

Received July 27, 2020, accepted August 15, 2020, date of publication September 4, 2020, date of current version September 21, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3021948

Novel Cloud-RRH Architecture With Radio Resource Management and QoS Strategies for 5G HetNets

OLFA CHABBOUH^{1,2}, SONIA BEN REJEB^{1,5}, (Member, IEEE),
NIDAL NASSER³, (Senior Member, IEEE),
NAZIM AGOULMINE⁴, (Senior Member, IEEE),
AND ZIÈD CHOUKAIR¹

¹Mediatron Laboratory, Higher School of Communication of Tunis (Sup'Com), Ariana 2083, Tunisia

²Private Higher School of Engineering and Technology (ESPRIT), Tunis 2083, Tunisia

³Department of Software Engineering, Alfaisal University, Riyadh 11533, Saudi Arabia

⁴IBISC-IBGBI Laboratory, University of Évry Val d'Essonne, 91000 Évry, France

⁵Higher Institute of Computer Science (ISI), Ariana 2080, Tunisia

Corresponding author: Nidal Nasser (nnasser@alfaisal.edu)


This work was supported by the Alfaisal University through the Internal Research Grant 18206.

ABSTRACT With the increase of data traffic in the global mobile network, the limitation in resources of data computing close to the edge is becoming an important issue to resolve. This article addresses the cloud radio access network (CRAN) in 5G HetNets architecture and proposes to take benefit of extra computing and storage resources in the edge to enable the offloading of a set of mobile user services from the remote cloud computing servers to an edge cloud computing infrastructure deployed next to remote radio heads (RRHs) to better serve mobile users and improve energy efficiency. However, this architecture poses many challenges. The first one is related to the clustering of the various deployed RRH to better serve end-users. For that, we propose a two-stage RRH clustering mechanism in order to fully exploit the benefits of C-RAN architecture. The second challenge is related to the scheduling of the offloading. Therefore, we propose a cost-based scheduling scheme (CBSS) that aims to minimize the scheduling cost while considering resource availability in the infrastructure, resource requirements from users' applications, services execution deadlines, and load balancing. The proposed solution permits us to make better offloading decisions and to improve the users' experiences. The solution was implemented in a simulator to highlight its performances and compare them with other existing approaches.

INDEX TERMS Cloud RAN, 5G HetNets, edge Cloud-RRH, radio resource management, offloading, resource scheduling, resource cost.

I. INTRODUCTION

The evolution toward global mobile networks has been characterized by an exponential traffic growth during the last years. The network traffic has roughly doubled every year since 2011 [1]. This growth is mainly due to the huge success of smartphones and tablets and emerging applications. Currently, smartphones and tablets are powerful enough to run a large variety of applications (entertainment, health care, business, social networking, traveling, news, etc.). Many of these applications are based on a client/server model with

The associate editor coordinating the review of this manuscript and approving it for publication was Abbas Jamalipour .

a light client running in the mobile device (i.e., native or web client) and a heavy server running in the cloud, thereby overcoming the limited capacities of end-user mobile devices. This necessitates to setup end-to-end communication from the mobile terminal to the application or service logic running in the far-end cloud computing infrastructure.

Mobile Cloud Computing (MCC) has been introduced to allow the execution of the application or service logic closer to the user localization. For that, it advocates the deployment of additional cloud computing resources in the edge network to run some applications and eventually improving the end-user experience by reducing the response times since the requests of the application clients are processed in the

edge cloud computing infrastructure and not in the far end cloud computing infrastructure.

In the 5G network architecture, a cloud-based radio access network has been proposed for decoupling the baseband units (BBUs) from remote radio heads (RRHs) and for moving them into the cloud to enable centralized processing and management. With this approach, a complex base-station operation can be simplified to cost-effective and power-efficient RRHs by centralizing the processing. This simplification facilitates the efficient management of large-scale small-cell systems.

In conventional mobile network architecture, one BBU is logically associated with only one RRH. Therefore, radio resources may be underutilized since one RRH may not consume all the available BBU resources. Moreover, this one-to-one assignment is very expensive for 5G networks, where a large number of small cells will be deployed. Cloud RAN (C-RAN) architecture solves this problem by allowing a one-to-many mapping between BBUs and RRHs. Thus, BBU radio resources could be more efficiently used, decreasing not only the cost of infrastructure but also its energy consumption.

Another important mechanism that is used in this work is the Mobile data offloading. This mechanism aims to benefit from external additional communication resources (such as the availability of Wi-Fi network access at the same location as cellular networks) to migrate the communication of the data from the mobile network access to the other access to benefit of its higher capacities. The concept of offloading can also be applied to the processing resources and in this case, the objective is to migrate the communication of the data to the near system where there are additional storage and processing resources. Offloading has also been used with IoT in order to improve traffic management. In fact, with the explosive growth of connected objects and the corresponding IoT applications, massive traffic needs to be processed in time. To deal with this challenge, offloading tasks to fog nodes was proposed [43].

Several state-of-the-art approaches have exploited cloud computing technology for this purpose [2]. However, these approaches have mainly introduced the computing capacities in the core network and not on the edge of the network with the risk of degrading the performances if the geographic distance with the users is far. Indeed, the offloading benefits may be easily wasted and energy consumption increased.

In this work, we propose a novel cloud RAN heterogeneous architecture, namely, Edge Cloud-RRH that aims to introduce an edge cloud computing infrastructure close to the C-RAN infrastructure introduced by 5G networks. In this approach, we propose adding additional computational and storage resources to high RRH infrastructures (macrocells) installed in the edge network close to mobile end-users. With this infrastructure, it will be possible to offload (i.e., move closer) applications/services from the far end cloud computing infrastructure close to install and execute them in the edge cloud computing infrastructure, the Edge Cloud-RRH

allowing short distance interaction between the end-users terminals and the applications/services servers. The technology that is able to support this offloading is named containerization [3]. It provides a resource abstraction in terms of virtualization and isolation based on containers that surpass other virtualization techniques. To fully capitalize on the proposed architecture, we further introduce radio resource management (RRM) strategies that aim to efficiently schedule offloading requests among the containers. The proposed solution encompasses a two-stage RRH clustering mechanism and a cost-based task scheduling scheme to reduce the overload and the migration costs.

The remainder of this article is organized as follows. Section II discusses relevant literature and the motivation for this work. Section III describes preliminaries and the system model of our proposed approach and provides an overview of the operational aspect of the solution. Then, the radio resource management strategies are introduced in Section IV, together with the proposed solution algorithms. Performance evaluation of the strategies is presented in Section V. Finally, Section VI presents the conclusions of this work and envisioned future work.

II. RELATED WORK

In this section, we present and discuss several studies that have considered resource management in C-RAN RRH heterogeneous networks.

A. RRH CLUSTERING

RRH clustering has already been proposed in the literature for many purposes. In [4], the authors formulated the problem of RRH clustering as a bin packing problem. They proposed optimal and heuristic solutions to improve network energy consumption without compromising the QoS (Quality of Service) expressed as numbers of required resource blocks in available RRHs. In [5], the authors studied the problem of joint activation and clustering of RRH to improve user utility function. They introduced a coverage constraint to the problem formulation to ensure the connectivity for all end-users. The overall end-user QoS is defined as a weighted sum of SIR (signal to interference ratio) and the average number of users assigned to one RRH. In another work [6], authors proposed a lightweight and load aware dynamic RRH assignment (DRA) algorithm. The proposed algorithm reduced the complexity of the clustering procedure and offered quite close BBU resource savings as compared to FFD (First Fit Decreasing). In [7], the authors introduced a greedy RRH clustering algorithm for C-RAN downlink, which aims to maximize the sum-rate gain and reduce network piloting overhead. Authors in [8] presented a greedy dynamic RRH clustering mechanism-based multi-objective optimization to balance the throughput maximization and RRH energy consumption minimization. In [9], the authors focused on delay-tolerant best-effort traffic to derive a dynamic clustering and user scheduling approach for cooperative base stations. In [41], the authors proposed to jointly optimize RRHs clustering

and beamforming assignment in Cloud-RAN downlink. They developed a heuristic algorithm to provide sub-optimal performance by Dinkelbach's transformation. Next, to handle non-convex formulation, they proposed an iterative approximation algorithm with linear approximation and constraints relaxation.

In all these works, the proposed approaches were focused on homogeneous scenarios (i.e., clustering one types of cooperating RRHs). However, 5G systems are deemed to be heterogeneous networks with various type of RRHs associated with the different types of cells (macro and small).

In our work, we address explicitly the heterogeneity in the Cloud RAN architecture and we propose a two stages RRH clustering mechanism. In the first step, we propose to use a fuzzy logic controller to cluster Low RRHs (small cells) considering their inter and intracluster interferences. In the second step, we propose to cluster High RRHs (macro-cells) considering the global system interference and the BBU system capabilities. The problem is modelled as a linear program optimization and a heuristic resolution algorithm. The aim is to study different RRHs clustering techniques in order to reduce network power consumption and resource utilization while assuring the required level of QoS to end-users applications.

B. OFFLOADING

Offloading mobile data from cellular networks to Wi-Fi and vice versa has become an important feature in cellular networks for alleviating the impact of data on the quality of cellular connections.

Various cloud offloading systems were proposed in the literature [10], [12]. MAUI [11] and ThinkAir [12] profile hardware components and perform offloading to optimize mobile devices' energy consumption. These approaches consider only the energy consumption but unfortunately, ignore other offloading aspects that may impact the service quality. CloneCloud [13] optimizes mobile applications' partitioning between local executions (in the mobile terminal) and offloading the execution in the cloud computing with the objective of minimizing the execution time or the energy consumption.

In addition to mobile cloud frameworks, other research efforts focused on offloading decision-making. For example, Yuh-Shyan *et al.* [14] proposed an energy-aware data offloading scheme for C-RANs in which a BBU takes the offloading decisions considering the mobile terminal's transmission rate and energy consumption of both the cellular network and the Wi-Fi network. The main strategy of C-RAN is to schedule the offloading decision of a mobile terminal from the RRH to the Wi-Fi access point. The proposed scheme eventually reduced the energy consumption and improved the throughput of the network. However, the performance improvement was readily observable only if the data size is sufficiently large.

Another offloading decision algorithm in the context of the femto-cloud paradigm is proposed by Oueis *et al.* in [15].

The algorithm performs a series of classifications that consider several parameters, including the latency, battery level, and mobile terminal memory, without including them in a complex optimization problem. The proposed algorithm categorizes the application tasks into "urgent" and "not urgent" and sends urgent and offloadable tasks to the femto-cloud regardless of the channel conditions. This strategy affects mobile handset's energy consumption and violates the latency constraints if the total offloading exceeds a certain threshold (3%).

In [16], Ting-Yi *et al.* proposed a context-aware decision algorithm (CADA) for offloading mobile applications to the cloud servers. The decision engine is composed of four components: a context-aware decision algorithm, a context profiler, an energy model, and a context database. Through the context profiler, CADA uses the location and the time of day in its decision-making for the application of individual methods. However, such profiling is a source of overhead and requires extensive memory for storing and processing users' profiles.

In [40], the authors' objective was to jointly optimize the task offloading decision, elastic computation resource scheduling, and radio resource allocation. They formulated the problem as a stochastic mixed-integer nonlinear programming problem. They eventually theoretically analyzed the tradeoff between energy efficiency and service delay.

Various factors have been considered in the offloading decisions introduced in the literature. In [17], Huijun *et al.* proposed an offloading decision model that takes network unavailability into consideration. Based on the network connection states and durations that are recorded in a history buffer, the partition of the application is calculated, such that to improving the performance of a network. Based on the level of improvement, the decision offloading is validated or not. The proposed algorithm enhances network performances in terms of execution time and energy consumption; however, system's complexity increases with the number of mobile users.

In summary, mobile cloud offloading was already studied for various networks and several decision algorithms were proposed. The proposed solutions exploited multiple-objective optimization techniques. However, due to the changes in the network states, a decision must be refreshed if the system conditions change.

C. SCHEDULING CONSIDERATIONS

Scheduling user's computing tasks is a key challenge in a cloud computing environment. The optimal allocation of resources and scheduling of the offloading requests helps to guarantee application performance and reduce operating costs. Several existing works have addressed these issues and are discussed in this section.

In [18], Katyal and Mishra proposed a selection algorithm that uses the standard deviation to decide which scheduling algorithm to use among two; namely, Min-Min and Max-Min, for minimizing the total execution time of tasks. In [19],

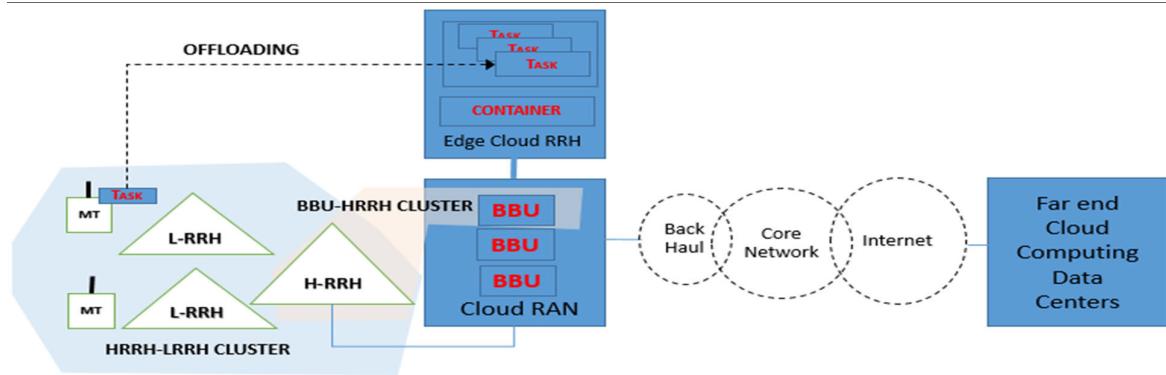


FIGURE 1. Proposed CRAN Architecture.

an improved Max-Min algorithm is modified by Santhosh and Manjaiah to define two new algorithms based on the average execution time. In contrast to Max-Min, a task with a run time that is slightly above the average run time is selected and assigned to the resource that yields the minimum run time. The average run time is calculated via the arithmetic mean for independent tasks and via the geometric mean for dependent tasks. The main objective is to reduce task makespan. Barbarossa *et al.* proposed a task scheduler with three priority levels [20]: the scheduling level, which represents the objective to be achieved by the planner; the resource level that represents the available attributes to fulfill the desired objective; and the task level, which represents the available alternatives among which the best task should be scheduled first. Therefore, each task requests resources under a preset priority and is scheduled accordingly. In [21], Thomas *et al.* proposed a task scheduler that is based on credits. In this approach, tasks are assigned credits based on two parameters, namely, the user priority and the task duration, and the task with the highest credit value is executed first. A scheduler that is based on particle swarm optimization (PSO) was proposed by Khalili and Babamir in [22]. PSO is a population-based search algorithm that is inspired by bird flocking and fish schooling, where each particle learns from its neighbors and itself during the time it travels in space. As with other metaheuristics, this PSO-based scheduler does not offer guarantees or bounds on the generated solution, especially as the search space expands.

Himani and Sidhu proposed a cost-deadline based (CDB) scheduler in [23], where the scheduling cost is calculated according to the task length, the deadline, and the number of required processing elements. Then, tasks are sorted to determine the execution order via a task-to-VM mapping that is based on a Min-Min heuristic. Via this approach, the CDB minimizes missed deadlines.

In [24], the authors have investigated cost-based scheduling using linear programming. They have proposed a task scheduling algorithm that is based on delay bound constraint (SAH-DB) for improving the task execution concurrency: when a task is received, all the resources (CPU, memory,

and network) are sorted in descending order of the resource processing capacity; then, the task is dispatched to resources with the minimum execution time.

In this article, we propose a novel CRAN architecture in which an Edge Cloud-RRH is introduced into the edge network. By Edge Cloud-RRH we mean, cloud computing capabilities associated with the RRH and able to support the execution of containers. While most previous works have focused on the job completion time, we propose in this work resource management strategies based on a joint offloading and scheduling optimization mechanism aim to reduce the cost of task scheduling across all application tasks encapsulated in containers. In contrast to previous works, we model the cost of tasks as a function of the overloading state, delay time, and migration time. The scheduling process is implemented considering the available resources, the resource requirements, deadline, and load balancing in the Edge Cloud-RRH infrastructure.

III. SYSTEM MODEL AND OPERATION OVERVIEW

A. PRELIMINARIES

The objective of the proposed approach is to overcome the challenges that are presented by the exponential increase in mobile data traffic. Via a CRAN-based framework, the proposed system aims to optimize the delay, throughput, and application response time. The presented strategies for resource management in 5G HetNets with cloud management target offloading and scheduling algorithms that are dynamic and computationally efficient.

The proposed novel CRAN framework is illustrated in Figure 1. It is characterized by a heterogeneous-access architecture composed of macrocell (managed by High-RRHs or H-RRHs) and small cells (managed by Low-RRHs or L-RRHs). It also introduces an edge-cloud infrastructure, namely, Edge Cloud-RRH. It is connected to the H-RRHs using high-speed fibers links or alternatively high-speed radio links.

In traditional CRANs, all RAN functionalities are centralized in BBU pools. In this work, we propose a flexible division of the functionalities between edge and central

(core) clouds. We further consider additional computational and storage resources in the Edge Cloud-RRHs to be utilized for computational offloading of user applications.

B. VIRTUAL MACHINES VS. CONTAINERS

In this section, we motivate the use of containers over virtual machines (VMs) in our architecture. Figure 2 serves as a reference for the argument.

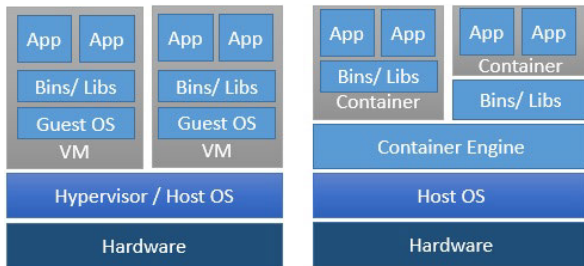


FIGURE 2. VM vs. container virtualization architecture.

Machine virtualization requires that the full guest OS (operating system) images are instantiated in each deployed VM. In addition, all binaries (Bins) and libraries (Libs) are necessary for running user applications. Such requirements result in slow VM initialization and sometimes degradation of performances.

Meanwhile, containers represent a virtualization solution that is flexible and lightweight. Recall that containers are packaged, self-contained, ready-to-deploy parts of applications that may, if necessary, include middleware and business logic in the form of binaries and libraries that are needed for running user applications [25], [32], [34]. Containers a) have a lightweight, portable runtime; b) facilitate the development, testing, and deployment of applications to many servers; and c) facilitate the interconnection between applications.

In current datacenters, the control of VMs entails the use of a virtual infrastructure manager (VIM) to oversee the VM lifecycle. Similarly, we introduce in our solution, a new functional entity, namely, the containers' manager (CM), for overseeing container placement/deployment, container monitoring, and application scheduling.

Using the CM, mobile users can access their services directly in the edge cloud by requesting the CM to instantiate them in the edge and request an offloading (in part or in whole) of the service logic computation from the far cloud computing into these containers. Furthermore, containers need not to be always active; rather, they can be activated or deactivated as needed. These different interaction schemes are illustrated in Figure 3. End-user terminal can trigger an offloading request (Off Req) to the CM (in red). The CM schedule this request and execute it. It then sends back offloading responses (Off Res) to the CM (in green).

C. PROBLEM DEFINITION

In conventional architectures, one BBU is logically associated with one RRH. Therefore, radio resources may

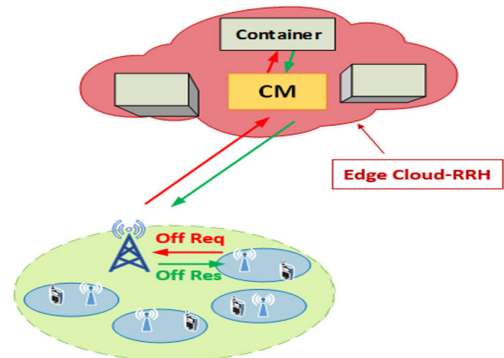


FIGURE 3. Containers' Manager Interactions.

be underutilized since one RRH may not consume all the resources of the BBU but cannot be used by other RRH. Moreover, this one-to-one assignment is very expensive for 5G networks, where a massive number of cells is deployed. Cloud RAN architecture resolves this problem by allowing a one-to-many mapping between BBUs and RRHs. Thus, we propose to efficiently cluster RRHs in order to allocate a pool of BBU to each cluster and allow them to share the same pool of resources. This will eventually improve the utilization of the resources, reduce the power consumption, and maintain the same quality of service level as the one-to-one mapping case with an appropriate clustering and scheduling.

In the proposed architecture, the mobile application may be moved to the Edge Cloud-RRH infrastructure and executed as tasks. This will reduce the response time of the applications and eventually increase the end-user experience. The CM is responsible for deciding which application tasks are to be executed. In this scheme, a container is characterized by the three-tuple (CPU, RAM, and network bandwidth). We consider that the request from the end-users equipment is composed of a set of tasks to instantiate in the Edge Cloud-RRH. Each task is associated with its execution delay constraint and resource requirements in terms of CPU, RAM and network bandwidth.

The CM should execute a scheduling algorithm to accommodate a substantially important number of offloading requests. The following section addresses two objectives when designing the scheduler:

a) the scheduler should consider the overloading and migration costs (minimization) when selecting the right container to execute the application task.

b) the schedule should take into account the level of load of each Edge Cloud-RRH to achieve a load balancing between them and achieve better scalability.

IV. RRM STRATEGIES FOR EDGE CLOUD-RRH WITH QOS AND COST OPTIMIZATION

A. RRH CLUSTERING STRATEGY

First of all, we propose a two-stage control loop for our heterogeneous C-RAN system. The first loop involves the decision of L-RRH clustering, while the second loop involves

the decision of H-RRH clustering. The decision-making process of the first loop should be aware of intra- and inter-cluster interference and traffic load of all base stations in order to define a clustering schema that minimizes these interferences. Furthermore, the second loop's decision-making process should be aware of system interference and BBU capacity. The global objective is eventually to enhance the QoS, reduce the resource utilization and power consumption.

1) INTERFERENCE COORDINATION MODEL

In our architecture, end-user terminals can be either connected to L-RRH or H-RRH. The user terminals are connected to the RRH that provides them the best SINR.

When a user i is associated with L -RRH j , the received signal to noise and interference ratio is formulated by the following expression [5], [44]:

$$SINR_{i,j} = \frac{|h_{i,j}w_{i,j}|^2 p_{i,j}^r}{\sum_{r \in C_{L/j}} |h_{i,r}w_{i,r}|^2 p_{i,r}^r + \sum_{r \in C_r/C_L} p_{i,r}^r + \sigma^2} \quad (1)$$

where C_L is the RRH cluster containing L -RRH j , C_r are other L-RRH clusters which don't contain L -RRH j , $h_{i,r}$ is the channel vector of user i served by L -RRH r , $w_{i,r}$ is the precoding weight between user i and RRH r , $p_{i,r}^r$ is the corresponding transmission power, $p_{i,r}^r$ is the received power at user i from L -RRH r and σ^2 is the noise. We assume $h_{i,r}$ the channel between a user i and a L -RRH r to be distributed as a Rayleigh random variable and counting pathloss. The first interference term, $\sum_{r \in C_{L/j}} |h_{i,r}w_{i,r}|^2 p_{i,r}^r$, stands for the intra-cluster interference. It depends on the coordination strategy used within this L-RRH cluster. The second term, $\sum_{r \in C_r/C_L} p_{i,r}^r$, stands for the inter-cluster interference.

For L-RRH clustering, we will consider both intra-cluster and inter-cluster interference. However, for H-RRH clustering, we will assume zero-forcing (ZF) precoding. Therefore, intra-cluster interferences are eliminated and the SINR, when a user i is associated with H -RRH j , is formulated as follows:

$$SINR_{i,j} = \frac{p_{i,j}^r}{\sum_{k \in C_k/C_H} p_{i,k}^r + \sigma^2} \quad (2)$$

where C_H is the RRH cluster containing H -RRH j , C_k are other H-RRH clusters that don't contain H -RRH j .

2) ENERGY MODEL

Besides radio performance improvement, Cloud RAN architecture enables high energy savings, essentially thanks to efficient radio network hardware utilization. In order to estimate energy saving, we adopt an energy consumption model where the BBU and RRH power consumption are measured separately [35]. The total power consumed by a BBU at time t is formulated by the following equation:

$$P_{BBU_t} = P_{BB} + \sum_{i=1}^n P_{RRH_i} \quad (3)$$

where P_{BB} is energy consumed by the BBU and $\sum_{i=1}^n P_{RRH_i}$ is the sum of energy consumed by all RRHs allocated to this BBU at instant t . The power consumed by a RRH is given by the following expression:

$$P_{RRH} = \frac{P_r}{\eta_{PA}} + (P_{RF} \times N_{TRX}) \quad (4)$$

where P_r is the radiated power, η_{PA} is the power amplifier efficiency, P_{RF} is the power consumed by radio frequency circuits, and N_{TRX} is the number of transceiver antennas.

The total energy consumed by a BBU pool is given by the following expression:

$$P_{BBU-Pool} = \sum P_{BBU} + P_I \quad (5)$$

where $\sum P_{BBU}$ is the sum of the energy consumed by all active BBUs in the pool and P_I is the energy consumed by the common infrastructure of the site which includes cooling systems, backhaul transmission equipment, lights, AC/DC converters, site monitoring systems and possibly site access control systems. This energy is expressed as follows:

$$P_I = P_{cooling} + P_{backhaul} + P_{lighting} + P_{monitoring} \quad (6)$$

3) L-RRHS CLUSTERING

For L-RRHs clustering (first stage of our clustering mechanism), we propose to use a fuzzy logic controller (FLC) in order to reduce uncertainties in clusters creation and reduce overhead while considering a multitude of parameters.

FLC is composed of three steps as illustrated in Figure 4. The first is the fuzzification. It consists of mapping each crisp input into a fuzzy variable. In our work, we will consider three variables which are SINR, load level of L-RRH, and load level of H-RRH. We chose to use the load level as the number of used RB (Resource Blocks). We divide each variable into three fuzzy levels: low, medium and high. Mapping is done using a membership function which defines the membership degree of the crisp input with the fuzzy variable. In our case, we will apply a triangular membership function, giving accurate and exact values. Membership functions for SINR H-RRH load and L-RRH load are represented in Figure 5.

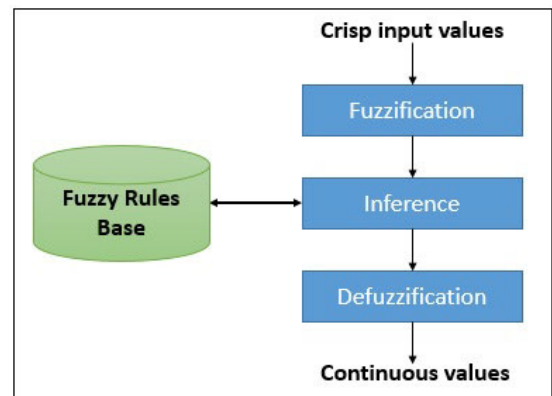


FIGURE 4. Fuzzy logic control proces.

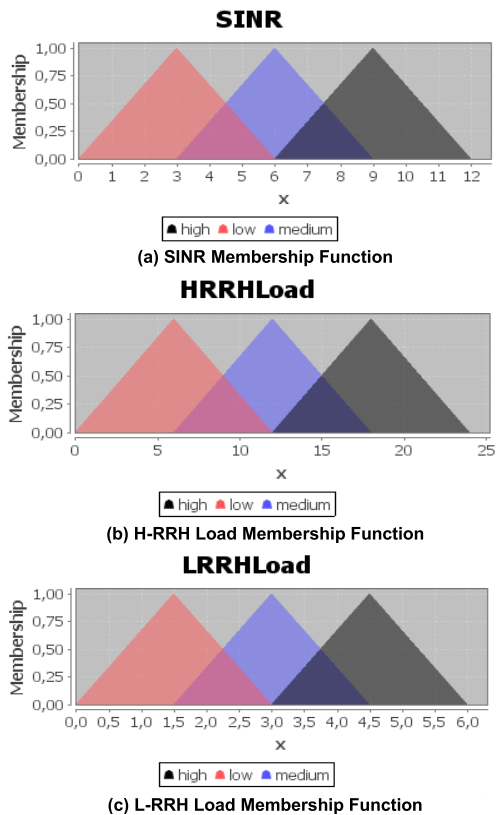


FIGURE 5. Membership functions.

In the second step, inference is modelled as a set of fuzzy rules. We have specified and applied 27 rules that are summarized in Table 1. In defuzzification step, the output action which is denoted here by the probability that a particular L-RRH is added to the cluster C, is given by the gravity center of conclusions c_l in each rule weighted by the membership function using the following expression:

$$a(x_1, x_2, x_3) = \frac{\sum_{l=1}^{27} \mu_l(x_1)\mu_l(x_2)\mu_l(x_3) c_l}{\sum_{l=1}^{27} \mu_l(x_1)\mu_l(x_2)\mu_l(x_3)} \quad (7)$$

4) H-RRHS CLUSTERING

H-RRH clustering objective is to allow obviously a better resource utilization by avoiding one-to-one mapping between BBUs and H-RRHs. The aim of clustering is to reduce the number of used BBUs in order to decrease the network energy consumption.

We propose here to model the problem as an optimization problem under resources' constraints. We consider a system composed of a set of n H-RRHs, each with an average RBs demand (d_1, \dots, d_n) and m user terminals (u_1, \dots, u_m). We consider that each BBUs has the capacity to simultaneously process K RBs.

The optimization problem consists of finding the minimum number of BBUs $B \leq n$ and a B -partition $Cl_1 \cup Cl_2 \cup \dots \cup Cl_B$ of the set of n H-RRHs such that $\sum_{i \in Cl_k} d_i \leq K$

TABLE 1. FLC rules for L-RRHs clustering.

Rule	SINR	L-RRH load	H-RRH load	Cluster membership
1	Low	Low	Low	Out
2	Low	Low	Medium	Out
3	Low	Low	High	Out
4	Low	Medium	Low	Out
5	Low	Medium	Medium	Out
6	Low	Medium	High	Out
7	Low	High	Low	Out
8	Low	High	Medium	Out
9	Low	High	High	Out
10	Medium	Low	Low	In
11	Medium	Low	Medium	In
12	Medium	Low	High	Out
13	Medium	Medium	Low	In
14	Medium	Medium	Medium	In
15	Medium	Medium	High	Out
16	Medium	High	Low	Out
17	Medium	High	Medium	Out
18	Medium	High	High	Out
19	High	Low	Low	In
20	High	Low	Medium	In
21	High	Low	High	Out
22	High	Medium	Low	In
23	High	Medium	Medium	In
24	High	Medium	High	Out
25	High	High	Low	Out
26	High	High	Medium	Out
27	High	High	High	Out

(each BBU cannot process a sum of demands higher than its capacity).The solution is optimal when B is minimal.

We introduce two binary variables $x_{(i,j)}$ to indicate if a H-RRH is added to a cluster or not:

$$x_{(i,j)} = \begin{cases} 1 & \text{if } H - RRH i \text{ is added to cluster } j \\ 0 & \text{otherwise} \end{cases}$$

And b_i to indicate if a BBU is active or not:

$$b_i = \begin{cases} 1 & \text{if a BBU is active} \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{cases} 1 & \text{if } H - RRH i \text{ is added to cluster } j \\ 0 & \text{otherwise} \end{cases}$$

The linear programming formulation of the problem is as follows:

$$\text{Minimize } B = \sum_{i=1}^n b_i$$

$$\text{Subject to } \sum_{j=1}^n d_j x_{(i,j)} \leq K b_i, \quad \forall i \in \{1, \dots, n\} \quad (8)$$

$$\sum_{i=1}^n x_{(i,j)} = 1, \quad \forall j \in \{1, \dots, n\} \quad (9)$$

The optimization is subject to constraints given by (8) and (9). Constraint (8) guarantees that BBU maximum capacity is not exceeded, i.e., the sum of loads from the cluster

of H-RRHs associated with a BBU does not exceed the BBU peak capacity. Constraint (9) ensures the fact that each H-RRH is associated with exactly one cluster.

H-RRH clustering is formulated as a modified one-dimensional bin packing problem. Since the bin packing problem is NP-hard [10], it is hard to find an optimal solution in large network systems. Therefore, we propose a heuristic algorithm to cluster H-RRHs. The proposed algorithm is inspired from Best Fit Decreasing (BFD) and Worst Fit Decreasing (WFD) strategies [10], however, it has the particularity to fill all unused resource blocks and reduce the number of BBUs.

The proposed heuristic algorithm is specified in the following Algorithm 1. R stands for the set of H-RRHs r_i , D is the set of their demands d_i expressed in term of the number of requested RBs per frame (sum all terminals demands in the same cluster) and C represents the set of remaining candidate H-RRHs to cluster (C is initialized to R). The proposed heuristic algorithm implements the following steps:

- Sort the set of demands D in a decreasing order.
- Select the H-RRH r_h with the highest demand and form cluster Cr_h .
- Form the set R' of H-RRHs $r_n \in R' \subset R$ that can be clustered with r_h and the set D' of their associated demands $d_n \in D' \subset D$.
- Select from R' the H-RRH r_s with the smallest RB demand in order to first be clustered with r_h , subject to $r_h + r_s \leq K$, where K is the maximum capacity of a BBU expressed in the number of available RBs.
- Add r_s to Cr_h and update its cluster $r_{h \cup s}$ with its demand $d_{h \cup s}$.
- If $r_h + r_s \leq K$ is not satisfied, r_h to which it belongs cannot be extended and the algorithm repeats with the following H-RRH that comes straight below r_h .
- The algorithm is performed until no more clustering can be performed.

Algorithm 1 H-RRH Clustering Algorithm

1. **Initialize** d_i
 2. **Attribute** d_i to r_i
 3. **Initialize** $C = R$
 4. **Repeat**
 5. Sort $d_i \in D$
 6. Select $d_h = \max d_i$ and its associated r_h
 7. **Form** $D' \subset D$ and $R' \subset R$
 8. Select $d_s = \min d_n / d_n \in D'$
 9. **If** $d_h + d_s \leq K$ **then**
 10. $r_{h \cup s} \leftarrow r_h \cup r_s$
 11. $d_{h \cup s} \leftarrow d_h + d_s$
 12. Update D, R and C
 13. **Else**
 14. $C \leftarrow C - \{r_h\}$
 15. Select $d'_h = \max d_i$
 16. $r_h \leftarrow r'_h$
 17. Go to step 7
 18. **Until** $C = \emptyset$
-

When this algorithm is executed, it results in a set of clusters of H-RRHs that are grouped together in clusters, each being served by the same BBU. The number of assigned BBU is eventually lower than the number of RRHs and a sum of demands in each cluster lower than the maximum capacity of a BBU. This eventually allows saving the energy consumption of all the inactivated BBUs while maintaining the QoS of the end-users.

B. OFFLOADING STRATEGY

Regarding the offloading strategy, the general principle is illustrated in Figure 6.

When a Mobile Terminal should execute a heavy application task, it takes a decision regarding the local execution of the task or its possible offloading. Examples of criteria maybe time-delay, energy-saving, end-user policy, etc. If the decision is to offload the task, it triggers an offloading request to the serving RRH.

The offloading request specifies a six-tuple, namely, $(TaskID, S_{TD}CAP_{MT}, CAP, E_{loc}, B)$, where S_{TD} is the size of the task, CAP_{MT} is the mobile terminal's capacity, CAP is the capacity that is required by the received task, E_{loc} is the energy that is expended for local execution and B is the bandwidth that is needed between the RRH and the mobile terminal.

The Containers' Manager (as shown in Figure 6), processes the offloading request and if granted, it sends a resource allocation request to the serving container to specify the required capacity after receiving the task code. Once processed, an offloading response will be routed to the mobile terminal. The Cloud RAN maintains a set of active containers to execute end-users tasks at the edge of the network. The CM maintains an up-to-date state of the load of the containers in order to schedule the execution of the end-users tasks depending on their size and their deadlines. It also performs load balancing between active containers for better use of the resources.

The bandwidth of the uplink connection between the mobile terminal and the serving RRH is assumed to be B . Following the notation in [30], [33], we denote the time that is required for transmitting S_{up} bits in uplink with rate r_{up} by t_{up} , namely, $t_{up} = S_{up}/r_{up}$. Similarly, for the downlink transmission requirement for remote processing from the Edge Cloud-RRH to the serving RRH, $t_{dl} = S_{dl}/r_{dl}$.

For power consumption at the mobile terminal in the uplink and the downlink, we adopt the following models:

$$p_{up} = k_{(tx,1)} + k_{(tx,2)} \cdot p_{tx} \quad (10)$$

$$p_{dl} = k_{(rx,1)} + k_{(rx,2)} \cdot p_{rx} \quad (11)$$

In the above, $k_{(tx,1)}$, $k_{(tx,2)}$, $k_{(rx,1)}$ and $k_{(rx,2)}$ are constants that are evaluated based on measurements that are provided in [27], [32]. Respectively, the constants correspond to the uplink baseline power, the uplink power that is associated with the transmission, the downlink baseline power, and the downlink power that is associated with receiving data.

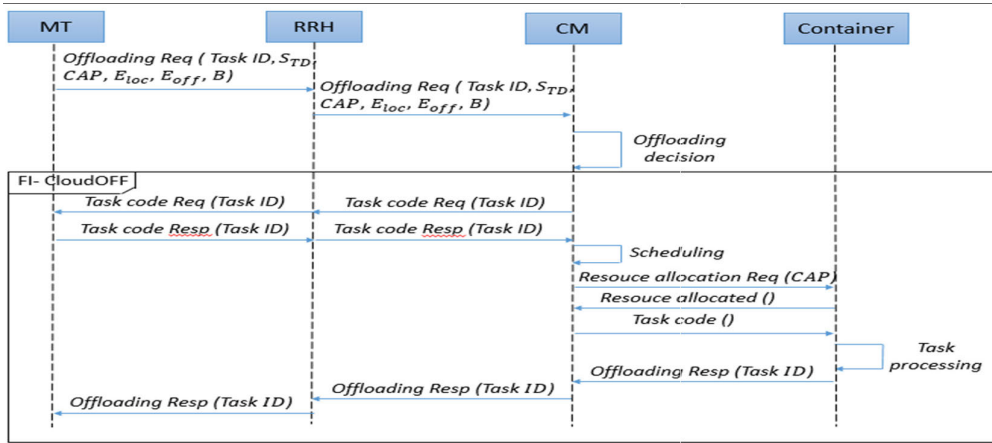


FIGURE 6. Cloud RAN: offloading mechanism.

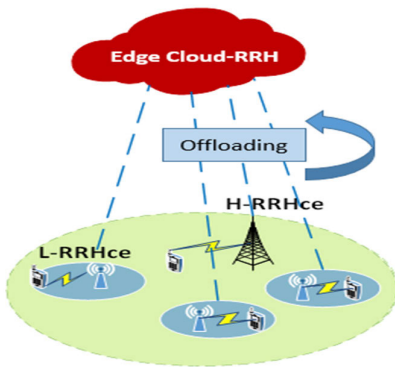


FIGURE 7. Offloading to Edge Cloud-RRH.

Shannon’s theorem identifies the maximum possible rate for a channel with K users:

$$r_{(up,k)} = B \cdot \log (1 + G_{up} \cdot p_{(tx,k)}) \quad (12)$$

$$r_{(dl,k)} = B \cdot \log (1 + G_{dl} \cdot p_{(tx,RRHce)}) \quad (13)$$

In the above, G_{up} and G_{dl} represent the channel gains that are normalized by the noise power in the uplink and downlink, respectively, while $p_{(tx,k)}$ and $p_{(tx,RRHce)}$ are the transmission powers of the user and the serving RRH.

Considering equations (10) and (11), the energy that is expended by a mobile terminal in the uplink and downlink can be computed as follows:

$$E_{up} = k_{(tx,1)} \cdot t_{up} + k_{(tx,2)} \cdot t_{up} \cdot p_{tx} \quad (14)$$

$$E_{dl} = k_{(rx,1)} \cdot t_{dl} + k_{(rx,2)} \cdot t_{dl} \cdot p_{rx} \quad (15)$$

According to (12), $p_{tx} = \frac{r_{up}}{G_{up} \cdot B} - 1$. Thus, the energy that is consumed by a mobile terminal during offloading can be computed as follows:

$$E_{off} = E_{up} + E_{dl} \quad (16)$$

Since the energy that is consumed by the mobile terminal in local processing is proportional to the number of processed

bits, E_{loc} can be computed as follows:

$$E_{loc} = \epsilon_0 \cdot S \quad (17)$$

where ϵ_0 is a constant that accounts for both the Joules/cycle and the cycles/bit for the processor at the mobile terminal and S is the number of bits.

For latency, we considered t_0 and t_1 as the times that are needed to process one bit at the mobile terminal and at the serving RRH, respectively. The time that is required for the offloading process to be completed is equal to the sum of the following elements: the time that is required for sending the bits from the mobile terminal to the serving RRH via the uplink; the time for the remote processor to execute the offloaded computation; and the time for sending all the output bits via the downlink. Thus, the expressions for the latency for the local processing and offloading are as follows:

$$L_{loc} = t_0 \cdot S \quad (18)$$

$$L_{off} = t_{up} + t_1 \cdot S + t_{dl} \quad (19)$$

C. OFFLOADING ALGORITHM

The main objective of this algorithm is to determine whether an application should be processed locally or must be offloaded to the Edge Cloud-RRH. The objective of the decision is to enhance the user’s quality of experience (QoE) while optimizing the use of resources in both the network and at the mobile terminal.

To facilitate the decision-making, we introduce a set of parameters to be utilized without increasing the solution’s complexity to the formulated optimization problem. The algorithm, which is illustrated in Figure 8, is applied at each time slot to process the set of tasks that are generated by the launched applications.

The algorithm starts by examining channel conditions. Computing the channel capacity using Shannon’s theorem (as described above), the channel coefficient is compared to the average channel coefficient, which is calculated and updated over time. If the current channel realization exceeds

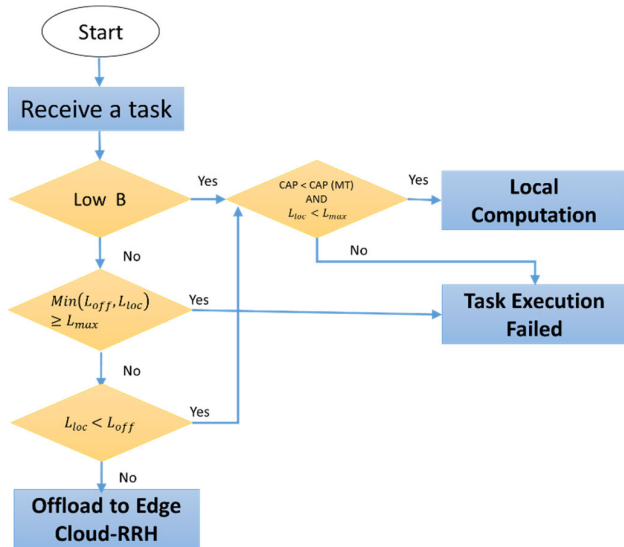


FIGURE 8. Proposed offloading decision algorithm for CRANs.

the computed average, the channel is considered to be in a relatively “good” state and the task can be offloaded. Otherwise, the MT will try to process the application locally. However, mobile terminal resources are limited in terms of computational capacity and latency generated by local execution should not exceed the maximum latency. Accordingly, if the required computational capacity exceeds the predefined percentage of the total locally available capacity or latency condition is not verified, the task execution is filed. Otherwise, the task is executed locally at the MT;

Next, latencies are compared using equations (18) and (19). If the minimum between latency that is generated when the task is offloaded to the Edge Cloud-RRH and that generated by local execution is higher than the maximum latency authorized by the application, the task execution is failed.

If not, the algorithm decides whether the latency that is generated by offloading is higher. If so, it verifies that the MT has the necessary resources to execute the task before deciding to execute it locally. Otherwise, the task is offloaded to Edge Cloud-RRH.

D. SCHEDULING OPTIMIZATION STRATEGY

The second strategy that is proposed in this work is concerned with scheduling while optimizing both the load balance and the execution time. This strategy, which is illustrated in Figure 9, proceeds as follows.

First, the resource monitor collects the information about container resources and sends a report to the load evaluator. This information is used to compute a score for each container, which is used to determine whether a container is overloaded or not. The load on a container can be defined by CPU usage, RAM usage, and network bandwidth utilization. All three resources should be considered uniformly when checking a container load [28], [33]. Otherwise, three cases are possible: CPU overloading, memory overloading, and network overloading.

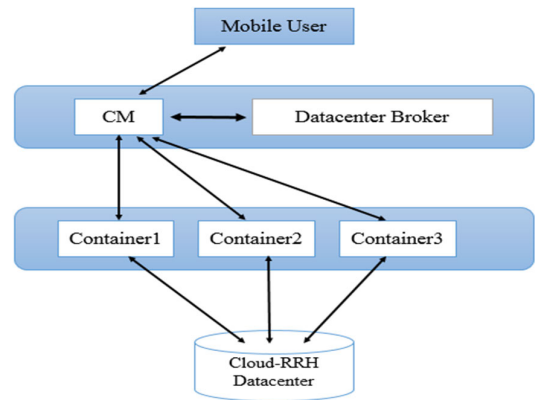


FIGURE 9. Scheduling Stages in Edge Cloud-RRH.

A container may not be using its network bandwidth or RAM memory at full capacity. However, the number of processes that are running on it may be overloading the CPU, thereby resulting in an average CPU usage that exceeds the number of cores in the machine. This corresponds to the case of CPU overloading.

Similarly, a container may not be using the full capacity of its CPU and memory but may have more connections and congested bandwidth. This corresponds to the case of network overloading.

Finally, memory overloading can be detected via swap activities [29]. By tracking the memory utilization and whether the swapping activities exceed a threshold, memory overloading can be detected.

Assuming uniform consideration of the metrics (CPU, memory and network), a score can be calculated as follows [29]:

$$score = \frac{1}{1 - CPU} \cdot \frac{1}{1 - net} \cdot \frac{1}{1 - mem}$$

where CPU, net, and mem represent the respective utilizations on the container, which are expressed as percentages of maximum capacity. If one resource is highly utilized, the overall volume would be considered as high. By using this approach, the above volume formula can be used to calculate a score for comparison against a threshold for determining whether a container is overloaded or not.

The structure of the containers’ manager that utilizes this score is illustrated in Figure 10, and the algorithm that is used to evaluate the loads is detailed in Algorithm 2.

Algorithm 2 returns to step 1 every 30 seconds to auto-organize Containers list formation until the container resource utilization changes with time. Thus, our approach considers system dynamics.

Finally, the dynamic scheduler uses the best-fit list that is generated by the load evaluator and the execution time to allocate the offloading request to a container. The dynamic allocation algorithm is presented as Algorithm 3.

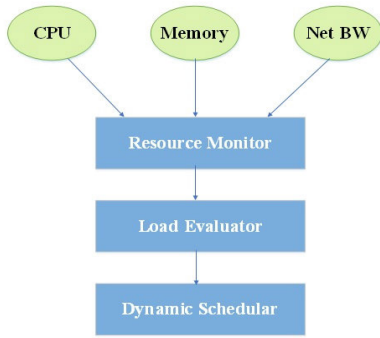


FIGURE 10. Container manager structure.

If all activated containers are overloaded and the best-fit list is empty, a new container is created and the offloading request is assigned to it.

Algorithm 2 Load Evaluation Algorithm

Input: Container load metrics
Output: BestFit_list
C_list: Containers that are active
Container_score: $\frac{1}{1-CPU} * \frac{1}{1-mem} * \frac{1}{1-net}$
Overload_list: list of containers that are overloaded
BestFit_list: list of containers that are suitable for offloading
Step1: For $i=1$ to number_C in C_list do
 calculate Container_score[i];
 If Container_score[i] > Threshold **then**
 add Container[i] to Overload_list;
 Else
 add Container[i] to BestFit_list;
 End if
End for
Step2: Sort BestFit_list in increasing order of Container_score
 Sleep (30 sec)
 return to Step1

E. STRATEGY FOR SCHEDULING BASED ON THE OVERLOAD COST

In this section, we elaborate on our proposed strategy that is based on the overload cost. This scheduling strategy considers the deployment cost, deadline, and available resources. This strategy aims to reduce the complexity of the offloading decision-making while enhancing the user computing experience.

Fulfilling the application time constraints, we assume that each Edge Cloud-RRH infrastructure can run N predefined containers. Each container is characterized by its available capacity resources, which are denoted by CPU_i , RAM_i and Net_i , where $i \in N$. An offloading request is specified as a set of M tasks to be executed with a deadline D , where each task is characterized by its requirements CPU_j , RAM_j and

Algorithm 3 Containers Dynamic Allocation Algorithm

Input: BestFit_list
Output: Container allocation
If BestFit_list $\neq 0$ **then**
 For $i=1$ to number_C in BestFit_list **do**
 Compute ExecutionTime_Container[i];
 $f(Container_i) \leftarrow ExecutionTime_Container[i]$;
 End for
 New_list \leftarrow Sort containers in ascending order of $f(Container_i)$;
 Affect Offloading_Request to New_list[1];
Else
 Wake a new Container;
 Affect Offloading_Request to Container_new;
End if

Net_j and has an expected execution time $T_{ex,j}$, $j \in M$. This is the execution time if all resource requirements are met. We further consider a binary variable $t_{(i,j)}$, which indicates whether a task has been allocated to a container.,

$$t_{(i,j)} = \begin{cases} 1 & \text{if task } j \text{ is allocated to container } i \\ 0 & \text{otherwise} \end{cases}$$

A cost C is associated with each container-to-task assignment, the value of which depends on whether the container is overloaded after the execution of the task and whether the task migration was necessary (due to user mobility). (As discussed previously, the monetary costs of energy consumptions are not considered in this work.) The considered costs are specified in detail as follows:

Denote the computational capacity of container i at time t by C_{cap} . Accordingly,

$$C_{cap_i} = \begin{pmatrix} C_{cap_i}^{CPU} \\ C_{cap_i}^{RAM} \\ C_{cap_i}^{Net} \end{pmatrix} \tag{20}$$

Denote the average cost of resource utilization of task j on container i by $C_{ut_{j,i}}$. This cost can be computed as follows:

$$C_{ut_{j,i}} = \begin{pmatrix} C_{ut_{j,i}}^{CPU} \\ C_{ut_{j,i}}^{RAM} \\ C_{ut_{j,i}}^{Net} \end{pmatrix} \tag{21}$$

We can further define the utilization rate μ_i of container i that corresponds to the system configuration to be computed as follows:

$$\mu_i = \begin{pmatrix} \mu_i^{CPU} = \frac{\sum(t_{(i,j)} \cdot C_{ut_{j,i}}^{CPU})}{C_{cap_i}^{CPU}} \\ \mu_i^{RAM} = \frac{\sum(t_{(i,j)} \cdot C_{ut_{j,i}}^{RAM})}{C_{cap_i}^{RAM}} \\ \mu_i^{Net} = \frac{\sum(t_{(i,j)} \cdot C_{ut_{j,i}}^{Net})}{C_{cap_i}^{Net}} \end{pmatrix} \tag{22}$$

The container is overloaded if $\max(\mu_i^{CPU}, \mu_i^{RAM}, \mu_i^{Net}) > 1$. If a task j is allocated to an overloaded container, we associate a penalty that is positively proportional to the level of the overloading.

The overload cost metric, which is denoted ov_{cost} , is defined as follows:

$$ov_{cost_i} = \begin{cases} (\mu_i - 1)^\lambda & \text{if } \max(\mu_i^{CPU}, \mu_i^{RAM}, \mu_i^{Net}) > 1 \\ 0 & \text{otherwise} \end{cases} \quad (23)$$

Then, the total overload cost for the Edge Cloud-RRH to execute all tasks can be calculated as follows:

$$ov_{cost_t} = \sum_{i=1}^N ov_{cost_i} \quad (24)$$

where N is the number of containers.

Migration costs are considered in accounting for user mobility. As a user moves from one cell to another in the network, the user's task can also be migrated. We associate a penalty r_j when a MT task j is migrated from one container to another one to capture the resulting service downtime that is incurred. The total migration cost can be computed as follows:

$$mig_{cost_t} = \sum_i \sum_j t_{(i,j)} \cdot r_j \quad (25)$$

Only migrations within the same Edge Cloud-RRH within the same provider are considered. The applied penalty is further dependent on the type of task that is migrated.

We now formulate the proposed strategy. Recall that the objective is to minimize the total task assignment cost while considering both the overload and migration costs in executing all offloading requests. The problem is formulated as follows:

$$\text{Minimize } \alpha \sum ov_{cost_t} + \beta mig_{cost_t} \quad (26)$$

$$\text{Subject to } \sum_j t_{(i,j)} \cdot T_{ex_j} \leq D \quad (27)$$

$$\sum_j t_{(i,j)} \cdot C_{ut_{j,i}^{CPU}} \leq CPU_i \quad (28)$$

$$\sum_j t_{(i,j)} \cdot C_{ut_{j,i}^{RAM}} \leq RAM_i \quad (29)$$

$$\sum_j t_{(i,j)} \cdot C_{ut_{j,i}^{Net}} \leq Net_i \quad (30)$$

$$\frac{(\sum_i \mu_i)^2}{N \cdot \sum_i \mu_i^2} \geq 1 \quad (31)$$

$$\sum_j t_{(i,j)} = 1 \quad (32)$$

The objective function in (26) aims at minimizing the total costs, with α and β used as weights to facilitate the implementation of preference. The objective is constrained by equations (27) through (32). Constraint (27) ensures that each offloading request is executed prior to the application's deadline, with D identifying the worst case, namely, the case in which all tasks are sequentially executed. Meanwhile, constraints (28-30) enforce that all tasks' requirements in terms of CPU, memory and network bandwidth are within the container capabilities. Constraint (31) guarantees load balancing

between containers in the same Edge Cloud-RRH. Finally, constraint (32) limits the assignment of each application to a single container. This problem is a MIP and, therefore, can be solved as a linear program since the objective function is linear with respect to all the variables.

V. PERFORMANCE EVALUATION

In this section, we describe the evaluation methodology and results for our proposed resource management strategies. We use CloudSim, which is a simulation framework for cloud-based scenarios. The simulator is used in tandem with a 5G HetNet CRAN RHH topology.

Three scenarios are considered, with each evaluating the performance of our strategies toward a single objective: offloading, scheduling, and cost.

A. SCENARIO FOR EVALUATING RRH CLUSTERING STRATEGY

This section provides simulation results for the proposed two-stage RRH clustering scheme. As a first step, the fuzzy logic controller was developed using jFuzzyLogic java library [36], [37]. We considered a heterogeneous C-RAN system where H-RRHs have coverage of 500m and a maximum capacity of 24 RBs and L-RRHs are 30m-radius [26]. To evaluate the energy consumed by the network, we used the power values listed in Table 2 [38].

TABLE 2. RRH clustering Simulation parameters.

Parameters	Values
Coverage of H-RRH	500m
Coverage of L-RRH	30m
K	24 RBs
S	16dB
Number of H-RRH	1000
P_{BB}	200W
$P_{cooling}$	2000W
$P_{backhaul}$	200W
$P_{lighting}$	50W
$P_{monitoring}$	50W

Figure 11 shows the considered normalized mean values of the traffic load given by considering loads of all spatially distributed H-RRHs within a 24-hour interval as introduced in [39].

Figure 12 shows the evolution of the average SINR versus the number of L-RRHs per H-RRH. Results show that while without using clustering, the QoS degrades with the increase of the density of RRHs, the proposed clustering scheme is able to improve the QoS with the increasing of L-RRHs density.

We have executed different simulations for different L-RRHs density conditions: sparse, medium, and dense scenarios with 20, 40 and 80 L-RRHs/H-RRH, respectively. We evaluated the network QoS expressed by average SINR

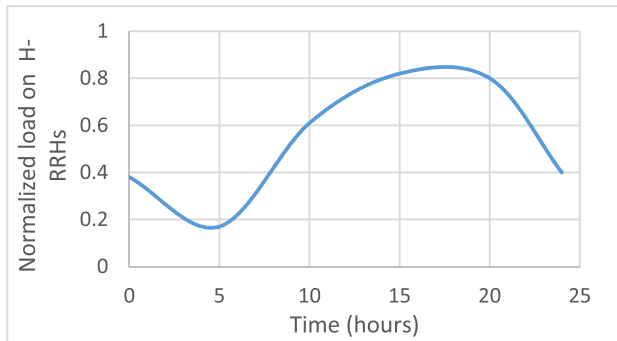


FIGURE 11. H-RRHs load variation.

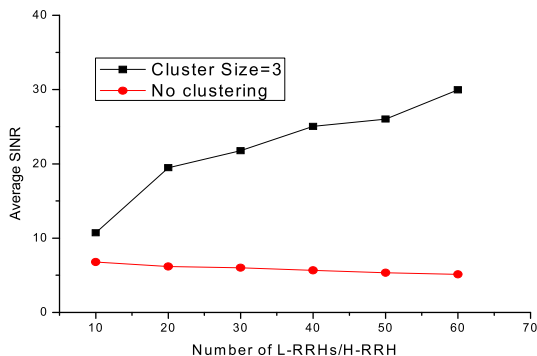


FIGURE 12. SINR variation as the number of active L-RRHs per HRRH.

for different clusters sizes. The results that are highlighted in Figure 13 show that presented scheme is more effective in the dense deployment scenario and bigger cluster size. Therefore, the proposed clustering mechanism can be deployed in high dense 5G C-RANs.

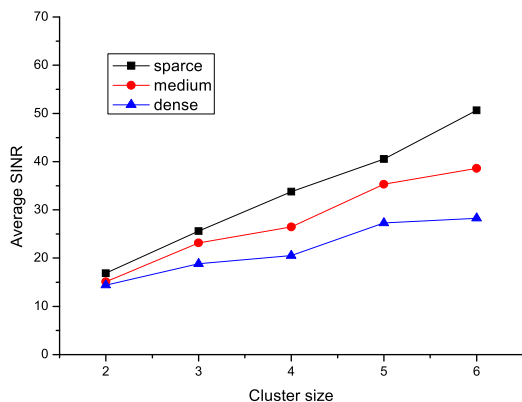


FIGURE 13. SINR variation as cluster size.

Figure 14 shows the variation in the average number of BBUs required to meet the C-RAN load of 1000 RRHs in the case of using FFD and proposed RRH clustering mechanism. In FFD (First Fit Decreasing), RRHs are ranked in descending order of their respective loads. Then, the first fit algorithm is applied to map the RRHs to the BBUs. We can see that the number of activated BBUs has been reduced through

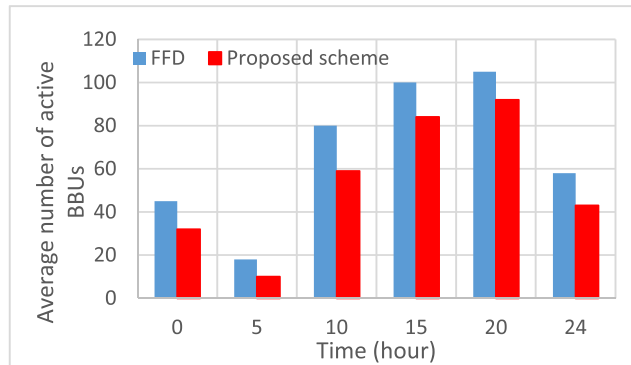


FIGURE 14. Variation of the number of active BBUs.

the use of our clustering mechanism. Therefore, the energy consumed by the network is reduced as shown in Figure 15.

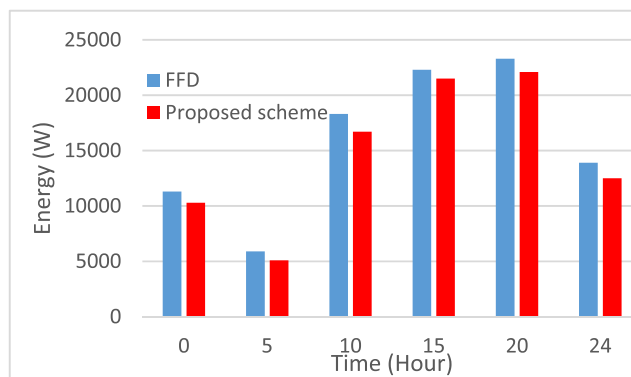


FIGURE 15. Network energy consumption.

B. SCENARIO FOR EVALUATING THE OFFLOADING STRATEGY

This scenario is used to evaluate our proposed offloading decision algorithm, especially with respect to the total offloading. We consider an urban environment with a heterogeneous CRAN in which each cell is comprised of seven H-RRH units and four L-RRH units. Per [26], the radial coverage is 500 m for an H-RRH unit and 30 m for an L-RRH unit. Other system simulation parameters that are relevant to this scenario are listed in Table 3. The values in the table for ϵ_0 and t_0 have been aligned with the measurements that are specified in [27] for the energy and frequency characteristics of localized computing in commercial handsets and for the computation of data ratios in practical applications.

First, we evaluate the application response time (in ms) while varying the data size (ranging from 1 to 100 kbits). The performance of the proposed scheme is compared to that of total offloading. The results (shown in Figure 16) demonstrate that the proposed offloading scheme improves the user experience by reducing the response time. If the data size is small, the proposed scheme and total offloading perform similarly. However, as the data size increases, the proposed scheme

TABLE 3. Simulation parameters to evaluate offloading strategy.

Parameter	Value
$k_{(tx,1)}$	0.4W
$k_{(tx,2)}$	18
$k_{(rx,1)}$	0.4W
$k_{(rx,2)}$	$2.86 \cdot 10^{-3}$ W/Mbps
ϵ_0	$8.6 \cdot 10^{-8}$ J/bit
t_0	10^{-7} s/bit
t_1	$t_0/2$
Bandwidth (B)	10 MHz
Maximum latency authorized by the application (L_{max})	4s

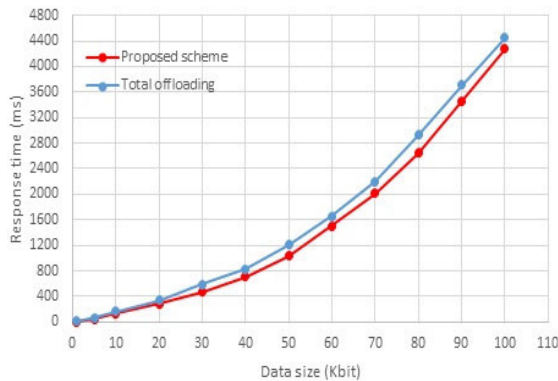


FIGURE 16. Application Response Time (in ms), with the Data Size Varied (from 1 to 100 kbits).

outperforms total offloading, with an average improvement of 14% in response time.

In Figure 17, we illustrate the results of evaluating the energy that is expended by the mobile terminal while varying the data size (from 1 to 100 kbits). According to the figure, the proposed offloading algorithm results in the mobile terminal expending less energy in our solution compared to the total offloading solution. As with the response time, the proposed scheme’s advantage becomes more apparent as the data size increases. On average, a 15% improvement in energy consumption is observed. Therefore, the proposed algorithm can augment the mobile handset battery lifetime while executing heavy applications.

C. SCENARIO FOR EVALUATING THE SCHEDULING STRATEGY

The proposed scheduling model for this strategy was implemented inside CloudSim as part of the cloud broker. The scenario entailed virtualizing ten datacenters; two to six hosts were created in each datacenter, with 2 GB RAM, 1 TB storage, and 10 GB/s bandwidth in connection. In addition, 50 containers and 100-1000 tasks were implemented under the simulation platform. The task length ranged from

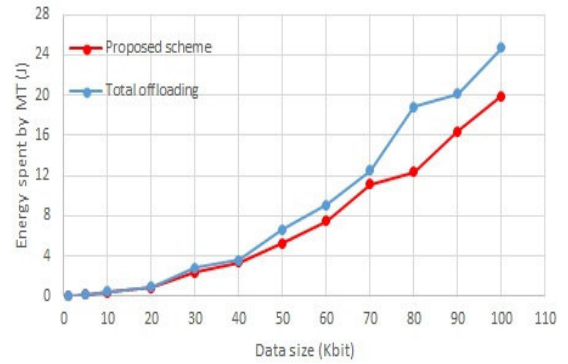


FIGURE 17. Total Consumed Energy by the MT, with the Data Size Varied (from 1 to 100 kbits).

1000 million instructions (MI) to 20000 MI. Other parameters that are relevant to the scenario are listed in Table 4 [30].

TABLE 4. Simulation parameters to evaluate the scheduling strategy.

Entity	Parameters	Value
Datacenter	Number of datacenters	10
	Number of hosts	2-6
	Container Scheduler	Space_shared and Time_shared
Container	Total number of containers	50
	MIPS	500-2000
	Memory (RAM)	256-2048
	Bandwidth	500-1000
	Cloudlet Scheduler	Space_shared and Time_shared
	Number of PEs requirement	1-4
Cloudlet	Total number of tasks	100-1000
	Length of task	1000-20000

In the scenario, we assume that tasks are mutually independent; namely, there is no constraint regarding the sequential processing of the tasks. We also assume that task processing does not undergo preemption, interruption, or migration between processors.

The performance of the proposed task scheduling model is evaluated via comparison with a round-robin (RR) scheduler and ACO [42]. We have implemented these three algorithms in the simulator. Figure 18 plots the average makespans of the three evaluating schedulers. As highlighted in this figure, the proposed scheduler outperforms ACO and RR.

Another basis of evaluation for the scheduling is the degree of imbalance, which is denoted as D_i , among the containers, and computed as follows:

$$D_i = \frac{T_{max} + T_{min}}{T_{avg}} \tag{33}$$

In the above equation, T_{max} , T_{min} , and T_{avg} are the maximum, minimum, and average execution times, respectively, of the containers’ tasks.

Figure 19 plots the average degree of imbalance of each algorithm as a function of the number of tasks, which is varied from 100 to 1000. According to the figure, the proposed task

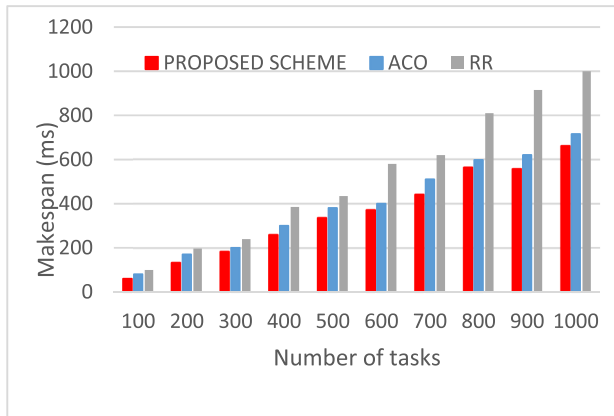


FIGURE 18. Average Makespan of the Proposed Scheme, ACO and RR.

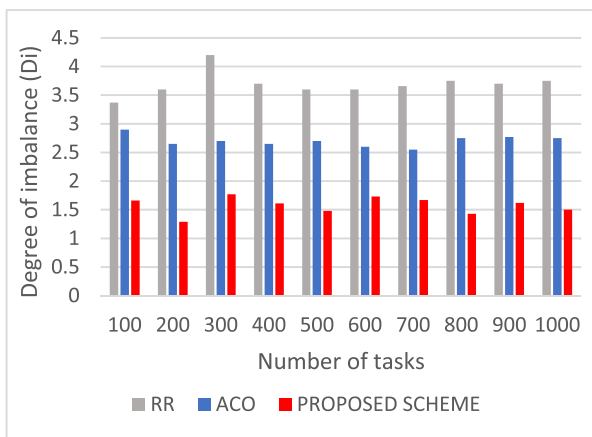


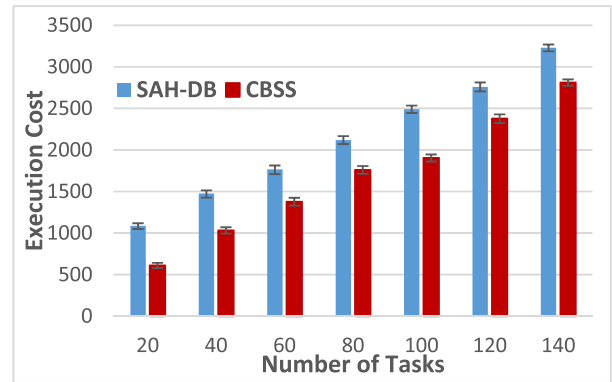
FIGURE 19. Average degree of imbalance.

scheduling model outperforms both RR and ACO algorithms in terms of the system load balance.

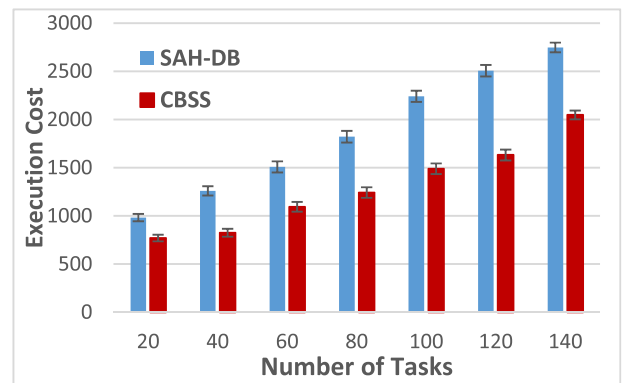
D. SCENARIO FOR EVALUATING THE STRATEGY FOR SCHEDULING BASED ON THE OVERLOAD COST

In this scenario, we consider an Edge Cloud-RRH with N heterogeneous containers, where $N = \{25, 50, 75, 100\}$. The computing capacity of the containers is varied between 1 and 10 CPUs, while the memory is varied from 128 Mbytes to 512 Mbytes. The network bandwidth is varied between 100 Kbps to 200 Kbps. The number of tasks M is varied in steps of 20 tasks, namely, $M = \{20, 40, 60, 80, 100, 120, 140\}$. Meanwhile, the task requirements are varied as follows: CPU between 1 to 4; memory between 128 and 1024 Kbytes; and bandwidth between 1 and 20 Kbps. Offloading requests are embedded sequentially, with requirements randomly varied. Simulation parameters that are relevant to this scenario are listed in Table 5. We also fix the values $\alpha = \beta = 0.5$ and $\lambda = 2$.

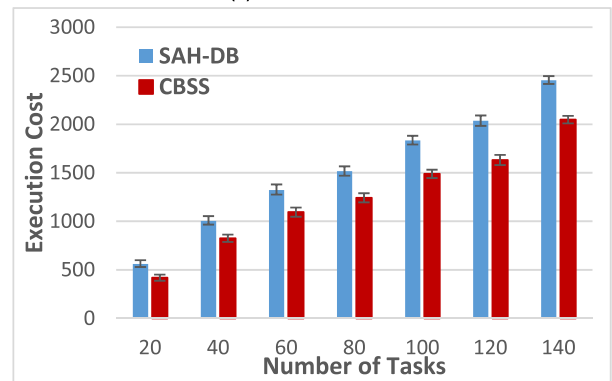
We compared the proposed cost-based scheduling scheme with the SAH-DB scheduling algorithm [24] (a task



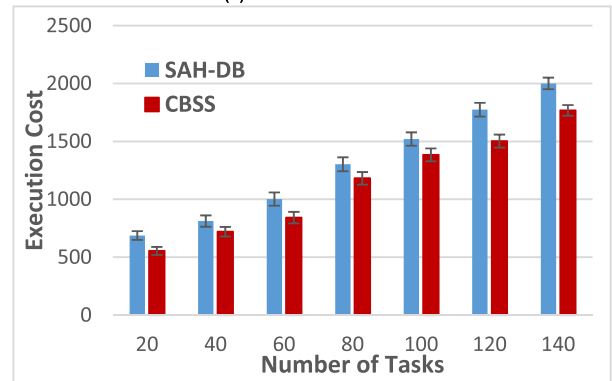
(a) Resources = 25



(b) Resources = 50



(c) Resources = 75



(d) Resources = 100

FIGURE 20. Execution Cost vs. the Number of Resources.

TABLE 5. Simulation parameters for the scenario that is used to evaluate the overload cost strategy.

Entity	Parameter	Value
Container	No. of containers	25 - 100
	CPU	1 - 10
	Memory (RAM)	128 - 512 Mbytes
	Network Bandwidth	100 - 200 Kbps
Task	Total No. of tasks	20 - 140
	CPU	1 - 4
	Memory (RAM)	128 - 1024 Kbytes
	Network Bandwidth	1 - 20 Kbps

scheduling algorithm that is based on a delay-bound constraint) for 25 to 100 cloud containers. For the proposed scheduler, we used IBM's linear programming solver CPLEX [31] to generate solutions for the formulated problem and multiple specified data inputs. The scheduling efficiency was evaluated in terms of the execution cost under a varying number of associated tasks. Figure 20 presents the results. The proposed scheduling algorithm can reduce the total execution cost compared with the SAH-DB algorithm for various numbers of associated tasks. Meanwhile, the total scheduling cost decreases with the increase of the number of resources and increases with the number of associated tasks.

VI. CONCLUSION

Cloud radio access networks (CRANs) have recently demonstrated considerable potential in mobile cellular networks, especially in terms of enhancing resource management and asset sharing. Our objective in this work was to introduce novel strategies that homogenize resource management, QoS evaluation, and cost optimization. The work involved three main aspects, which are described below.

First, we proposed a CRAN architecture that is based on a flexible RAN functionality division for 5G networks. We introduced the Edge Cloud RRH, which represents intermediate storage and computation capabilities. In order to enhance radio resource management and reduce energy consumption, we proposed a two-loop RRH clustering mechanism. The first loop used a fuzzy logic controller to decide about L-RRHs clustering. The second loop was dedicated to H-RRHs clustering. The problem was formulated as a bin packing problem and resolved using a heuristic algorithm. Simulation results showed that the proposed scheme is effective for high dense deployments and is able to reduce system energy consumption.

The architecture further entailed a division of functionalities between the central (core) and edge clouds. The objective of this division is to enhance resource utilization while improving network performance. The division makes additional resources available to the user, thereby enabling either total or partial offloading to the Edge Cloud-RRH. Overall, the offloading improves the user experience and enhances the energy efficiency.

Then, we introduced a decision framework for offloading. The framework utilizes an algorithm that considers several

key parameters in heterogeneous CRANs to employ our Edge Cloud-RRH for offloading. The algorithm optimizes the use of both network and mobile terminal resources and its efficiency is evaluated via numerical simulation. The simulation results demonstrate that the proposed algorithm reduces the task response time and the energy consumption in the mobile terminal.

Finally, we proposed a novel scheduling scheme for minimizing the computational scheduling cost in Edge Cloud-RRH that considers available resources, connection resource requirements, deadline, and load balancing. The considered scenario involves users offloading tasks to the Edge Cloud-RRH, and we focus on scheduling tasks that request several resources such as CPU, memory, and disk. The scheduling problem is formulated as a cost minimization problem in which user performance in terms of system overload and migration cost is considered. The scheme was evaluated via simulation; the results demonstrate that it can schedule offloading requests with minimal total execution cost.

In future work, we will consider the RRH clustering problem in the heterogeneous Cloud RAN architecture as a two-stage control loop. The first loop will use a fuzzy logic controller to identify the L-RRH clusters, and the second loop will target the H-RRH clusters. The problem will be formulated as a bin packing problem, and a heuristic will be employed to reduce the computational requirements of the solution.

REFERENCES

- [1] Cisco, "Visual networking index: Global mobile data traffic forecast update, 2015–2020," Cisco, San Jose, CA, USA, White Paper 12725, Feb. 2016.
- [2] K. Kumar, J. Liu, Y. H. Lu, and B. Bhargava, "A survey of computation offloading for mobile systems," *Mobile Netw. Appl.*, vol. 18, no. 1, pp. 129–140, 2013.
- [3] T. Kamarainen, Y. Shan, M. Siekkinen, and A. Yla-Jaaski, "Virtual machines vs. Containers in cloud gaming systems," in *Proc. Int. Workshop Netw. Syst. Support Games (NetGames)*, Dec. 2015, pp. 1–6.
- [4] K. Boulos, M. El Helou, and S. Lahoud, "RRH clustering in cloud radio access networks," in *Proc. Int. Conf. Appl. Res. Comput. Sci. Eng. (ICAR)*, Oct. 2015, pp. 1–6.
- [5] H. M. Soliman and A. Leon-Garcia, "QoS-aware joint RRH activation and clustering in cloud-RANs," in *Proc. IEEE Wireless Commun. Netw. Conf.*, Apr. 2016, pp. 1–6.
- [6] D. Mishra, P. C. Amogh, A. Ramamurthy, A. A. Franklin, and B. R. Tamma, "Load-aware dynamic RRH assignment in cloud radio access networks," in *Proc. IEEE Wireless Commun. Netw. Conf.*, Apr. 2016, pp. 1–6.
- [7] M. M. U. Rahman, H. Ghauch, S. Imtiaz, and J. Gross, "RRH clustering and transmit precoding for interference-limited 5G CRAN downlink," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2015, pp. 1–7.
- [8] X. Chen, N. Li, J. Wang, C. Xing, L. Sun, and M. Lei, "A dynamic clustering algorithm design for C-RAN based on multi-objective optimization theory," in *Proc. IEEE 79th Veh. Technol. Conf. (VTC Spring)*, May 2014, pp. 1–5.
- [9] Y. Du and G. de Veciana, "'Wireless networks without edges': Dynamic radio resource clustering and user scheduling," in *Proc. IEEE Conf. Commun. (INFOCOM)*, Apr./May 2014, pp. 1321–1329.
- [10] A. Khanafer, "Algorithms for mono- and multi-objective bin packing problems," Ph.D. dissertation, Dept. Fundam. Comput. Sci., Univ. Sci. Technol. Lille, Villeneuve-d'Ascq, France, 2010. [Online]. Available: <https://ori-nuxeo.univ-lille1.fr/nuxeo/site/esupversions/49ddd808-5557-41e2-bdfe-73b69e53264d>

- [11] E. Cuervo, A. Balasubramanian, D. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "MAUI: Making smartphones last longer with code offload," in *Proc. 8th Int. Conf. Mobile Syst., Appl., Services (MobiSys)*, New York, NY, USA, 2010, pp. 49–62.
- [12] S. Kosta, A. Aucinas, P. Hui, R. Mortier, and X. Zhang, "ThinkAir: Dynamic resource allocation and parallel execution in the cloud for mobile code offloading," in *Proc. IEEE INFOCOM*, Mar. 2012, pp. 945–953.
- [13] B.-G. Chun, S. Ihm, P. Maniatis, M. Naik, and A. Patti, "Clonecloud: Elastic execution between mobile device and cloud," in *Proc. 6th Conf. Comput. Syst. (EuroSys)*, New York, NY, USA, 2011, pp. 301–314.
- [14] C. Yuh-Shyan, H. Chih-Shun, J. Tong-Ying, and L. Hsin-Han, "An energy-aware data offloading scheme in cloud radio access networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Mar. 2011, pp. 1984–1989.
- [15] J. Oueis, E. C. Strinati, and S. Barbarossa, "Multi-parameter decision algorithm for mobile computation offloading," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2014, pp. 3005–3010.
- [16] L. Ting-Yi, L. Ting-An, H. Cheng-Hsin, and K. Chung-Ta, "Context-aware decision engine for mobile cloud offloading," in *Proc. IEEE Wireless Commun. Netw. Conf. Workshops (WCNCW)*, Apr. 2013, pp. 111–116.
- [17] W. Huijun, H. Dijiang, and S. Bouzefrane, "Making offloading decisions resistant to network unavailability for mobile cloud collaboration," in *Proc. 9th IEEE Int. Conf. Collaborative Comput., Netw., Appl. Workshar-ing*, Oct. 2013, pp. 168–177.
- [18] M. Kalyan and A. Mishra, "Application of selective algorithm for effective resource provisioning in cloud computing environment," *Int. J. Cloud Comput., Services Archit.*, vol. 4, no. 1, pp. 1–10, Feb. 2014.
- [19] B. Santhosh and D. H. Manjiah, "An improved task scheduling algorithm based on max-min for cloud computing," *Int. J. Innov. Res. Comput. Commun. Eng.*, vol. 2, no. 2, pp. 84–88, Apr. 2015.
- [20] S. J. Patel and U. R. Bhoi, "Improved priority based job scheduling algorithm in cloud computing using iterative method," in *Proc. 4th Int. Conf. Adv. Comput. Commun.*, Aug. 2014, pp. 199–202.
- [21] A. Thomas, G. Krishnalal, and V. P. Jagathy Raj, "Credit based scheduling algorithm in cloud computing environment," *Procedia Comput. Sci.*, vol. 46, pp. 913–920, 2015.
- [22] A. Khalili and S. M. Babamir, "Makespan improvement of PSO-based dynamic scheduling in cloud environment," in *Proc. 23rd Iranian Conf. Electr. Eng.*, May 2015, pp. 613–618.
- [23] Himani and H. S. Sidhu, "Cost-deadline based task scheduling in cloud computing," in *Proc. 2nd Int. Conf. Adv. Comput. Commun. Eng.*, May 2015, pp. 273–279.
- [24] M. Yingchi, X. Ziyang, P. Ping, and W. Longbao, "Delay-aware associate tasks scheduling in the cloud computing," in *Proc. IEEE 5th Int. Conf. Big Data Cloud Comput.*, Aug. 2015, pp. 104–109.
- [25] S. Soltesz, H. Pötzl, M. E. Fiuczynski, A. Bavier, and L. Peterson, "Container-based operating system virtualization: A scalable, high-performance alternative to hypervisors," in *Proc. 2nd ACM SIGOPS/EuroSys Eur. Conf. Comput. Syst.*, New York, NY, USA, 2007, pp. 275–287.
- [26] O. Muñoz, A. Pascual-Iserte, and J. Vidal, "Joint allocation of radio and computational resources in wireless application offloading," in *Proc. Future Netw. Mobile Summit*, Jul. 2013, pp. 1–10.
- [27] A. R. Jensen, M. Lauridsen, P. Mogensen, T. B. Sørensen, and P. Jensen, "LTE UE power consumption model: For system level energy and performance optimization," in *Proc. IEEE Veh. Technol. Conf. (VTC Fall)*, Sep. 2012, pp. 1–5.
- [28] S. Mour, P. Srivastava, P. Patel, H. Ram, B. N. Gohil, and D. Patel, "Load management model for cloud computing," in *Proc. 9th Int. Conf. Internet Technol. Secured Trans. (ICITST)*, Dec. 2014, pp. 178–184.
- [29] T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif, "Black-box and gray-box strategies for virtual machine migration," *Proc. 4th USENIX Conf. Netw. Syst. Design Implement.*, Cambridge, MA, USA, 2007, pp. 229–242.
- [30] A. Outtagarts, L. Roullet, B. Mongazon-Cazavet, and G. Aravinthan, "When IT meets telco: RAN as a service," in *Proc. IEEE/ACM 8th Int. Conf. Utility Cloud Comput. (UCC)*, Dec. 2015, pp. 422–423.
- [31] IBM. (2013). *IBM ILOG CPLEX Optimization Studio*. [Online]. Available: <http://www-03.ibm.com/software/products/it/ibmilogcplexoptistud>
- [32] O. Chabbouh, S. Ben Rejeb, Z. Choukair, and N. Agoulmine, "Offloading decision algorithm for 5G/HetNets cloud RAN," in *Proc. 24th Int. Conf. Softw., Telecommun. Comput. Netw. (SoftCOM)*, Sep. 2016, pp. 1–5.
- [33] O. Chabbouh, S. B. Rejeb, N. Agoulmine, and Z. Choukair, "Cloud RAN architecture model based upon flexible RAN functionalities split for 5G networks," in *Proc. 31st Int. Conf. Adv. Inf. Netw. Appl. Workshops (WAINA)*, Mar. 2017, pp. 184–188.
- [34] T. Rabia and O. Braham, "A new SDN-based next generation fronthaul interface for a partially centralized C-RAN," in *Proc. IEEE 32nd Int. Conf. Adv. Inf. Netw. Appl. (AINA)*, May 2018, pp. 393–398.
- [35] M. Khan, R. S. Alhumaima, and H. S. Al-Raweshidy, "Reducing energy consumption by dynamic resource allocation in C-RAN," in *Proc. Eur. Conf. Netw. Commun. (EuCNC)*, Jun. 2015, pp. 169–174.
- [36] P. Cingolani and J. Alcalá-Fdez, "JFuzzyLogic: A java library to design fuzzy logic controllers according to the standard for fuzzy control programming," *Int. J. Comput. Intell. Syst.*, vol. 6, no. 1, pp. 61–75, Jun. 2013.
- [37] P. Cingolani and J. Alcalá-Fdez, "JFuzzyLogic: A robust and flexible fuzzy-logic inference system language implementation," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, Jun. 2012, pp. 1–8.
- [38] *Reducing Energy Consumption in Access Networks Final Report Master of Engineering–Semantic Scholar*. Accessed: May 5, 2019. [Online]. Available: <https://paper/Reducing-Energy-Consumption-in-Access-Networks-Keating-McManis/c476122201a58a4507c5ed7fdec0dfdc94625493>
- [39] D. Mishra, P. C. Amogh, A. Ramamurthy, A. A. Franklin, and B. R. Tamma, "Load-aware dynamic RRH assignment in cloud radio access networks," in *Proc. IEEE Wireless Commun. Netw. Conf.*, Apr. 2016, pp. 1–6.
- [40] Q. Zhang, L. Gui, F. Hou, J. Chen, S. Zhu, and F. Tian, "Dynamic task offloading and resource allocation for mobile-edge computing in dense cloud RAN," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3282–3299, Apr. 2020, doi: [10.1109/JIOT.2020.2967502](https://doi.org/10.1109/JIOT.2020.2967502).
- [41] Y. Zhou, S. Ci, and Y. Yang, "Energy-aware joint clustering and scheduling for multicast beamforming in cloud-RAN downlink," *IEEE Wireless Commun. Lett.*, vol. 9, no. 4, pp. 461–464, Apr. 2020, doi: [10.1109/LWC.2019.2958929](https://doi.org/10.1109/LWC.2019.2958929).
- [42] M. A. Tawfeek, A. El-Sisi, A. E. Keshk, and F. A. Torkey, "Cloud task scheduling based on ant colony optimization," in *Proc. 8th Int. Conf. Comput. Eng. Syst. (ICCES)*, Nov. 2013, pp. 64–69, doi: [10.1109/ICCES.2013.6707172](https://doi.org/10.1109/ICCES.2013.6707172).
- [43] G. Zhang, F. Shen, Z. Liu, Y. Yang, K. Wang, and M.-T. Zhou, "FEMTO: Fair and energy-minimized task offloading for fog-enabled IoT networks," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4388–4400, Jun. 2019, doi: [10.1109/JIOT.2018.2887229](https://doi.org/10.1109/JIOT.2018.2887229).
- [44] J. Gong, S. Zhou, Z. Niu, L. Geng, and M. Zheng, "Joint scheduling and dynamic clustering in downlink cellular networks," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Dec. 2011, pp. 1–5, doi: [10.1109/GLOCOM.2011.6133679](https://doi.org/10.1109/GLOCOM.2011.6133679).



OLFA CHABBOUH received the Ph.D. degree in information and communications technologies from the Higher School of Communications of Tunis (SUPCOM). She is currently an Assistant Professor of telecommunication with the Private Higher School of Engineering and Technology (ESPRIT). She is a member of the Mediatron Research Laboratory, SUPCOM. Her research interests include wireless mobile networks, resources allocation and management, and cloud computing and networking.



SONIA BEN REJEB (Member, IEEE) received the DEA degree from the National Engineering School of Tunis (ENIT), in 1999, and the Ph.D. degree from the Higher School of Communications of Tunis (SUPCOM), Université de Bretagne Occidentale (UBO), France, in February 2006. She is currently an Assistant Professor of telecommunication with the Higher Institute of Computer Science of Tunis (ISI). She is a member of the Mediatron Research Laboratory, SUPCOM. Her

research interests include wireless mobile networks (5G/6G), resources allocation and management, the Internet of Things (IoT), cloud computing and networking, blockchains, big data, machine learning, and so on. She is the author or coauthor of 49 articles. She has been a member of the technical program of several international conferences. She received the Best Research Paper Award at the 23th International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2015) and the IBM Blockchain Developer Mastery Award, in 2019. She received the Excellent Record of Viewers at the *Journal of Network and Computer Applications* (JNCA), in 2014.



NIDAL NASSER (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees (Hons.) in computer engineering from Kuwait University, Kuwait, in 1996 and 1999, respectively, and the Ph.D. degree from the School of Computing, Queen's University, Kingston, ON, Canada, in 2004. He worked at the School of Computer Science, University of Guelph, Guelph, ON, Canada, from 2004 to 2011. He was the Acting Dean of the College of Engineering, Alfaisal University, Saudi

Arabia, from 2014 to 2017, where he is currently a Professor of software engineering. He has authored 180 journal publications, refereed conference publications, and book chapters in the area of wireless communication networks and systems. He serves as an Associate Editor for the IEEE *WIRELESS COMMUNICATIONS*, the *International Journal of Communication Systems* (Wiley), and the IEEE *COMMSOFT E-LETTER*. He regularly serves on the organizational and technical committees of a number of conferences.



NAZIM AGOULMINE (Senior Member, IEEE) has been a Full Professor with the University of Évry Val d'Essonne and Paris-Saclay University, France, since 2000. From 2011 to 2016, he worked at the French Funding Agency (ANR) in charge of several research funding programs in future networks hardware and software infrastructures. He is currently the Vice-President of the university in charge of the international relation and the Vice Director of the IBISC STIC Research Laboratory

(more than 100 researchers). During the last 30 years, he has been participating and leading numerous research projects in ICT funded by EU and other national and international funding agencies. He is nowadays providing expertise for several international funding agencies and organizations. His current research interests include cloud computing, the Internet of Things, 5G networks, security and privacy, and eHealth. He has authored numerous research articles in and several books and book chapters in the area.

ZIÈD CHOUKAIR is currently leading research work with Paris XI University and a Professor with Sup'Com Tunis in ICT. Previously, he was with French University and the Group of Telecom Engineering Schools in distributed systems and telecom services. He also worked at Capgemini, where he managed projects involving data management and distributed constrained systems. His research interests include wireless mobile networks, resources allocation and management, cloud computing and networking, machine to machine (M2M), the Internet of Things (IoT), blockchains, big data, machine learning, and so on.

...