum of ratios of linear and convex functions, which makes \((21)\) a suitable platform to apply a Dinkelbach-like algorithm \([38]\). Observe, however, that the non-convex feasible set of problem \((21)\), stemming from constraints \((21c)\), \((16b)\) and \((16c)\), would hinder a direct application of a Dinkelbach-like algorithm to solve \((21)\), as this would require solving a non-convex problem to obtain a stationary solution, which is computationally prohibitive, especially when the network becomes reasonably sized \([44]\). To overcome such difficulty, this paper uses a SICA approach so as to enable an efficient implementation of a Dinkelbach-like algorithm. A highlight of our proposed algorithm is that it can be implemented in a distributed fashion across the multiple CPs. We start by reformulating problem \((21)\) to get a formulation that is amenable to apply SICA techniques.

\end{document}
F. Convexification of Problem (21)

First, we tackle (21c) by introducing the slack variables: \( t = \text{vec}(\{t_{k,b}\forall (k,b) \in \mathcal{K} \times \mathcal{B}\}) \), \( \bar{t} = \text{vec}(\{\bar{t}_{k,b}\forall (k,b) \in \mathcal{K} \times \mathcal{B}\}) \), and \( u = \text{vec}(\{u_{c,k}\forall (k,b) \in \mathcal{K} \times \mathcal{B}\}) \). Then, for all \( k \), \( b \), and \( c \), define the following auxiliary constraints
\[
\begin{align*}
\beta_{b,k} \|w_{c,b,k}\|^2_2 & \leq t_{k,b}, \\
\tilde{\beta}_{b,k} \|w_{c,b,k}\|^2_2 & \leq \bar{t}_{k,b}, \\
\sum_{k \in \mathcal{K}} \beta_{b,k} \|w_{c,b,k}\|^2_2 & \leq u_{c,k}.
\end{align*}
\]
These slack variables \( t \) as follows
\[
\sum_{k \in \mathcal{K}} t_{k,b} r_{c,k} \leq F_{b,c}, \quad \forall b \in \mathcal{B}, \forall c \in \mathcal{C}.
\]
This function is non-convex as it is bilinear in the optimization variables. However, using some algebraic transformations of \( \sum_{k \in \mathcal{K}} t_{k,b} r_{c,k} \), the above constraint can be equivalently written as
\[
\sum_{k \in \mathcal{K}} \frac{1}{4} \left( (t_{k,b} + r_{c,k})^2 - (t_{k,b} - r_{c,k})^2 \right) \leq F_{b,c}.
\]
The left-hand side of (27) is a difference of convex functions, which allows for applying SICA methods. The idea here is to find a convex surrogate upper-bound to the non-convex function associated with (27), i.e., \( \sum_{k \in \mathcal{K}} t_{k,b} r_{c,k} \). This can be done by keeping the convex part intact, and linearizing the concave part using first-order Taylor expansion. Define \( g_1(t, r, t', r') \) as follows:
\[
\begin{align*}
g_1(t, r, t', r') & \triangleq \sum_{k \in \mathcal{K}} \left( (t_{k,b} + r_{c,k})^2 - (t_{k,b} - r_{c,k})^2 \right) \leq 4F_{b,c}.
\end{align*}
\]
Here \( t' = \text{vec}(\{t'_{k,b}\forall (k,b) \in \mathcal{K} \times \mathcal{B}\}) \) and \( r' = \text{vec}(\{r'_{c,k}\forall (k,b) \in \mathcal{C} \times \mathcal{K}\}) \) are feasible fixed values, which satisfy the pre-defined constraints (23) and (26). Such feasible fixed values would eventually be updated iteratively, so as to refine the feasible set at every iteration of the SICA.

**Lemma 1.** For all feasible values \((t', r')\) and all \((c, k) \in (\mathcal{C}, \mathcal{B})\), the function \( g_1(t, r, t', r') \) satisfies
\[
g_1(t, r, t', r') \geq \sum_{k \in \mathcal{K}} t_{k,b} r_{c,k} - F_{b,c}.
\]
**Proof.** The proof of the above lemma can be found in Appendix A of the extended version of the paper available on archive [2].

The above step allows us to convexify the fronthaul capacity constraint (21c), by means of finding the convex surrogate upper-bound function \( g_1(t, r, t', r') \). We next apply a similar procedure to the non-convex constraint (16b). To this end, based on (16b), we define \( \hat{g}_2(\gamma, r) \) as
\[
\hat{g}_2(\gamma, r) = r_{c,k} - \tau \log_2 (1 + \gamma_{c,k}) \leq 0.
\]
The function \( \hat{g}_2(\gamma, r) \) in (30) is non-convex in \( \gamma_{c,k} \). One can, however, linearize the non-convex part of \( \hat{g}_2(\gamma, r) \), namely
\[
\log_2 (1 + \gamma_{c,k}) \text{, around } \gamma \text{ using the first-order Taylor expansion. The convex upper-bound of } \hat{g}_2(\gamma, r), \text{ denoted by } g_2(\gamma, r, \gamma') \text{ can be then written as}
\]
\[
g_2(\gamma, r, \gamma') \triangleq \frac{r_{c,k}}{\tau} - \log_2 (1 + \gamma'_{c,k}) - \left( \ln(2) (1 + \gamma'_{c,k}) \right)^{-1} (\gamma_{c,k} - \gamma'_{c,k}),
\]
where the variables \( \gamma' = \text{vec}(\{\gamma'_{c,k}\forall (k,b) \in \mathcal{C} \times \mathcal{K}\}) \) are feasible fixed values, which allows to convexify (16b).

Consider now constraint (16c), which can be re-written as
\[
\omega^+(\bar{w}, \bar{w}) - \omega^-(\bar{w}, \bar{w}, \gamma) \leq 0,
\]
where
\[
\begin{align*}
\omega^+(\bar{w}, \bar{w}) & = \sigma^2 + \sum_{(c', k') \neq (c, k)} |h_{c', k'}^t (w_{c', k'} + \bar{w}_{c', k'})|^2, \\
\omega^-(\bar{w}, \bar{w}, \gamma) & = |h_{c, k}^t (w_{c, k} + \bar{w}_{c, k})|^2.
\end{align*}
\]
The formulation (32) is in DC form, since \( \omega^+ \) is a convex, quadratic function in \( w \) and \( \bar{w} \). \( \omega^- \) is also convex since it is a rational function with quadratic numerator and positive linear denominator [46]. Lemma 2 below states a viable first-order approximation of such function \( \omega^- \).

**Lemma 2.** Define \( \omega(\bar{w}, \omega(\bar{w}, \bar{w}), \omega', \omega', \omega') \), where \( \bar{w} \in \mathcal{C} \times \mathcal{B} \) and \( \omega > 0 \), as
\[
\omega(\bar{w}, \omega(\bar{w}, \bar{w}), \omega', \omega', \omega') = \frac{2\tilde{\omega}(\{\bar{w}'\})^t \bar{w}}{\tilde{\omega}(\{\bar{w}'\})} - \frac{\xi}{\tilde{\omega}(\{\bar{w}'\})} |\bar{w}'|^2.
\]
**Proof.** The proof of the above lemma can be found in Appendix B of the extended version of the paper available on archive [2].

In order to obtain a convex reformulation of (32), we use the first-order-approximation of \( \omega^- \) around the feasible point \((\bar{w}', \bar{w}', \gamma')\) according to Lemma 2 to get:
\[
|h_{c, k}^t (w_{c, k} + \bar{w}_{c, k})|^2 \approx \frac{2\tilde{\omega}(\{\bar{w}'\})^t h_{c, k} \bar{w}_{c, k}}{\tilde{\omega}(\{\bar{w}'\})} - \frac{\xi}{\tilde{\omega}(\{\bar{w}'\})} |\bar{w}'|^2.
\]
By substituting the linearized form above into the non-convex formulation (32), the inner convex approximation \( g_3(w, \bar{w}, \omega, \omega', \omega', \omega') \) of \( \omega^+(\bar{w}, \bar{w}) - \omega^- (\bar{w}, \bar{w}, \gamma) \) can be written as (37) on the top of the next page.
\[ g_3(w_c, \tilde{w}, \gamma, w'_c, \tilde{w}', \gamma') :\Delta = \sigma^2 + \sum_{(c', k') \neq (c, k)} \left| h_{c,k'}^T (w_{c',k'} + \tilde{w}_{c',k'}) \right|^2 + \sum_{c \in C} \left[ \frac{2}{\gamma_{c,k}} \mathbb{R} \{ (w_{c,k} + \tilde{w}_{c,k})^T h_{c,k} (w_{c,k} + \tilde{w}_{c,k}) \} + \frac{\gamma_{c,k}}{\gamma_{c,k}^2} \left| h_{c,k}^T (w_{c,k} + \tilde{w}_{c,k}) \right|^2 \right]. \]

\[ f_{3,EE}(c) :\Delta = \sum_{(b, k) \in B_c \times K} t_{b,k} P_{b}^{\text{thl}} + \sum_{(b, k) \in B_c \times K} \tilde{t}_{b,k} P_{k}^{\text{proc}} + \sum_{k \in K} u_{c,k} P_{k}^{\text{proc}} + P_{T_{x,c}} + P_{P_{c}}. \]  

To solve problem (39), we distinguish between an outer and an inner loop. In the outer loop, we update the feasible fixed values for SICA, initialize \( \lambda_0 \) so it would be used in the inner loop, and check for convergence. In the inner loop, we use the Dinkelbach-like algorithm, and solve \( F(\lambda_j) \) iteratively using (42)-(43). Such approach produces a solution to the underlying fractional program (39) with output values \( \tilde{y}_v \), where \( \nu \) is the iteration index of the outer loop. Then, update the values of \( y' \) using the calculated values \( \tilde{y}_v \) as fixed values for the next iteration. The algorithm stops at convergence.

Combining the above confectionation steps of all constraints of the original problem (21) gives the following reformulated problem:

\[
\text{maximize } \sum_{c \in C} f_{3,EE}(c) \tag{39a}
\]

subject to

\[
g_1(t, r, t', r') \leq 0 \quad \forall b \in B_c, \forall c \in C, \tag{39b}
\]

\[
g_2(\gamma, r', \gamma') \leq 0 \quad \forall k \in K, \forall c \in C, \tag{39c}
\]

\[
g_3(w, \tilde{w}, \gamma, w', \tilde{w}', \gamma') \leq 0 \quad \forall k \in K, \forall c \in C, \tag{39d}
\]

where we optimize \( y = [w^T, \tilde{w}^T, t^T, \tilde{t}^T, u^T, \tilde{u}^T, \gamma^T, r^T]^T \), which contains all optimization variables, and where \( y' = [w'^T, \tilde{w}'^T, t'^T, \tilde{t}'^T, u'^T, \tilde{u}'^T, \gamma'^T, r'^T]^T \) is a vector containing all fixed values, where \( Y \) is the convex feasible set defined by the constraints (3), (12b), (12g)-(12h), (21b), (23), (24), (25).

\[ g_4(c) \triangleq \sum_{k \in K} r_{c,k}, \tag{40} \]

and

\[ g_5(c) \triangleq \sum_{(b, k) \in B_c \times K} t_{b,k} P_{b}^{\text{thl}} + \sum_{(b, k) \in B_c \times K} \tilde{t}_{b,k} P_{k}^{\text{proc}} + \sum_{k \in K} u_{c,k} P_{k}^{\text{proc}} + P_{T_{x,c}} + P_{P_{c}}. \tag{41} \]

We then iteratively search for a unique solution to the following auxiliary convex optimization problem:

\[ F(\lambda_j) = \max_{y \in Y} \left\{ \sum_{c \in C} g_4(c) - \lambda_j(c) g_5(c) \right\}, \tag{42} \]

where \( \lambda_j = \text{vec} \{ \lambda_j(c) | \forall c \in C \} \) is a constant vector that is updated after each iteration as follows:

\[ \lambda_{j+1}(c) = \frac{g_4(c)}{g_5(c)}, \quad \forall c \in C. \tag{43} \]

Although the proposed solution does not guarantee the global optimality of the original complicated mixed-integer non-convex optimization problem (12), our numerical simulations illustrate the appreciable performance improvement of the proposed algorithm as compared to state-of-the-art solutions. The numerical results further highlight the fast convergence of Algorithm 1, as illustrated in Sec. V-F.

**G. Iterative Algorithm**

Despite the non-convexity of the fractional function \( f_{3,EE}(c) \) in (38), all constraints of problem (39) are convex, and so (39) can be iteratively solved using a SICA and Dinkelbach-like algorithm. More precisely, in order to apply a Dinkelbach-like algorithm, we define \( g_4(c) \) and \( g_5(c) \) as the numerator and denominator of \( f_{3,EE}(c) \), respectively:

![Algorithm 1 Combined SICA and Dinkelbach-like algorithm.](Image)
user-to-cloud association problem (14) can be done on a per-cloud basis using an iterative auction algorithm [13]. Secondly, given the set of users \( \mathcal{K}_c \) served by CP \( c \), i.e., \( \mathcal{K}_c = \{ k \in \mathcal{K} \mid z_{c,k} = 1 \} \), we define local beamforming vectors associated with cloud \( c \) as \( \mathbf{w}_c = \text{vec}(\{ \mathbf{w}_{c,k} | k \in \mathcal{K}_c \}) \), the serving clusters effectively reduced to \( \mathbf{t}_c = \text{vec}(\{ t_{k,b} | (k, b) \in \mathcal{K}_c \times \mathcal{B}_c \}) \), \( \mathbf{u}_c = \text{vec}(\{ u_{c,k} | k \in \mathcal{K}_c \}) \), also \( \mathbf{y}_c = \text{vec}(\{ y_{c,k} | k \in \mathcal{K}_c \}) \) and \( \mathbf{r}_c = \text{vec}(\{ r_{c,k} | k \in \mathcal{K}_c \}) \). Each cloud \( c \) would then be able to solve problem (39) locally, via exchanging the interference terms \( \sum_{(c',k') \neq (c,k)} \mathbf{h}_{c',k'}^T (\mathbf{w}_{c',k'} + \mathbf{w}_{c,k})^2 \) with all other clouds \( c' \neq c \), required for constraint (39d). In fact, as per Algorithm 1, a distributed formulation necessitates the change of a few selected steps. More specific, in step 2 of Algorithm 1, the convergence criteria have to be checked per cloud individually. Also, steps 4 to 8, i.e., the inner loop, are performed on a per cloud basis, which leaves step 5 as a local optimization problem at CP \( c \). Therefore, the CPs would exchange interference information in every iteration of the outer loop as an additional step, i.e., between steps 8 and 9, all of which enable the overall distributed implementation of the algorithm. Note that the herein described procedure is not a fully distributed method. While the algorithm itself is amenable to distributed (decentralized) implementation, the clouds require a control-level coordination at the initialization state and after each outer loops iteration. However, contrary to the fully centralized algorithm, which requires signal-level coordination between the clouds, the proposed implementation is more practical and offers a simpler computational complexity.

While considering the distributed implementation of Algorithm 1 locally, i.e., the local step describes the guaranteed convergence to a stationary point.

**Theorem 1.** The distributed implementation of Algorithm 1, while executed at cloud \( c \), converges to a stationary point of the \( t_0 \)-relaxed, distributed version of problem (21), i.e., the relaxed problem at cloud \( c \), given the assumption of fixed interference from all other clouds \( c' \neq c \).

**Proof.** The proof of the above theorem can be found in Appendix C of the extended version of the paper available on archive [2].

I. **Complexity Analysis**

Now we focus on the overall computational complexity of our proposed method. Starting with the inner loop that utilizes a Dinkelbach-like algorithm, the overall complexity depends on each subproblems’ complexity as well as the convergence rate of the auxiliary problem series (42)-(43). Each subproblem (42) has a quadratic convex objective subject to quadratic convex constraints, and can hence be cast as a second order cone program (SOCP) [48]. Such problems can be solved using interior-point methods. Since the total number of variables for each subproblem is given by \( d_1 = (K(2B(L + 1) + 3)) \), the complexity metric becomes \( O((d_1)^{3.5}) \). We let \( V_{t, \max} \) be the worst-case fixed number of iterations for convergence of the Dinkelbach-like algorithm.

Since no optimization problem is solved in the outer loop, we can define \( V_{2, \max} \) as the worst-case fixed number of iterations needed for it to converge. The overall computational complexity of Algorithm 1 becomes, therefore, polynomial in the order of \( O(V_{1, \max} V_{2, \max} (d_1)^{3.5}) \), which is an upper bound on the complexity metric. We note, finally, that our proposed method consists of two instances of Algorithm 1, one for determining the serving clusters and one for finding a high-quality solution of beamforming vectors. The second instance of Algorithm 1 operates on the sparse optimization problem (45) with even fewer optimization variables, which typically requires fewer iterations; thereby reducing the overall complexity of the algorithm’s implementation.

**J. Fixed Clustering-Based Baseline**

Algorithm 1, particularly, finds optimal association variables \( \mathbf{t}, \mathbf{f}, \mathbf{u} \), which defines the user-to-BS association, as known as the clustering strategy. To benchmark our solution, we now fix the clustering strategy, and focus on finding optimal beamforming vectors by revisiting problem (39). The optimization variables now become group sparse variables \( \mathbf{y}_2 = [\mathbf{w}^T, \tilde{\mathbf{w}}^T, \gamma^T, \mathbf{r}^T]^T \). The fixed feasible variables are \( \mathbf{y}_2' = [\tilde{\mathbf{w}}^T, \gamma^T, \mathbf{r}^T]^T \). The optimization problem with fixed clusters can be written as

\[
\begin{align*}
\text{maximize} & \quad \sum_{c \in C} f_{3, \text{EE}}(c), \\
\text{subject to} & \quad (12b), (12g), (12h), (26)
\end{align*}
\]

A simpler version of Algorithm 1 is used (45), where the set of optimization variables is reduced to \( \{ \mathbf{w}, \tilde{\mathbf{w}}, \gamma, \mathbf{r} \} \). Such simplified approach is used to assess the performance of our proposed algorithm, as shown next. We refer to this scheme as Fixed Clustering Scheme (FCS).

V. **Numerical Simulations**

In this section, we present numerical simulations that illustrate the performance of our proposed algorithms. Consider an MC-RAN scenario occupying a square area of \([-400 400] \times [-400 400] \) m². The BSs and the users are randomly placed in the studied MC-RAN. The distribution is uniform. Each BS is equipped with \( L = 2 \) transmit antennas and all BSs share the same fronthaul capacity constraint. The maximum transmit power is set to 32 dBm for each BS. The channel model used for our simulations consists of a path-loss model \( PL_{\ell, k} = 128.1 + 37.6 \log_{10}(d_{\ell,k}) \), where \( d_{\ell,k} \) is the distance between BS \( \ell \) and user \( k \) in km, a log-normal shadowing with 8dB standard deviation, and a Rayleigh channel fading with zero mean and unit variance. The noise power \( \sigma^2 \) is set to \(-102 + 10 \log_{10}(\tau) + n_f \) dBm, where the channel bandwidth is set to \( \tau = 10 \) MHz and the noise figure to \( n_f = 15 \) dBm. The number of total files for caching is \( F = 100 \), we adopt the popularity aware cache placement scheme from [49]. The local memory size at each BS is considered to be 10 files otherwise mentioned. As for the popularity of the files, we use the Zipf distribution [49] with parameter \( \alpha = 0.15 \). Hence, for the content request mode of each individual user, it is assumed that each user requests a random file following...
the given popularity distribution. We choose the convergence parameters of Algorithm 1 as $\epsilon_1 = 0.2$ and $\epsilon_2 = 0.03$. Regarding the costs of the EE metric, $P_b^{\text{fthl}}$ is chosen to be 40% of the processing power, $P_k^{\text{proc}} = 20$ dBm, and $P_c^{\text{proc}} = 38$ dBm unless specified otherwise [50]. At first, we assume the full availability of files setting, where $q_{f_k,c} = 1$, for all $f_k \in F$ and $c \in C$, i.e., each file is available at all clouds. Imperfect availability settings are investigated in section V-H. At last, we define the number of users to be 28 and the number of BSs to be 10 unless mentioned otherwise. We propose optimizing beamforming vectors and serving clusters jointly in a dynamic clustering scheme. To best benchmark our methods, we use a static clustering scheme as a baseline to our proposed algorithm, where predetermined fixed clusters are used instead and the optimization is carried out on the beamforming vectors only. To determine such clusters, we use a load balancing algorithm applied in [9] for the case of a single cloud. This benchmark is referred to as FCS. Both schemes can be implemented either in a centralized or distributed fashion. The former algorithm is implemented at one CP, processing data from all clouds, i.e., all CPs are treated as one logical CP, while the latter algorithm is implemented at each CP, managing their respective computations through a reasonable level of cooperation.

A. Impact of Fronthaul Capacity

First, in Fig. 2, we evaluate the performance of the two schemes, static and dynamic clustering, both using centralized and distributed implementations. Fig. 2 shows the EE as a function of the fronthaul capacity for two different cache sizes, i.e., 10 in Fig. 2a, and 20 in Fig. 2b. Both figures show that our proposed dynamic clustering outperforms the fixed clustering approaches regardless of being implemented distributively or centrally. Particularly, since BSs might drop out of serving clusters due to overloading or power constraints, the need for dynamic clustering emerges, as such situations cannot be compensated by a fixed cluster. Further, the centralized implementation outperforms the distributed implementation for both schemes in Fig. 2a and Fig. 2b. Note that the gain of using a centralized instead of distributed implementation increases jointly with fronthaul capacity, i.e., the gap widens. In a low-fronthaul regime, the difference of both iterations is visibly insignificant, which highlights the role of our proposed distributed algorithm in limited fronthaul-capacity regimes, i.e., in cases where alleviating the fronthaul congestion is mostly required, as the performances of distributed and centralized algorithms become relatively similar. To explain the behavior in high fronthaul regime, we note that the performance bottleneck shifts from fronthaul link (low capacity regime) to interference management (high capacity regime). Due to this relation, more sophisticated interference management techniques become necessary in the high fronthaul region. As the centralized implementation can manage the network-wide interference more efficiently, we observe the increased gap between the two implementations. In contrast, the distributed algorithm is only capable of managing its own cloud-wide interference, whereas the interference from other clouds is fixed in each iteration. This major difference leads to the observed gap.

A general observation from comparing Fig. 2a and 2b is that the EE gain increases when bigger cache sizes at the BSs are employed. For the dynamic centralized scheme, there is a 10% gain at 20 Mbps. This is particularly the case since the EE metric benefits from cache hits, mainly because a user can be served without utilizing the fronthaul link while requiring processing costs at the respective BS. At lower capacity regimes, fronthaul capacity is a scarce resource, and so the activation of a fronthaul link is a sensible decision, and the EE metric gains significantly from cache hits in such situations.

B. Processing Power vs. Caching Gain

In the second set of simulations, we consider only the distributed implementation of Algorithm 1 with dynamic clustering. In Fig. 3, we compare the EE for two different processing costs. We also vary the cache sizes for fixed costs, i.e., no cache up to a cache size of 50 files. From Fig. 3, it becomes clear that the EE gain from a larger cache size decreases with increasing the fronthaul capacity, i.e., the gap between the plots decreases at the high fronthaul regime. As expected, a higher caching processing power would reduce the EE. Such behavior can be recognized by comparing Fig. 3a and 3b. The EE for a cache size of 10 files at 40 Mbps fronthaul capacity decreases by 26.54% as the processing cost increases from 10 dBm to 25 dBm. Interestingly, the gain of utilizing a bigger cache size also decreases with higher processing powers. In fact, Fig. 3b shows that there is no conceivable gain from utilizing caching capabilities for $P_k^{\text{proc}} = 25$ dBm at 60 Mbps fronthaul capacity.

C. Comparison of Cache Placement Strategies

Before introducing other cache placement strategies, let us first revisit the user’s file request pattern. The popularity of
files follows a Zipf distribution with parameter \( a \). That is, if the elements of a the set of files, hereafter denoted by \( \mathcal{F} \), are ordered according to the order of popularity, i.e., the probability of requesting the \( f \)-th file is \( p(f) \sim f^{-a} \). In more details, a small \( a \) refers to the popularity of files being close to uniformly distributed, which means that users request each file with almost equal probability. In contrast, a large \( a \) shifts the distribution towards only some files being very popular whilst the rest has low popularity. This behavior can be observed in Fig. 4b for 25 files. We show the fraction of the files demand for each file \( f \). As an example, for \( a = 1 \), file 1 is demanded with a probability of over 25%, while file 25 is requested by a user with probability 2%.

In addition to the popularity aware cache placement scheme considered throughout the simulations, in this subsection, we consider three other caching strategies adopted from [49]. Under popularity aware cache placement, each BS’s cache is filled with random files following the Zipf distribution. That is, the probability of caching a file \( f \) is proportional to its popularity. Consequently, the cache hit ratio depends mostly on the parameter \( a \), where small values of \( a \) would produce less cache hits, as the popularity of files is almost uniformly distributed. As \( a \) becomes larger, the cache hits increase. Additionally, we include the conservative scheme, which stores the same files at every BS, i.e., the most popular files are cached at every BS respecting the individual cache size constraints. Under such scheme, the cache hit ratio becomes high, yet still dependent on \( a \). Two other cache placement schemes are implemented, namely minimum redundancy strategy and the random caching scheme. While the former tries to minimize the file redundancy among all BS caches, the latter stores random files in each BS’s memory. Note that random caching is rather considered as a lower performance bound.

Fig. 4a shows the EE as a function of the Zipf parameter \( a \) for the four considered cache placement strategies. Different to previous considerations, we set \( P_{b} = 15 \text{ dBm}, P_{b}^{\text{fhl}} = 9 \text{ dBm} \) and \( P_{b,c} = 40 \text{ Mbps} \), and we set the cache size to 10 files. As expected, the EE metrics of the conservative and popularity aware caching schemes (group (i)) increase with \( a \), and thus with increasing number of cache hits, as shown in Fig. 4a. While the file popularity is close to uniformly distributed at \( a = 0.1 \), all four considered cache placement strategies perform almost equally in terms of EE. However, with increasing number of cache hits, as shown in Fig. 4a.

The impact of cache hit ratio on EE becomes more pronounced. Conservative caching is particularly effective when \( a \) becomes large, and popularity aware caching behaves similarly. Under the minimum redundancy caching scheme (group (ii)), the EE is constant as \( a \) increases and decreases at \( a = 1 \), i.e., only a few files are very popular, while the rest of the files experiences low popularity. This is due to the fact that using such scheme, all files, including the less popular ones, are cached. The same argumentation can be done for the random caching scheme (group (ii)). That is, a lower cache hit ratio may impact the EE performance negatively. From Fig. 4a, we conclude that schemes which utilize knowledge of system parameters (i.e., \( a \)), e.g., conservative and popularity aware caching, are more beneficial from an EE perspective than more primitive schemes, and are thus better adopted in the context of our paper.

D. Impact of Processing Power

Before describing the next set of simulations, we introduce another state-of-the-art scheme, which is used as a baseline in our next set of simulations. In [25], the authors propose fixing the optimization variable \( \hat{f} \) a priori, as cache placement and user requests are known, before jointly optimizing serving clusters and beamforming vectors. A BS that caches the requested file for user \( k \) has to serve this user and process its data locally. This method leaves no choice for a BS to leave the computation to the respective CP. This is motivated by the fact, that from an EE perspective, local processing should be preferred to cloud computing since energy usage is lower in some regimes. This state-of-the-art scheme is referred to as Forced Local Computation (FLC).

In Fig. 5a, the EE as a function of processing power \( P_{k}^{\text{proc}} \) is shown for two different cache sizes, where the fronthaul capacity is set to 40 Mbps. In these simulations, our proposed scheme achieves better EE, when considering the required processing power for local caches. Interestingly, focusing on our proposed scheme, we notice a convergence of the EE for the two cache sizes, when \( P_{k}^{\text{proc}} \) increases. This matches the observations from V-B. In fact, in networks where users require computationally intensive services, caching may not be helpful as an EE metric. The main cause for such behavior comes from the EE metric itself, since both BSs and CPs have to allocate \( P_{k}^{\text{proc}} \) when serving user \( k \). As this value increases,
We only consider our proposed dynamic clustering scheme. Such characteristic is important for B5G networks, especially compared to the proposed distributed solution than the centralized counterpart. To tackle the problem of comparing a maximized sum-rate to a maximized EE, we propose converting the results of the final optimization variables from the sum-rate maximization problem using a different fractional programming formulation for the binary association part. Towards that end, [1] proposes an efficient iterative algorithm for joint association and beamformer design that can be implemented in a distributed fashion across multiple CPs. To conduct a fair comparison, we now examine a special case of Algorithm 1, i.e., the case when there are no cache hits ($K = \emptyset$ and $K_2 = K$). This is due to the fact that we do not consider edge intelligence and caches in [1]. We fix the processing power as $P_k^{\text{proc}} = 10$ dBm and the number of BSs as 14.

### E. Impact of Network Densification

In Fig. 5b, we examine the EE as a function of the number of users. Consider a network of 14 BSs, where the processing power is set to $P_k^{\text{proc}} = 10$ dBm and the fronthaul capacity is 80 Mbps. As a first observation, we find that, generally with more users, the EE increases. Similar to previous observations, we observe that the dynamic implementation always outperforms the fixed association implementation for all schemes, which highlights the numerical aspect of the clustering approach proposed in our paper.

Additionally, we observe the running times of centralized and distributed implementations under a cache size of 10 and 20 files per BS, respectively. Table I shows these times normalized to the distributed implementation with 10% caching versus the density of users in the network (users/km$^2$). While both distributed versions are almost on par with increasing network density, a substantial increase in the centralized implementations relative run times can be observed, which stems from its higher computational complexity. In fact, the centralized algorithm endures substantial speed loss and scales badly with network density. For example, at 31.2 users/km$^2$ density, both centralized versions take about 3 times more run time to provide a solution, while at 78.1 users/km$^2$ this factor becomes around 5. Such results implicate a better scalability of the proposed distributed solution than the centralized counterpart. Such characteristic is important for B5G networks, especially in the context of latency-sensitive applications, which further illustrates the numerical prospects of our proposed algorithm.

### F. Convergence Behavior

In another set of simulations, the processing power $P_k^{\text{proc}}$ is set to 10 dBm and the BSs can cache up to 20 files. We only consider our proposed dynamic clustering scheme.

To now illustrate the convergence behavior of our proposed Algorithm 1, we plot the objective as a function of the number of iterations executed until converging in Fig. 6a. We compare a distributed as well as a centralized implementation for different fronthaul capacities, i.e., $F_{b,c} \in \{20, 40, 60, 80\}$ Mbps. In Fig. 6a, the advantages of our algorithm in terms of convergence behavior and execution speed are highlighted, as the maximum iterations required for convergence are comparatively low. In all cases, the centralized implementation takes more iterations until convergence as compared to the distributed implementation. Fig. 6a also shows that, at low fronthaul capacities, the distributed algorithm has acceptable loss in terms of EE against a centralized approach and, at the same time, performs better in terms of convergence, which constitutes an additional numerical feature of our proposed distributed resource allocation framework. Again, in high fronthaul regime, the distributed implementation is outperformed by the centralized algorithm, as the latter is a more sophisticated interference management technique, which is necessary as interference becomes the major bottleneck in such regime.

### G. Comparison with Sum-Rate Maximization Algorithm

Herein, we make a connection to the weighted sum-rate maximization in MC-RAN from our conference version [1], which solves a mixed discrete-continuous, non-convex optimization problem using a different fractional programming approach to tackle the non-convex part and $l_0$-norm approximation for the binary association part. Towards that end, [1] proposes an efficient iterative algorithm for joint association and beamformer design that can be implemented in a distributed fashion across multiple CPs. To conduct a fair comparison, we now examine a special case of Algorithm 1, i.e., the case when there are no cache hits ($K_1 = \emptyset$ and $K_2 = K$). This is due to the fact that we do not consider edge intelligence and caches in [1]. We fix the processing power as $P_k^{\text{proc}} = 10$ dBm and the number of BSs as 14. To tackle the problem of comparing a maximized sum-rate to a maximized EE, we propose converting the results of the algorithm from [1] into the EE metric. Therefore we parse the final optimization variables from the sum-rate maximization into the EE formulation in (38). Such comparison is done for both centralized and distributed implementations in Fig. 6b. Fig. 6b shows that, for every fronthaul capacity, both Dinkelbach-like implementations, referring to Algorithm 1.

---

**TABLE I: Normalized Run Times**

| Density in users/km$^2$ | 10% Cache Centralized | 10% Cache Distributed | 20% Cache Centralized | 20% Cache Distributed |
|-------------------------|-----------------------|-----------------------|-----------------------|-----------------------|
| 15.6                    | 1.00                  | 1                     | 1.58                  | 1.02                  |
| 31.2                    | 3.03                  | 1                     | 3.13                  | 1.02                  |
| 46.9                    | 4.00                  | 1                     | 4.09                  | 0.96                  |
| 62.5                    | 5.04                  | 1                     | 5.07                  | 0.91                  |

---

Fig. 6: EE as a function of different system parameters comparing various schemes.
outperform the sum-rate maximization (SRM) implementation. Interestingly, we see a difference in the gain of using a centralized over a distributed implementation in the SRM case. Different from previous observations about our proposed scheme, the loss of the distributed SRM implementation in terms of EE is vastly increased.

H. Imperfect Availability

In the last set of simulations, we consider a network consisting of 28 users and 12 BSs coordinated by 4 clouds via a fronthaul link of 50 Mbps capacity. At this point, it is important to analyze the impact of various file availability scenarios. Therefore, we focus on a new parameter $\kappa$, which refers to different availability partitions. In more details, we now assume each cloud to only store a fraction of the whole library of files $F = \{1, \ldots, F\}$. Every cloud stores only $\kappa \cdot F$ random chosen files. Initializing $Q$, we make sure to keep each file available at one cloud at least. In Fig. 6c, we plot the EE for the distributed and centralized implementation of Algorithm 1 as a function of different values of file availability at the clouds. That is, we choose $\kappa \in \{0.3, 0.4, \ldots, 1\}$. For reference, we also plot the EE performance of the full availability scheme ($\kappa = 1$) as an upper bound line of constant value for both centralized and distributed implementations, respectively. From Fig. 6c, we observe that the EE of both implementations decreases with lower values of $\kappa$. In the $\kappa = 1$ case, each user may be associated with the best BSs possible (i.e., according to the channel gain) by solving problem (14). Because each cloud stores the whole library of files, such association is performed without connection restrictions from $q_{f_k,c}$. These connection restrictions become more severe with decreasing values of $\kappa$, as the file availability decreases. That is, a user may possibly connected to weaker BSs, and thus weaker clouds, only since no other cloud stores the requested file. Indeed, more realistic scenarios, where different clouds store different files, endure performance loss compared to the optimistic assumption of full availability as can be seen in Fig. 6c.

I. Discussions and Design Recommendations

Rounding up the numerical simulation section, we herein aim to provide some concluding discussions, pros and cons of the proposed method, and design recommendations. In summary, our work provides a framework for including cache-assisted C-RAN in the multi-cloud paradigm. Considering the EE metric under a hybrid transmit strategy, a distributed implementation which requires only reasonable information exchange between the cloud instances is developed. For easy overview, we next present pros and cons of the proposed algorithm. The proposed algorithm is able to outperform various reference schemes, e.g., FCS, FLC, and SRM. Further, the distributed implementation performs close to the centralized implementation in low fronthaul regime and achieves a constant gap to the centralized solution for different numbers of users. Different results show the scalability of the proposed scheme in dense networks. On the other hand, the solution is not globally optimal and depends on good quality heuristics. Further, the gap to the centralized solution increases at high fronthaul capacity levels. More specifically, while the proposed solution is not globally optimal, the simulation results illustrate its numerical prospects, both from performance and complexity perspectives (including the running time).

The general recommendation is, therefore, to adopt our proposed solution, especially in large-scale networks, where the scalability analysis shows that the computational burden of a centralized algorithm becomes huge in dense networks. Also, in low fronthaul regime, there is no major EE gain from utilizing a centralized implementation. The authors recommend the consideration of MC-RAN architecture for most 5G networks, especially in dense areas where a single-cloud solution is often infeasible. Usage of edge caching in the considered MC-RAN is, nevertheless, mostly effective at low and medium fronthaul capacities, low to medium processing powers, and utilizing sophisticated cache placement strategies. However, in high fronthaul and processing power regime, edge caching has few to none benefits for the EE.

VI. CONCLUSION

Managing wireless systems with multiple CPs is a promising technique to cope with 5G network requirements. This paper considers an MC-RAN, where each cloud is connected to a distinct set of BSs via limited capacity fronthaul links. The BSs are equipped with local cache storage and baseband processing capabilities, as a means to alleviate the fronthaul congestion problem. The paper then investigates the problem of jointly assigning users to clouds and determining their beamforming vectors so as to maximize the network-wide energy efficiency subject to fronthaul capacity and transmit power constraints. This paper solves such a mixed discrete-continuous, non-convex optimization problem using fractional programming and successive inner-convex approximation techniques to deal with the non-convexity of the continuous part of the problem, and $l_0$-norm approximation to account for the binary association part. A highlight of the proposed algorithm is the capability of implementing it in a distributed fashion across the network’s multiple clouds through a reasonable amount of information exchange. The numerical simulations illustrate the pronounced role the proposed algorithm plays in alleviating the interference of large-scale MC-RANs, especially in dense networks.

REFERENCES

[1] A. A. Ahmad, H. Dahrouj, A. Chaaban, A. Sezgin, T. Y. Al-Naffouri, and M. Alouini, “Distributed cloud association and beamforming in downlink multi-cloud radio access networks,” in IEEE ICC Workshops, 2020, pp. 1–6.
[2] A. A. Ahmad, R.-J. Reifert, H. Dahrouj, A. Chaaban, A. Sezgin, T. Y. Al-Naffouri, and M.-S. Alouini, “Distributed resource management in downlink cache-enabled multi-cloud radio access networks,” 2021. [Online]. Available: https://arxiv.org/abs/2104.03664
[3] L. Zhang, Y. Liang, and D. Niyato, “6G Visions: Mobile ultra-broadband, super internet-of-things, and artificial intelligence,” China Communications, vol. 16, no. 8, pp. 1–14, Aug. 2019.
[4] M. R. Palattella, M. Dohler, A. Greico, G. Rizzo, J. Torsner, T. Engel, and L. Ladid, “Internet of things in the 5G era: Enablers, architecture, and business models,” IEEE J. Sel. Areas Commun., vol. 34, no. 3, pp. 510–527, 2016.
[5] L. Zhang, M. Xiao, G. Wu, M. Alam, Y. Liang, and S. Li, “A survey of advanced techniques for spectrum sharing in 5G networks,” IEEE Wireless Commun., vol. 24, no. 5, pp. 44–51, 2017.
“Ericsson mobility report november 2019,” Ericsson, Tech. Rep. MSU-CSE-06-2, Nov. 2019. [Online]. Available: https://www.ericsson.com/en/mobility-report/reports/november-2019

T. Quek, M. Peng, O. Simone, and W. Yu, *Cloud Radio Access Networks: Principles, Technologies, and Applications*. Cambridge University Press, 2017.

K. Shamnamug, N. Golrezai, A. G. Dimakis, A. F. Molisch, and G. Caire, “Femtocaching: Wireless content delivery through distributed caching helpers,” *IEEE Trans. Inf. Theory*, vol. 59, no. 12, pp. 8402–8413, 2013.

B. Dai and W. Yu, “Sparse Beamforming and User-Centric Clustering for Downlink Cloud Radio Access Network,” *IEEE Access*, vol. 2, pp. 1326–1339, 2014.

J. Tang, W. P. Tay, T. Q. S. Quek, and B. Liang, “System cost minimization in cloud RAN with limited fronthaul capacity,” *IEEE Trans. Wirel. Commun.*, vol. 16, no. 5, pp. 3371–3384, May 2017.

S. Park, O. Simone, O. Sahin, and S. Shamai, “Inter-cluster design of precoding,” vol. 29, no. 2, pp. 6–14 for cloud radio access networks,” *IEEE Wireless Commun. Lett.*, vol. 3, no. 4, pp. 369–372, Aug. 2014.

O. Dhiyalallah, H. Dahrourj, T. Y. Al-Naffouri, and M. S. Alouini, “Distributed robust power minimization for the downlink of multi-cloud radio access networks,” in *IEEE GLOBECOM*, Dec. 2016, pp. 1–6.

H. Dahrourj, T. Y. Al-Naffouri, and M. S. Alouini, “Distributed cloud association in downlink multicloude radio access networks,” in *49th CISS*, Mar. 2015, pp. 1–3.

S. Gelinic and G. Rakaya-Ben Othman, “Degrees-of-freedom in multi-cloud based sectorized cellular networks,” *Entropy*, vol. 22, no. 6, 2020. [Online]. Available: https://www.mdpi.com/1099-4300/22/6/668

A. Alameer and A. Sezgin, “Interference mitigation via rate-splitting and common message decoding in cloud radio access networks,” *IEEE Access*, vol. 7, pp. 80350–80365, 2019.

K. Weinberger, A. A. Ahmad, A. Sezgin, and A. Zappone, “Synergistic benefits in IRS- and RS-enabled C-RAN with energy-efficient clustering,” *IEEE Trans. Wirel. Commun.*, pp. 1–1, 2022.

H. Dahrourj, T. Y. Al-Naffouri, and M. S. Alouini, “Distributed cloud association in downlink multicloud radio access networks,” in *49th CISS*, Mar. 2015, pp. 1–3.

D. Liu and C. Yang, “Will caching at base station improve energy efficiency of downlink transmission?” in *IEEE GlobSIP*, 2014, pp. 173–177.

J. Liu, B. Bai, J. Zhang, and K. B. Letaief, “Cache placement in fog-RANs: From centralized to distributed algorithms,” *IEEE Trans. Wirel. Commun.*, vol. 16, no. 11, pp. 7039–7051, 2017.

M. A. Maddah-Ali and U. Niesen, “Fundamental limits of caching,” *IEEE Trans. Inf. Theory*, vol. 60, no. 5, pp. 2856–2867, 2014.

X. Han, N. Ansari, M. Wu, and H. Yu, “On accelerating content delivery in mobile networks,” *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1314–1333, 2013.

Y. Cheng, M. Pesavento, and A. Philipp, “Joint network optimization and downlink beamforming for comp transmissions using mixed integer conic programming,” *IEEE Trans. Signal Process.*, vol. 61, no. 16, pp. 3972–3987, 2013.

A. De Domenico, Y. F. Liu, and W. Yu, “Optimal virtual network function deployment for 5G network slicing in a hybrid cloud infrastructure,” *IEEE Trans. Wirel. Commun.*, vol. 19, no. 12, pp. 7942–7956, 2020.

NR: physical layer procedures for data, v15.3.0,” document 3GPP TS 38.214, Sep. 2018.

H. Dahrourj, A. Douik, O. Dhifallah, T. Y. Al-Naffouri, and M.-S. Alouini, “Resource allocation in heterogeneous cloud radio access networks: advances and challenges,” *IEEE Wirel. Commun.*, vol. 22, no. 6, pp. 66–73, 2017.

L. Liu and W. Yu, “Cross-layer design for downlink multihop cloud radio access networks with network coding,” *IEEE Trans. Signal Process.*, vol. 65, no. 7, pp. 1728–1740, Apr. 2017.

A. Douik, H. Dahrourj, T. Y. Al-Naffouri, and M.-S. Alouini, “Distributed hybrid scheduling in multi-cloud networks using conflict graphs,” *IEEE Trans. Commun.*, vol. 66, no. 1, pp. 209–224, Jan. 2018.

H. Dahrourj, R. Alghamdi, H. Alwazani, S. Bahanshal, A. A. Ahmad, A. Faisal, R. Shalabi, R. Alhadrami, A. Subasi, M. T. Al-Nory, O. Kitan, and J. S. Shamha, “An overview of machine learning-based techniques for solving optimization problems in communications and signal processing,” *IEEE Access*, vol. 9, pp. 74908–74938, 2021.

S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.