KORA: A Framework for Dynamic Consolidation & Relocation of Control Units in Virtualized 5G RAN

Debashisha Mishra, Himank Gupta, Bheemarjuna Reddy Tamma and Antony Franklin A
Department of Computer Science and Engineering, Indian Institute of Technology Hyderabad, India
Email: [cs15mtech01003, cs16mtech01001, tbr, antony.franklin]@iith.ac.in

***Abstract*—The ambitious goals of Fifth Generation (5G) mobile networks for higher system capacity, massive number of devices and flexibility in operations demand the network architecture to be much more flexible, efficient and autonomous. The design of Radio Access Network (RAN) is undergoing architectural transformations to increase the flexibility of deployment and programmability by leveraging Network Functions Virtualization (NFV), Software Defined Networking (SDN), and Cloud Computing. Hence, efficient resource management strategies with enhanced service quality play a vital role in realizing true benefits of 5G RAN. In this work, we propose a novel and dynamic resource management framework called "KORA" for 5G Cloud Radio Access network (C-RAN) considering spatio-temporal traffic heterogeneity exhibited at Remote Radio Units (RRUs). To minimize net energy consumption in the cloud data center and to maximize the service quality to end users, we formulate an Integer Linear Programming model (ILP) for KORA that performs efficient consolidation and relocation of Control Units (CUs) in 5G C-RAN. To alleviate the computational heaviness of the ILP optimization model, we propose a light-weight heuristic algorithm that is scalable and applicable to real-world dense deployments spanning a large set of CUs. By simulations, we compare and contrast between various distinctive features of 5G C-RAN architecture under study as well as evaluate the efficacy of our proposed KORA framework. The heuristic algorithm can save 27% of relocations and 33% of GBR flows from disruption, at increased energy consumption of 6.6% in data center as compared to KORA.**

I. Introduction

The cellular operators are facing a lot of challenges in deploying and managing Radio Access Network (RAN) infrastructure due to ever increasing demands from mobile subscribers for wider variety of services and higher rates. A standard way to meet these challenges require site expansions to lay out massive number of base stations (*i.e.,* Macro, Micro, and, Small cells). However, the CAPital EXpenditure and OPerational EXpenditure (CAPEX & OPEX) are very high for such RAN expansion policy. Additionally, limited availability of licensed spectrum and high complexity of maintenance, upgradation of network services do not seem to improve the economics of Mobile (Virtual) Network Operators (MNO, MVNO). Cloud RAN (C-RAN) is a novel and innovative cellular network architecture which takes the advantage of softwarization, programmability and network functions virtualization (NFV) to address the above mentioned deficiencies in a cost effective manner [1]. Transformation of traditional RAN architecture to support RAN function virtualization and cloud native implementations is identified as one of the key use cases of ETSI NFV framework [2]. A minimal real-time prototype encompassing above goals is demonstrated in our previous work [3] using OpenAirInterface(OAI) platform [4].

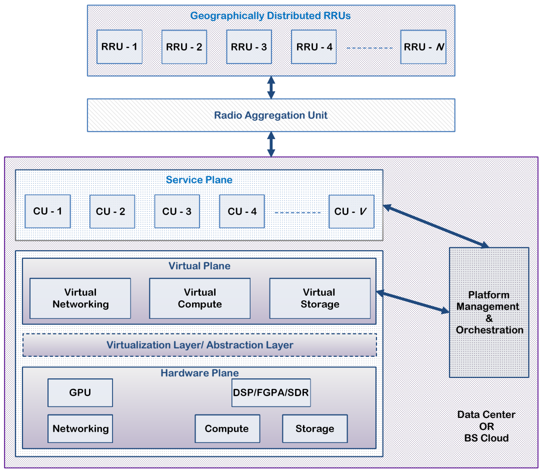

Fig. 1: Integrated & shared computing facility for 5G C-RAN architecture.

Fig. 1 depicts 5G C-RAN architecture envisioned in our study, compliant with the ETSI NFV reference architecture. In C-RAN, the base station functionality is segregated into geographically distributed radio components known as remote radio units (RRUs) and baseband units known as Control Units (CUs), remotely pooled in a data center (BS Cloud). Control Unit (a.k.a. Central Unit in 5G New Radio) is a more generic terminology followed after 3GPP Rel-14 technical specification, which was commonly referred as BBU in C-RAN literature. This protocol level split of base station facilitates the CU to be placed on a virtualized platform and share the computing infrastructure controlled by a cloud platform management and service orchestrator (MANO). From implementation perspective, the CU can be realized by a hypervisor based virtual machine (VM) or containerized LXC/Docker instance provisioned with required number of CPU cores, frequency of CPU operation (measured in GHz), allocated RAM size (in GB), disk storage space (in GB), and network bandwidth. Each CU serves a designated RRU (1:1 mapping) over a transport network called fronthaul adhered to strict latency and bandwidth requirements. The site resident RRU is responsible for digitizing, transmitting or receiving user signal in its coverage while CU performs the compute-intensive baseband signal processing of user traffic to/from RRU.

Nevertheless, the functional split in C-RAN between CU and RRU is flexible, and operators may decide to shift baseband functionalities partially or fully to match with the available bandwidth on the fronthaul [5]. In general, each base station protocol split option is viable with different requirements and trade-offs. The protocol split option where lower PHY stays at RRU end and higher PHY along with upper layers are moved to data center (Option-7 as per 3GPP TR 38.801), provides more centralization benefits at the cost of high fronthaul bandwidth. In another kind of split where RF, PHY and Layer 2 are kept at RRU end and upper layers (L3+) are moved to data center, relaxes the fronthaul bandwidth constraint at the cost of less opportunity for collaborative processing. Hence, depending upon the fronthaul transport network, a more flexible distribution of baseband processing to CU is feasible. This concept is known as "flexible centralization of base station functionalities". IEEE Next Generation Fronthaul Interface (NGFI) [6] has proposed many protocol split options which require different portions of the protocol stack divided between cell site and remote data center. However, for a given split, depending on the amount of real-time user traffic generated at RRU, corresponding CU's computational resource requirement may grow or shrink dynamically, thereby requiring unequal number of compute resources at different time instants [7].

In this work, we aim to design and develop an efficient resource management framework for CUs in 5G C-RAN based on traffic heterogeneity exhibited at RRUs (*i.e.,* cell sites). To this end, we propose "KORA", a dynamic resource management framework, capable of auto-scaling (automatically scale-up & scale down) compute servers to cope with fluctuating RRU traffic load. In next section, we characterize important considerations pertaining to resource pooling of CUs in a cloud data center and highlight main contributions of KORA.

II. Motivation : Traffic-aware Resource Pooling

A. Traffic Heterogeneity : Resource Imbalance

The traffic demands from mobile subscribers in the RRU coverage are typically random and consists of independent streams of user flows. Thus, the total traffic load generated at RRU imitates a random variable, best explained in terms of stochastic patterns. If we look at time series data of any day, the aggregate load from all the RRUs follow a diurnal human activity pattern *i.e.,* a periodic trend with low load during night time and high load during day time. In spatial domain, RRUs exhibit greater imbalance in traffic loads. For instance, urban/city areas show larger load variation than that of rural regions. Similarly, public workplaces (offices) tend to be very active in day time in weekdays while residential areas are moderately active during nights, off-hours, and holidays. This effect is known as spatio-temporal traffic pattern or tidal wave. Figs. 2 and 3 show normalized traffic loads in the range of [0,1] at 8 PM for 100 RRUs in an urban region of 10 $km \times 10$ km spatial layout considering both weekday and weekend profiles, respectively. These results are generated using a Gaussian Mixture Model (GMM) highlighted in [8].

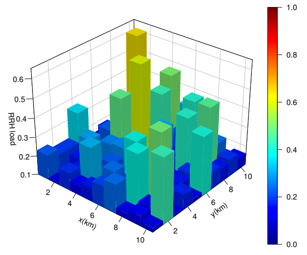

Fig. 2: Weekday spatial plot at 8 PM.

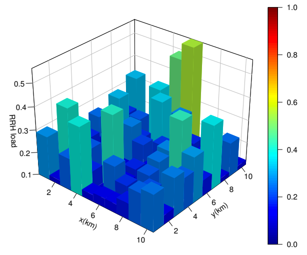

Fig. 3: Weekend spatial plot at 8 PM.

Each CU serves a designated RRU and is provisioned with virtual resources by abstracting (using hypervisor/containers) the compute server. But, if we allocate each CU to one compute server *i.e.,* 1:1 mapping between CU and compute server, traffic heterogeneity tends to under-utilize the computational resources, because the server has to be active even when there are very few users being served by the corresponding CU. Again, the peak load occurs only for few hours of the day. This type of 1:1 allocation between CU and compute server is very inefficient and increases CAPEX and OPEX for operators.

B. Consolidation of CUs : Resource Minimization

Multiplexing gain is determined by the degree to which the underlying computing resources can be shared. By fundamental principles of "statistical multiplexing gain", it is beneficial to place multiple CUs in the same compute server provided that the total utilization by all the hosted CUs do not exceed server's rated capacity. Mobile or C-RAN operators benefit from this approach by requiring fewer active compute servers for all the CUs and save energy by switching-off idle compute servers [1]. This phenomenon of assigning CUs into a subset of total compute servers available in the cloud is known as **Consolidation**. The allocation between CUs and compute servers at time t can be represented by an Allocation Matrix 'A_t'. Each entry $[x, y]$ in A_t is a binary value which equals to 1, if CU x is placed on compute server y, otherwise 0.

C. Relocation of CUs : Resource Regularization

In the next time epoch, due to tidal traffic heterogeneity, the end users within an RRU coverage may impose different traffic loads to its corresponding CU for processing. Hence, the allocation Matrix A_t may not be a suitable allocation at time $(t+1)$, as some compute servers might have been over subscribed or underloaded. Hence, some of the virtualized CUs need to be migrated to the appropriate destination compute servers, so that revised allocation matrix A_{t+1} represents an optimal allocation at time $(t+1)$. This phenomenon is known as **Relocation**. A given CU v is said to be relocated iff, $A_t(v) \neq A_{(t-1)}(v)$. As per cloud computing paradigm, these relocations are realized via live migrations of virtualized CUs.

One of the indigenous metrics to evaluate the efficiency of relocation of CUs is to measure the number of services interrupted during relocation downtime. In RAN, the CU is responsible for processing the baseband functions within strict deadlines. If the CU is relocated, then it may degrade the active user sessions in corresponding RRU, thereby impacting the QoS and SLA guarantees for the end users. Moreover, CU relocations may result in radio failures/outages. Note that, these relocations are costly and incur extra overhead. Therefore, careful decisions to have minimal impact on the offered QoS must be prioritized for better cellular service. To collectively address the challenges in consolidation and relocation of CUs with minimum service discontinuity, we present "KORA", a novel and efficient resource management framework for 5G C-RAN in this paper.

D. Related Work

The scope of most of existing resource management works on C-RAN involved the consolidation aspect while not factoring the relocation overhead and its effect on service continuity due to tidal traffic burstiness. This is an important aspect in data center environments which needs a careful study for 5G. A multiple knapsack formulation for dynamic RRU-BBU assignment is presented in [9] and solved using IBM CPLEX solver. The approach minimizes the total number of required BBUs for a given set of RRU loads and suitably adjusts transmit powers of RRUs, but this work did not consider the space-time traffic heterogeneity from cell sites. Also, it did not consider BBU overload scenario and change of RRU assignment needed to cope with peak/off-peak hours. By using multi-dimensional Markov model, Liu et. al. in [10] evaluated the statistical multiplexing gain of virtual BS pool considering user sessions and cloud processing constraints, but it lacked consideration for non-uniform peak/off-peak hour traffic burstiness. Khan et. al. in [11] presented a QoS-aware optimization formulation to solve RRU-BBU mapping problem based on genetic algorithm, but no results have been shown w.r.t. its effect on active number of BBUs, relocations as well as the scalability for dense C-RAN environment spanning hundreds of RRUs. In [12], a load-aware, dynamic RRU assignment (DRA) clustering algorithm based on modified 1D bin packing is proposed w.r.t. heterogeneous traffic. DRA focused mainly on minimization of compute servers in the data center and did not factor relocations and QoS guarantees.

The main contributions of this work are as below.

- Design of a novel and traffic-aware dynamic resource management framework, "KORA", which performs efficient consolidation and relocation of CUs by factoring in the adverse effect of relocation on user services.
- Formulated an Integer Linear Programming (ILP) model to minimize net energy consumption and service disruptions to users during relocation of CUs.
- Proposed a light-weight, greedy algorithm to consolidate and relocate CUs in fine granularity of time which converges to a near-optimal solution.

III. System Model & Problem Formulation

In this section, we describe the RRU user traffic and CU computing resource models for 5G RAN. Consider $\mathcal{N} = \{1, 2, \ldots, N\}$, $\mathcal{V} = \{1, 2, \ldots, V\}$, $\mathcal{M} = \{1, 2, \ldots, M\}$ as the finite set of RRUs, CUs, and compute servers in BS Cloud, respectively. Since each RRU traffic is served by its corresponding CU, we have $V = N$. Let $\mathcal{U} = \{U_1, U_2, \ldots, U_k\}$ be the set of k users distributed under coverage of N RRUs. Based on stochastic point process theory [13], we deploy UEs and RRUs in a given geographical range and simulate the streams of different application traffic at UEs.

A. Point Processes for modeling UE and RRU Distribution

For modeling the cellular layouts, tools from stochastic geometry, spatial statistics and point processes prove to provide more realistic and accurate analysis. These tools provide tractable approach to derive several important observations and results to the operators to estimate coverage, rate and cost apriori to the deployment. We model the RRU deployment by a single operator as per Matern hard core point process type II (MHCPP II) and the UEs distribution as per Poisson Point Process (PPP) [13]. Let the spatial distribution of UEs and RRUs be Φ_{UE} and, Φ_{RRU}, respectively, where $\{\Phi_{UE}, \Phi_{RRU}\} \in \mathbb{R}^2$. Individual RRU coverage regions are plotted and separated by voronoi tessellation. For accuracy, the channel is modeled including the propagation loss with shadowing and fading.

B. RRU Traffic Model : Cell Load

At time t, each RRU $r \in \mathcal{N}$ is associated with a set of users $U_{\{r\}}(t) \subset \mathcal{U}$. This association policy can be based on the strongest received signal strength. Let $h^u_{\mathcal{N}}(t) = (h^u_{\{1\}}(t), h^u_{\{2\}}(t), ..., h^u_{\{N\}}(t))$ be the complex channel vectors from all the RRUs to user $u \in \mathcal{U}$ at time t. Note that, channel $h^u_{\{r\}}(t)$ is known as signal channel for serving RRU r, else considered as an interference channel. The RRU is assumed to have instantaneous knowledge of achievable transmission rate R_u to each user u at each TTI. We utilize a Priority Set Scheduling (PSS) algorithm at CU which uses the achievable rate information and channel quality to determine the amount of Physical Resource Blocks (PRBs) to be allocated to each user. Hence the load at each RRU r at time t, *i.e.,* $l_r(t)$ is given by,

$$l_r(t) = \sum_{u \in U_{\{r\}}(t)} \frac{No\_of\_PRB\_Allocated\_to\_user\_u}{No\_of\_Available\_PRB} \quad (1)$$

To model realistic traffic conditions as in 3GPP TS 23.203, this work focuses on four Guaranteed Bit Rate (GBR) applications (*i.e.,* number of Voice traffic flows (N_voice), Real-Time Gaming (N_game), Conversational Videos (N_conv), Live Streaming Videos (N_stream)) and number of Non-GBR data flows (N_ngbr) for each user. For each RRU, we compute a weighted score metric ws_r which represents a unified value quantifying active user flows as in Eqn. 2.

$$\begin{aligned} ws_r = (w_1 \times N\_ngbr) + (w_2 \times N\_voice + \\ w_3 \times N\_conv + w_4 \times N\_game + w_5 \times N\_stream) \end{aligned} \quad (2)$$

such that, $\sum_{i=1}^{5} w_i = 1$. Individual RRU weighted score ws_r such that $ws_{min} <= ws_r <= ws_{max}$ can be normalized within range [0,1] as follows.

$$\frac{(ws_r - ws_{min})}{(ws_{max} - ws_{min})} \quad (3)$$

An RRU possessing high normalized score indicates more prioritized user streams. Similarly, two CUs $v1, v2 \in \mathcal{V}$ serving RRUs $r1, r2 \in \mathcal{N}$, can also be prioritized as in Eqn. 4.

$$\rho_{v1} > \rho_{v_2} \iff ws_{r_1} > ws_{r_2} \tag{4}$$

where ρ_v denotes normalized weighted score for CU $v \in \mathcal{V}$.

Remark 1. *Since there are many weighted decision variables $(w_1, w_2, w_3, w_4, w_5)$ involved in score metric calculation of an RRU, an analytical proof for finding optimal value of weights is non-trivial. However, these parameters can be decided by an operator through policy planning. Note that, each of these policies is operator specific and KORA is flexible enough to be tuned to any set of policy parameters to show efficacy of our proposed framework.*

C. CU Computing Resource Model : Compute Load

The baseband processing load for user $u \in \mathcal{U}$ at time t in coverage of RRU r consists of two main components. The first component is a constant cell specific base processing load fixed for given r, which is independent of users. The second component is a dynamic user dependent value modeled as a function of allocated time-frequency channel resources and modulation and coding scheme (MCS) assigned to those channel resources [14]. The baseband processing time per subframe $proc(u, t)$ in microsecond is given by,

$$proc(u, t) = r_{base} + p_{base} + u(mcs, prb) + u(r) \tag{5}$$

where r_{base} and p_{base} are the constant base offsets for the cell and virtualized platform, respectively. $u(mcs, prb)$ is the user dependent processing as a function of allocated PRB and MCS, and $u(r)$ is the remainder of other user specific tasks.

The fundamental unit of measurement of any cloud compute platform is the number of instructions that it processes per second. However, for accurate analysis, scientific formulations use the count of FLoating point Operations Per Second (FLOPS) as a measure of computer performance. For example, in double precision convention, a general purpose Intel CPU core can perform four floating point operations per CPU cycle. Consider a single core Intel CPU of 2 GHz frequency, which denotes that the CPU is capable of 2 billion CPU cycles per second, thus resulting in a theoretical performance of $(2 \times 10^9 \times 4) = 8$ GFLOPS. Let us denote this limit as L_{max}. Note that, this limit may vary for different vendor specific target hardware and architecture platforms. The compute load $l_v(t)$ for CU $v \in \mathcal{V}$ in FLOPS serving RRU $r \in \mathcal{N}$ is given by,

$$l_v(t) = L_{max} \times \left(\sum_{u \in U_{\{r\}}(t)} proc(u, t) \right) \tag{6}$$

D. Cost Functions for CU Consolidation & Relocation

As mentioned in Section II-C, CU relocation through live migration incurs additional cost or resource overhead, because it iteratively writes all the active memory pages of CU from source to target compute server. Similar to [15], we model this extra cost in terms of additional rise in the power requirement during relocation. The linear relationship between compute load and power consumption is adopted to model the power overhead incurred while performing relocations. Considering P_{idle} as the idle power drawn at 0% load and P_{cap} as the maximum power drawn by the compute server at 100% compute load, the power consumption of a compute server with compute load l_m is given by,

TABLE I: Glossary

Notation	Definition
l_v	Compute load at CU $v \in \mathcal{V}$
C_m	Capacity of compute server $m \in \mathcal{M}$
z_m	1 if compute server $m \in \mathcal{M}$ is active; otherwise 0
y_{vm}	1 if CU v is active on compute server m; otherwise 0
A_t	Allocation matrix of all CUs to compute servers at time t
$A_t(v)$	Allocation of CU $v \in \mathcal{V}$ to compute server m at time t
$cost_m$	Energy Cost of operating compute server $m \in \mathcal{M}$
$cost_{vm}$	Additional energy cost incurred in relocation of CU $v \in \mathcal{V}$ to compute server $m \in \mathcal{M}$
ρ_v	Normalized score of $v \in \mathcal{V}$ indicating relocation impact

$$P_m = P_{idle} + (P_{cap} - P_{idle}) \times l_m \tag{7}$$

The energy consumption is given by $(P_m \times t_m)$, where t_m represents the time duration for which server operated at P_m power. As per SPECpower benchmark[1], average power consumption for a standard General Purpose Processor (GPP) server with compute load of 100% is approximately 259 Watt. The additional energy consumption E_v in Watt-second during the relocation of CU v between source and destination compute server is given by,

$$E_v = 0.512 \times B_v + 20.165 \tag{8}$$

where B_v is the total amount of data volume relocated from source to destination server [16].

E. Integer Linear Programming (ILP) Optimization Model

Table I presents the notations used in problem formulation of KORA as an integer linear programming optimization model. The ILP model broadly caters two distinct objectives.

(A) At any instant of time, the virtual compute resources at CU must be allocated to minimum number of compute servers so as to minimize the total energy consumption (1st term in Eqn. 9).
(B) The additional energy incurred during relocations of CUs (2nd term in Eqn. 9) needs to be minimized while factoring the effect on service continuity during CU relocations.

Objective Function : *Minimize*

$$\left(\sum_{m=1}^{M} (z_m \times cost_m) \right) + \left(\sum_{\substack{v \in \mathcal{V}, m \in \mathcal{M}, \\ \text{such that} \\ m \neq A_{(t-1)}(v)}} (\rho_v \times y_{vm} \times cost_{vm}) \right) \tag{9}$$

Constraints :

$$\sum_{m=1}^{M} y_{vm} = 1, \quad \forall v \in \mathcal{V} \tag{10}$$

[1]http://www.spec.org/power_ssj2008/results/res2010q4/

$$\sum_{v=1}^{V}(y_{vm} \times l_v) \leq (C_m \times z_m), \qquad \forall m \in \mathcal{M} \tag{11}$$

Eqn. 10 ensures that each CU is associated with exactly one compute server. Eqn. 11 ensures capacity constraint *i.e.,* sum of compute loads from CUs associated to a compute server does not exceed the server's rated capacity.

The above ILP optimization model minimizes the total energy consumption of the C-RAN system as well as helps to minimize the service disruption to user flows due to CU relocations. Let us denote the two distinct terms of the objective function in Eqn. (9) as $(A+B)$. If we omit the 2nd objective term B, the ILP model is oblivious to CU relocations and only tries to minimize the total energy consumption of active compute servers. This specific formulation of KORA ILP model is known as *relocation-oblivious* formulation. By adding the 2nd objective term B, the formulation of KORA becomes *relocation-aware*, thereby ensures minimum disruption to the UE data flows. Note that, in this work, the name KORA refers to *relocation-aware* framework and these are synonymous to each other. Section V presents some important distinctive results of these two different formulations *i.e.,* KORA and *relocation-oblivious* KORA.

F. Challenges of KORA ILP Model

Although ILP model for KORA leads to optimality in 5G C-RAN spanning a large number of RRUs, KORA takes longer execution time to converge to a solution. Hence, it is not scalable for the dense deployment scenario. To showcase the computational heaviness of the proposed ILP model, we varied the input size from 10 to 50 CUs and calculated its execution time. We employed a mixed-integer-programming (MIP) solver called "Gurobi" executing 4 concurrent threads in an Intel Xeon E5620, 2.40 GHz machine running GAMS Version 24 under 64-bit Windows 7. The results in Fig. 4 shows the maximum execution time taken by the ILP model over 240 iterations. In worst case, it takes 122 seconds (~2 minutes) to converge to the solution for 50 CUs.

Typically in 5G C-RAN, the consolidation and relocation decisions must be taken in a very fine granularity of time *e.g.,* 1 ms LTE scheduling interval to react to real-time tidal traffic variations at RRUs. Till date, to the best of our knowledge, there are no such algorithms in place which can attain convergence at very fine granularity. The proposed ILP model takes several minutes in worst case. Therefore, we propose a novel, *relocation-aware* heuristic algorithm which produces efficient performance guarantees.

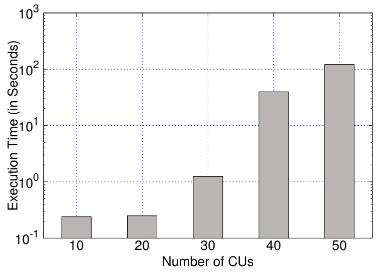

Fig. 4: Execution Time for KORA ILP Model.

IV. Relocation-Aware Heuristic Algorithm

In this section, we propose a *relocation-aware* greedy heuristic algorithm for KORA that is scalable and computationally more viable for larger problem instances. Algorithm aims not only in minimizing energy cost of active compute servers, but also minimizes energy cost of relocations considering the adverse effect on user services during relocations. Based on the compute load, all compute servers in the data center can be classified into three classes. The set of servers whose utilization exceeded the maximum threshold is said to be overloaded set S_o. The set of servers whose utilization is below the maximum threshold is said to be non-overloaded set S_n. The set of servers whose utilization is below a minimum threshold is said to be underloaded set S_u. The heuristic algorithm regularizes the overloaded compute servers into non-overloaded ones and the underloaded servers are merged to minimize the total number of active compute servers.

The *relocation-aware* heuristic operates in three distinct stages for every overloaded compute server. They are :

I. Selecting a suitable candidate CU for relocation from an overloaded compute server (identified as source server).
II. Determining a non-overloaded, active target server to place chosen candidate CU. If no such server found, instantiate a new server as a target for candidate CU.
III. Iteratively write the active memory pages/contexts of candidate CU from source to target compute server.

In the first stage, while selecting a candidate CU for relocation, we adopt a Minimum Relocation Cost (MRC) policy, i.e., relocating a CU $v \in \mathcal{V}$, that has lowest relocation score (ζ_v). The relocation score metric ζ_v is calculated for each CU $v \in \mathcal{V}$

Algorithm 1 : Relocation-aware Greedy Heuristic for KORA

```
    Input : Previous allocation matrix $A_{t-1}$ and $l_v$ for all CUs.
    Output : Best possible allocation matrix $A_t$ at time epoch $t$.
 1: procedure GETALLOCATIONMATRIX
 2:     while ($S_o$ is not $NULL$) do
 3:         $excess \leftarrow \left(\sum_{A_t(v)=m}(l_v)\right) - C_m$
 4:         Find eligible CUs for relocation i.e., $l_v > excess$
 5:         Compute $\zeta_v$ for all eligible CUs
 6:         $CandidateCU \leftarrow$ CU with lowest $\zeta_v$
 7:         Select a target compute server $\beta$ for $CandidateCU$
 8:         if $\exists\beta$ then
 9:             Relocate $CandidateCU$ to $\beta$
10:         else
11:             Instantiate a new compute server $\beta'$ as target
12:             Relocate $CandidateCU$ to $\beta'$
13:         end if
14:         Update $S_o$ and $S_n$
15:     end while
16:     while ($S_u$ is not $NULL$) do
17:         Merge elements of $S_u$ respecting capacity constraint
18:     end while
19:     Return the new allocation matrix $A_t$
20: end procedure
```

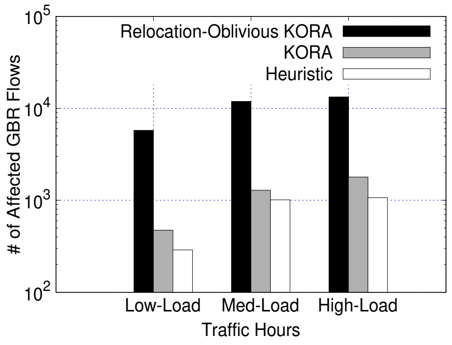

Fig. 5: Total Number of Affected GBR Flows.

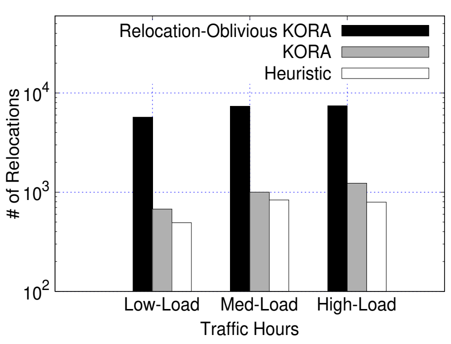

Fig. 6: Total Number of CU Relocations.

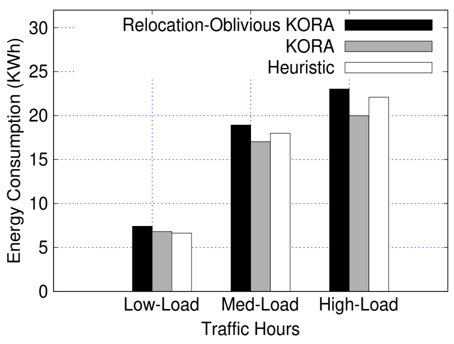

Fig. 7: Total Energy Consumption (KWh).

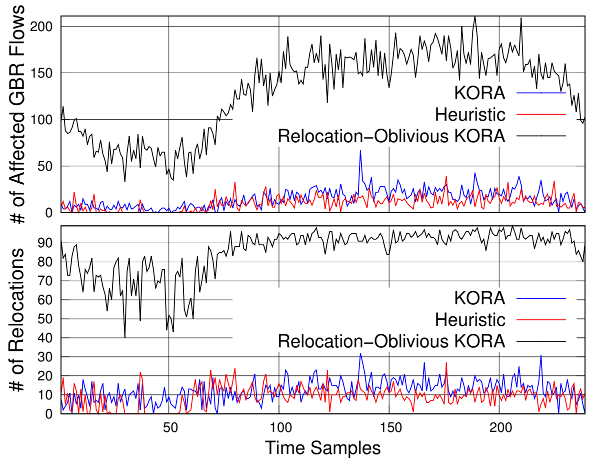

Fig. 8: Number of Affected GBR Flows w.r.t. time.

TABLE II: Simulation Parameters

Parameter	Value
Number of RRUs	100
Sampling Interval	6 Minutes
Total Traffic Capture Window	24 Hours
Total Generated Samples	240
RRU workload Range	Normalized in [0,1]
$[w_1, w_2, w_3, w_4, w_5]$	$[0.10, 0.30, 0.20, 0.25, 0.15]$
Time-varying rate parameter	Gaussian Mixture Model

and is a weighted average of ws_v and lv (computed in Eqns. 2 and 6). Metric ζ_v is calculated as, $\zeta_v = (\alpha \times ws_v) + ((1-\alpha) \times l_v)$ such that $0 \leq \alpha \leq 1$. The CUs with lower ζ_v are considered as better candidates for relocation. Note that, a higher value of α favors ws_v *i.e.,* CUs serving RRU with least GBR flows are preferred for relocation, thereby improving overall QoS of system. Alternatively, lower α favors compute load l_v *i.e.,* CUs serving RRU with lower compute load are preferred for relocation, thereby saving the energy cost. This inherent flexibility in choosing α significantly influences the operator policy planning. Depending upon whether the operator's policy is prioritizing QoS or energy, MRC can be well tuned for the heuristic algorithm.

In the second stage, we use a variant of Best Fit (BF) bin packing approximation algorithm to identify a target for candidate CU. Here best bin refers to the compute server which can accommodate the candidate CU and the residual utilization (i.e., Capacity - Total_Utilization) after loading the candidate CU on that server must be minimum. As an extension to classical BF approach, our relocation heuristic instantiates a new compute server in case there are no existing non-overloaded compute servers to accommodate candidate CU. The pseudo-code of proposed algorithm is given in Algorithm 1. After regularization process by the heuristic algorithm, all the servers become either non-overloaded or underloaded. For each underloaded server, heuristic algorithm merges one/more underloaded compute servers to scale down the number of active compute servers used in the data center.

V. Experimental Setup and Performance Results

The large-scale simulations for KORA framework are carried out considering a service region of $10KM \times 10KM$ with 100 RRUs distributed geographically as per MHCPP-II and 1000 users are distributed as per PPP as described in Section III-A. The channel gains are generated by considering large scale fading and COST-231 path loss model. Standard deviation of log normal shadowing is 10 dB and noise PSD is fixed as -184 dBm/Hz. For simplicity, transmission powers of all RRUs are kept same *i.e.,* one Watt. Table II lists out essential simulation parameters. The user traffic generated in a full day of 24 hours is divided into three segments. They are "Low_Load" from 12AM to 8AM, "Med_Load" from 8AM to 12 Noon and 8PM to 12AM, "High_load" from 12 Noon to 8PM. For a full day, a total of 240 time series samples are collected by the Gaussian Mixture Model (GMM) highlighted in [8] where each sample is collected in 6 minutes epoch.

We conducted four sets of experiments to compare and contrast between various specifications of KORA framework. The experimental evaluations are shown with respect to (a) Number of active GBR flows impacted during relocations, (b) Number of Relocations, (c) Energy Consumption, and (d) Scalability and execution time performance. The performance metrics are compared with three specifications of KORA framework *i.e.,* (a) *Relocation-Oblivious* KORA, (b) KORA, and (c) Heuristic algorithm for KORA. KORA and Heuristic implicitly refer to *relocation-aware* mechanisms.

Figs. 5, 6, and 7 present a comparative study of total number of GBR flows affected, total number of relocations, and energy consumption during Low_Load, Med_Load and High_Load traffic scenarios, respectively. *Relocation-Oblivious* KORA only focuses on minimizing the total energy consumption due to active compute servers and does not factor the relocation cost. Therefore, it incurs disruption to a large number of GBR flows (~13296 in High_Load) in all the three traffic scenarios. In case of KORA, the framework intelligently leverages the "affinity" property of CUs to compute servers for finding suitable allocation matrix. It means CU chooses to remain at the previously allocated compute server. KORA is able to outperform *relocation-oblivious* scheme by saving 88.53% of affected GBR flows. The same argument is valid for Fig. 6 as well, where the number of active relocations incurred are 85.74% less than *relocation-oblivious* scheme. If we consider the total energy consumption per different traffic hours as shown in Fig. 7, *relocation-oblivious* scheme consumes more energy than KORA due to the additional energy spent on higher number of relocations. Energy spent on each relocation is significant and added to the compute server energy consumption to calculate the total energy consumption per traffic scenario. However, in case of KORA, relocations are optimized as per necessity, thereby consuming 11.2% of less energy compared to *relocation-oblivious* scheme. The motive of heuristic approach is to maintain the striking optimization balance between *relocation-oblivious* KORA and KORA. In Fig. 7, the energy consumption by heuristic approach tends to overestimate the total energy consumption, as it performs local search optimization on the allocation matrix following a greedy procedure. Hence, heuristic may use more number of active compute servers which leads 6.67% more energy consumption than KORA. The increase in the number of compute servers reduces the possibility of relocations, resulting fewer relocations and less number of affected GBR flows. As shown in Figs. 5 and 6, heuristic is able to save 27.39% of number of relocation and 33.26% of number of affected GBR flows compared to KORA.

Fig. 8 presents the instantaneous variation in number of GBR flows affected and number of relocations occurred w.r.t. 240 time series samples throughout the day. This result establishes a proportional relationship between number of GBR flows affected and number of relocations occurring at any time epoch. A rise in relocation count also impacts the flow disruption proportionally. Fig. 9 shows trade-off between energy consumption and number of affected GBR flows controlled by parameter α in heuristic algorithm. By controlling α value appropriately, the service provider can optimally choose a suitable policy for their users. At $\alpha = 1$, the heuristic algorithm is able to save 39.62% of affected GBR flows than that of $\alpha = 0$, but the energy consumption is increased by 7.45%. We considered $\alpha = 0.43$, where two contrasting objectives are equally good. Fig. 10 shows the scalability of proposed heuristic algorithm by varying the number of CUs. In contrast to the execution time of ILP model shown in Fig. 4, heuristic is light-weight and executes in few seconds. This feature makes this scheme a suitable option for realistic deployment of 5G C-RAN in data centers.

VI. Conclusions

This work presented a novel and efficient resource management framework, KORA, for the virtualized 5G RAN. In this

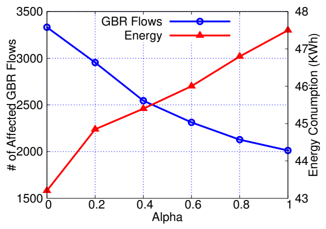

Fig. 9: Effect of $\boldsymbol{\alpha}$ on GBR flows and Energy in Heuristic Algorithm.

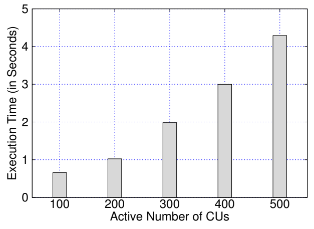

Fig. 10: Run-time performance and Scalability of Heuristic Algorithm.

paper, we investigated different trade-offs involved in dynamic consolidation and relocation of CUs across compute servers to minimize the service disruption. First, we formulated the problem as an ILP optimization model and then characterized the optimal solution w.r.t. *relocation-oblivious* and *relocation-aware* objectives. To reduce the computational complexity of ILP, a scalable, time-efficient relocation-aware heuristic algorithm is proposed to find solution for the same. The proposed heuristic algorithm saves 27% of relocations and 33% of GBR flows from disruption, but consumes 6.6% more energy than KORA.

VII. Acknowledgements

This work is supported by the project "Converged Cloud Communication Technologies", Meity, Govt. of India.

References

[1] A. Checko *et al.*, "Cloud RAN for mobile networks - A technology overview," *IEEE Comm. Surveys & Tutorials*, vol. 17, Sep 2014.
[2] "Network Functions Virtualisation (NFV); Use Cases." http://www.etsi.org/deliver/etsi_gs/NFV/001_099/001/01.01.01_60/gs_NFV001v010101p.pdf.
[3] S. S. Kumar *et al.*, "FLEXCRAN: Cloud radio access network prototype using OpenAirInterface," in *IEEE COMSNETS Demos*, Jan 2017.
[4] "OpenAirInterface Software Alliance." http://www.openairinterface.org/.
[5] P. Rost *et al.*, "Cloud technologies for flexible 5G Radio Access Networks," *IEEE Comm. Magazine*, vol. 52, May 2014.
[6] Y. Zhiling *et al.*, "White Paper of Next Generation Fronthaul Interface v1.0," tech. rep., China Mobile Research Institute, Oct 2015.
[7] M. Qian *et al.*, "Baseband processing units virtualization for Cloud Radio Access Networks," *IEEE Wireless Comm. Letters, Jan 2015.*
[8] E. Nan, X. Chu, W. Guo, and J. Zhang, "User data traffic analysis for 3G cellular networks," in *CHINACOM*, IEEE, Aug 2013.
[9] M. Y. Lyazidi *et al.*, "Dynamic resource allocation for Cloud-RAN in LTE with real-time BBU/RRH assignment," in *IEEE ICC*, May 2016.
[10] J. Liu *et al.*, "Statistical Multiplexing Gain Analysis of Heterogeneous Virtual Base Station Pools in Cloud Radio Access Networks," *IEEE Trans. on Wireless Communications, vol. 15, May 2016.*
[11] M. Khan, Alhumaima, *et al.*, "Quality of Service aware dynamic BBU-RRH mapping in Cloud Radio Access Network," in *IEEE International Conference on Engineering and Technology* , Dec 2015.
[12] D. Mishra *et al.*, "Load-aware dynamic RRH assignment in Cloud Radio Access Networks," in *IEEE WCNC*, April 2016.
[13] H. ElSawy *et al.*, "Modeling and analysis of cellular networks using stochastic geometry: A tutorial," in *IEEE Comm. Surveys and Tutorials*, vol. 19, Nov 2017.
[14] N. Nikaein, "Processing radio access network functions in the cloud: Critical issues and modeling," in *International Workshop on Multiple Classifier Systems, Sep 2015.*
[15] Q. Huang, F. Gao, R. Wang, and Z. Qi, "Power consumption of virtual machine live migration in clouds," in *IEEE CMC*, Jun 2011.
[16] H. Liu *et al.*, "Performance and energy modeling for live migration of virtual machines," *Cluster Computing*, Jun 2013.