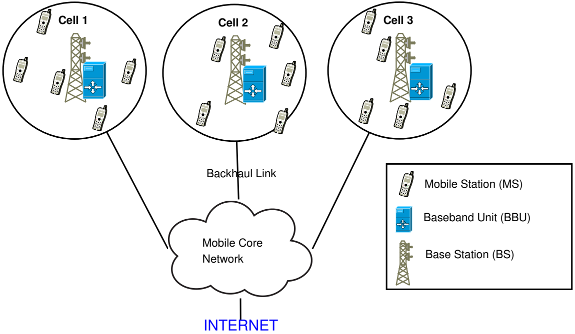
KORA: A Framework for Dynamic Consolidation & Relocation of Control Units in Virtualized 5G RAN

Debashisha Mishra, Himank Gupta, **Bheemarjuna Reddy Tamma** and Antony Franklin A

Networked Wireless Systems (NeWS) Lab
Dept. of Computer Science and Engineering
Indian Institute of Technology Hyderabad, INDIA

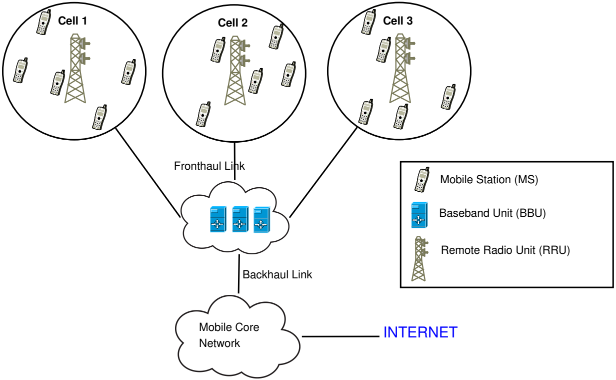
IEEE ICC 2018 SAC Symposium on
Cloud Communications and Networks Track
21^{st} May, 2018

Traditional RAN: 1G → 4G



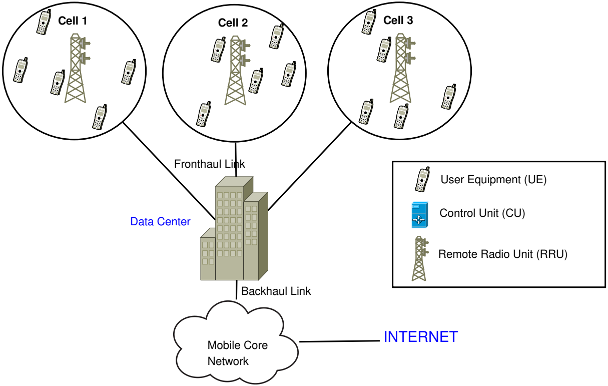
- Collocated RF & Baseband (inc. protocol stack) of Base Station (BS) at Cell sites
- Dedicated and proprietary hardware and software

RAN Evolution to Centralization: 4G+



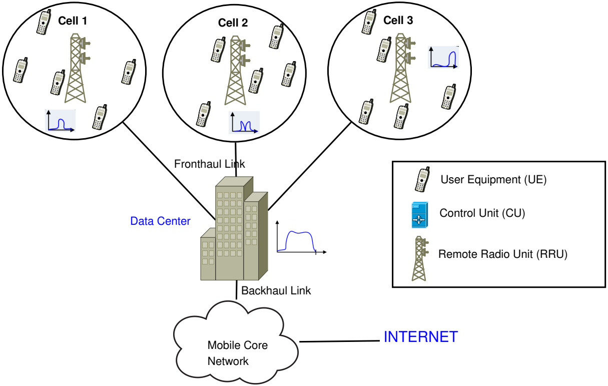
- RF and Baseband components spread at different locations
- Baseband functions from cell sites are pooled in a central office

RAN Virtualization : Cloud RAN



- RF and Baseband components spread at different locations
- Baseband functions are virtualized in a cloud data center
- Seen in 5G architectures from 3GPP & in IEEE NGFI

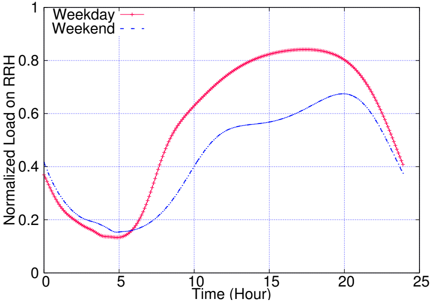
Multiplexing Opportunities of CUs in Cloud DC



- Multiplexing gain by sharing of resources as in *cloud computing*
- Tidal traffic pattern provides an opportunity to multiplex CU compute load on GPP servers

Spatio-temporal Pattern of Traffic Load

Temporal Dynamics



Load on individual base stations do not follow any periodicity, however the trend is consistent with diurnal activity patterns of human beings.

Spatio-temporal Pattern of Traffic Load

Spatial Dynamics

The residential zones tend to be active in off-hours (nights, weekends and holidays) while business or office areas are active during daytime in weekdays.

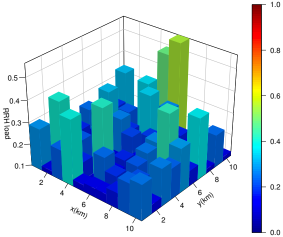


Figure: Weekend Spatial Plot

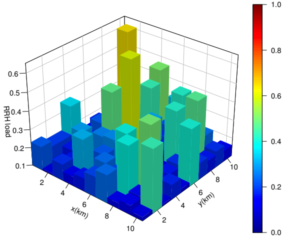
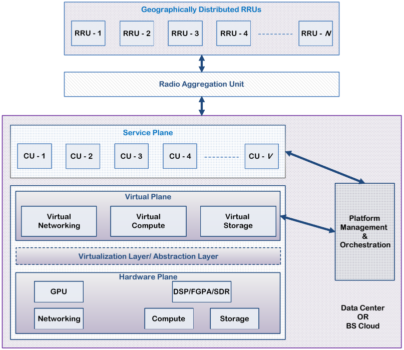


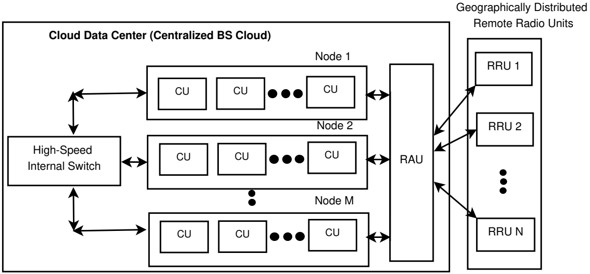
Figure: Weekday Spatial Plot

KORA: Integrated & shared computing framework for Cloud RAN in 5G

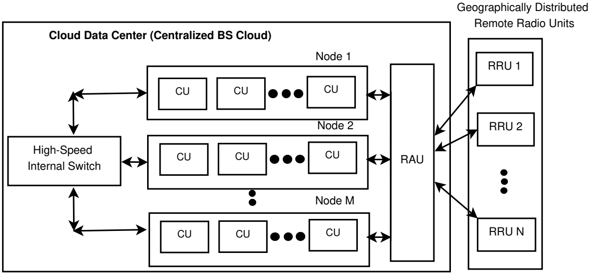


- CUs can be realized on a hypervisor based virtual machine (VM) or containerized LXC/Docker instances
- Depending upon fronthaul transport network, a more flexible distribution of baseband functions & higher layers of stack b/w RRU and CU is feasible
- Based on amount of real-time user traffic generated at RRU (aka DU), corresponding CU's computational resource requirement may grow (→ relocation) or shrink (→ consolidation) dynamically

System Notations

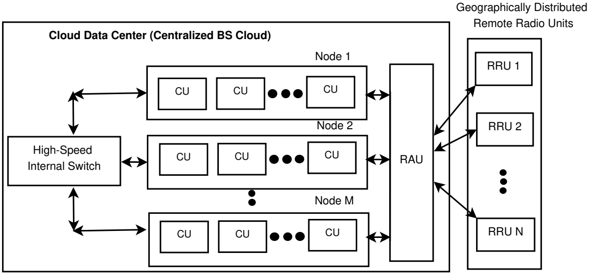


System Notations



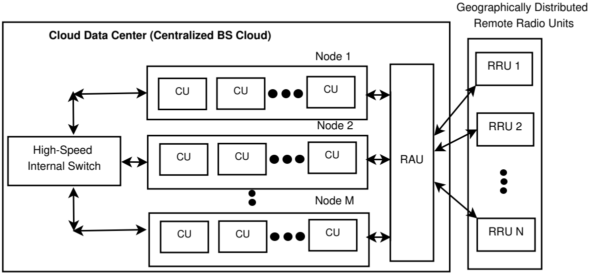
- $\mathcal{N} \rightarrow$ Set of RRUs, $\mathcal{N} = \{1, 2, ..., N\}$
  $\mathcal{V} \rightarrow$ Set of CUs, $\mathcal{V} = \{1, 2, ..., V\}$
  $\mathcal{M} \rightarrow$ Set of Compute Servers/Nodes, $\mathcal{M} = \{1, 2, ..., M\}$

System Notations



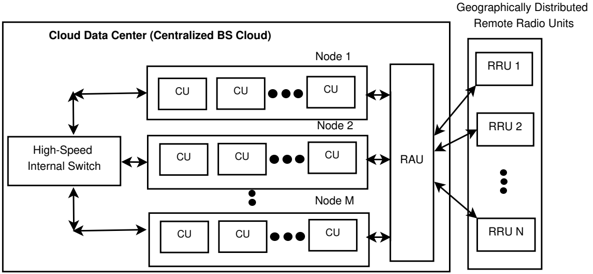
- $\mathcal{N} \rightarrow$ Set of RRUs, $\mathcal{N} = \{1, 2, ..., N\}$
  $\mathcal{V} \rightarrow$ Set of CUs, $\mathcal{V} = \{1, 2, ..., V\}$
  $\mathcal{M} \rightarrow$ Set of Compute Servers/Nodes, $\mathcal{M} = \{1, 2, ..., M\}$
- $N = V$ and $M \leq V$.

System Notations



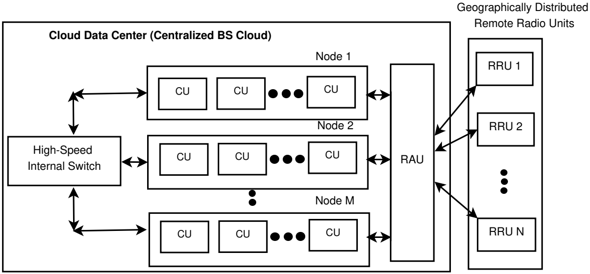
- $\mathcal{N} \rightarrow$ Set of RRUs, $\mathcal{N} = \{1, 2, ..., N\}$
  $\mathcal{V} \rightarrow$ Set of CUs, $\mathcal{V} = \{1, 2, ..., V\}$
  $\mathcal{M} \rightarrow$ Set of Compute Servers/Nodes, $\mathcal{M} = \{1, 2, ..., M\}$
- $N = V$ and $M \leq V$.
- Let l_v = Computer load of CU $v \in \mathcal{V}$ in FLOPS.

System Notations



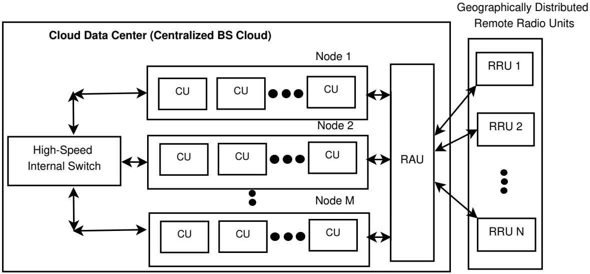
- $\mathcal{N} \rightarrow$ Set of RRUs, $\mathcal{N} = \{1, 2, ..., N\}$
  $\mathcal{V} \rightarrow$ Set of CUs, $\mathcal{V} = \{1, 2, ..., V\}$
  $\mathcal{M} \rightarrow$ Set of Compute Servers/Nodes, $\mathcal{M} = \{1, 2, ..., M\}$
- $N = V$ and $M \leq V$.
- Let l_v = Computer load of CU $v \in \mathcal{V}$ in FLOPS.
- Let L_{max} = Maximum capacity of computer node in FLOPS.

System Notations



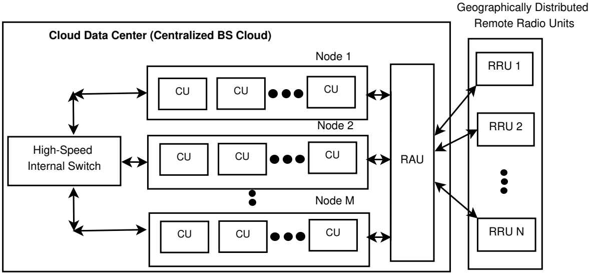
- $\mathcal{N} \rightarrow$ Set of RRUs, $\mathcal{N} = \{1, 2, ..., N\}$
  $\mathcal{V} \rightarrow$ Set of CUs, $\mathcal{V} = \{1, 2, ..., V\}$
  $\mathcal{M} \rightarrow$ Set of Compute Servers/Nodes, $\mathcal{M} = \{1, 2, ..., M\}$
- $N = V$ and $M \leq V$.
- Let l_v = Computer load of CU $v \in \mathcal{V}$ in FLOPS.
- Let L_{max} = Maximum capacity of computer node in FLOPS.
- Let C_m = Capacity of compute node $m \in \mathcal{M}$.

System Notations



- $\mathcal{N} \rightarrow$ Set of RRUs, $\mathcal{N} = \{1, 2, ..., N\}$
  $\mathcal{V} \rightarrow$ Set of CUs, $\mathcal{V} = \{1, 2, ..., V\}$
  $\mathcal{M} \rightarrow$ Set of Compute Servers/Nodes, $\mathcal{M} = \{1, 2, ..., M\}$
- $N = V$ and $M \leq V$.
- Let l_v = Computer load of CU $v \in \mathcal{V}$ in FLOPS.
- Let L_{max} = Maximum capacity of computer node in FLOPS.
- Let C_m = Capacity of compute node $m \in \mathcal{M}$.
- Let δt = Time interval (epoch) for continuous traffic measurements

System Notations



- $\mathcal{N} \rightarrow$ Set of RRUs, $\mathcal{N} = \{1, 2, ..., N\}$
  $\mathcal{V} \rightarrow$ Set of CUs, $\mathcal{V} = \{1, 2, ..., V\}$
  $\mathcal{M} \rightarrow$ Set of Compute Servers/Nodes, $\mathcal{M} = \{1, 2, ..., M\}$
- $N = V$ and $M \leq V$.
- Let l_v = Computer load of CU $v \in \mathcal{V}$ in FLOPS.
- Let L_{max} = Maximum capacity of computer node in FLOPS.
- Let C_m = Capacity of compute node $m \in \mathcal{M}$.
- Let δt = Time interval (epoch) for continuous traffic measurements
- Let T = Traffic Measurement Window (≤ 24 Hours), $0 \leq T \leq 24$

Research Problem

Problem Statement

Determine the minimum possible number of compute nodes needed to serve ‘N’ RRUs in each time interval δt with minimum disruption to users considering spatio-temporal traffic dynamics

Expected Solution

A flexible allocation and relocation schedule (Allocation Matrix) for CUs in each time interval δt that optimizes the aforementioned objective

- Architectural Considerations
  - 1-to-1 mapping between RRU and CU
  - Many-to-1 mapping between CUs and Compute Node
- Performance Metrics
  - Number of active computer nodes/servers
  - User Service Continuity
  - Scalability
  - Responsiveness

Traffic Model

- Cell load at RRU r at time t, $l_r(t)$ is given by,

$$l_r(t) = \sum_{u \in U_{\{r\}}(t)} \frac{No\_of\_PRBs\_Allocated\_to\_user\_u}{No\_of\_Available\_PRBs\_in\_cell} \tag{1}$$

- Weighted score metric ws_r gives a unified value by quantifying active user flows,

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

Notation	Definition	Weight
(N_voice)	GBR - Voice traffic flows	0.30
(N_game)	GBR -Real-Time Gaming	0.25
(N_conv)	GBR -Conversational Videos	0.20
(N_stream)	GBR - Live Streaming Video	0.15
(N_ngbr)	Non-GBR data flows	0.10

Processing Load Model

- The baseband processing time per subframe in microsecond ($proc(u, t)$) on a GPP server is given by,

$$proc(u, t) = r_{base} + p_{base} + u(mcs, prb) + u(r) \tag{3}$$

Notation	Definition
r_{base}	Constant cell offset
p_{base}	Platform dependent constant
$u(mcs, prb)$	User dependent processing
$u(r)$	Other user processing task

- We use FLoating point Operations Per Second (FLOPS) [1] as a measure of computer performance. Let us denote the maximum FLOPS limit as L_{max}. The compute load $l_v(t)$ for CU $v \in \mathcal{V}$ in FLOPS serving RRU $r \in \mathcal{N}$ is given by,

$$l_v(t) = L_{max} \times \left(\sum_{u \in U_{\{r\}}(t)} proc(u, t) \right) \tag{4}$$

---

[1] 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.

Cost Models

- The power consumption of a compute server with CPU utilization of u is given by,
  $$P_u = P_{idle} + (P_{cap} - P_{idle}) \times u \tag{5}$$
  As per SPECpower benchmark[2], average power consumption for a standard GPP server with CPU utilization of 100% is approximately 259 Watt.

---

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

[2] Performance and Energy Modeling for Live Migration of Virtual Machines, `https://doi.org/10.1007/s10586-011-0194-3`

Cost Models

- The power consumption of a compute server with CPU utilization of u is given by,

$$P_u = P_{idle} + (P_{cap} - P_{idle}) \times u \tag{5}$$

As per SPECpower benchmark[2], average power consumption for a standard GPP server with CPU utilization of 100% is approximately 259 Watt.

- The additional energy spent (in Watt-second) on live migration (reconfiguration)[3]is given by,

$$E_{Reconf} = 0.512 \times S + 20.165 \tag{6}$$

where S is the data volume (in MB) of CU to be transferred from source to target server to realize live migration of CU.

---

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

[2] Performance and Energy Modeling for Live Migration of Virtual Machines, `https://doi.org/10.1007/s10586-011-0194-3`

Integer Linear Programming (ILP) Formulation

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 a compute server at time t
$cost_{vm}$	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

Objective Function : *Minimize*

$$\left(\sum_{m=1}^{M} (z_m \times cost_m)\right) + \left(\sum_{\substack{v\in\mathcal{V}, m\in\mathcal{M},\\ \text{such that}\\ m\neq A_{(t-1)}(v)}} (\rho_v \times y_{vm} \times cost_{vm})\right) \qquad (7)$$

Constraints :

$$\sum_{m=1}^{M} y_{vm} = 1, \qquad \forall v \in \mathcal{V} \qquad (8)$$

$$\sum_{v=1}^{V} (y_{vm} \times l_v) \leq (C_m \times z_m), \qquad \forall m \in \mathcal{M} \qquad (9)$$

Numerical illustration

Server Capacity $= 100\%$, $\delta t = 1$, $T = 4$

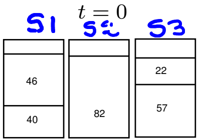
CU	l_v
1	40
2	46
3	57
4	82
5	22

$t = 0$

Numerical illustration

Server Capacity $= 100\%$, $\delta t = 1$, $T = 4$

CU	l_v
1	40
2	46
3	57
4	82
5	22

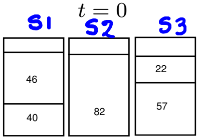


$t = 0$

Numerical illustration

Server Capacity $= 100\%$, $\delta t = 1$, $T = 4$

CU	l_v
1	40
2	46
3	57
4	82
5	22



$t = 0$

$$\begin{array}{c|cccc} & S1 & S2 & S3 & S4 \\ \hline C1 & 1 & 0 & 0 & 0 \\ C2 & 1 & 0 & 0 & 0 \\ C3 & 0 & 0 & 1 & 0 \\ C4 & 0 & 1 & 0 & 0 \\ C5 & 0 & 0 & 1 & 0 \end{array}$$

A_0

Numerical illustration

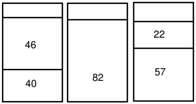
Server Capacity $= 100\%$, $\delta t = 1$, $T = 4$

CU	l_v
1	40
2	46
3	57
4	82
5	22

$t = 0$

CU	l_v
1	70
2	20
3	30
4	35
5	50

$t = 1$



$t = 0$

1	0	0	0
1	0	0	0
0	0	1	0
0	1	0	0
0	0	1	0

A_0

Numerical illustration

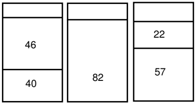
Server Capacity $= 100\%$, $\delta t = 1$, $T = 4$

CU	l_v
1	40
2	46
3	57
4	82
5	22

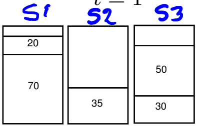
$t = 0$

CU	l_v
1	70
2	20
3	30
4	35
5	50

$t = 1$



$t = 0$



$t = 1$

S1	S2	S3	S4
1	0	0	0
1	0	0	0
0	0	1	0
0	1	0	0
0	0	1	0

A_0

Numerical illustration

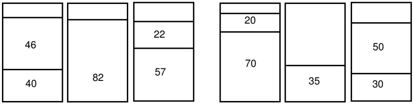
Server Capacity $= 100\%$, $\delta t = 1$, $T = 4$

CU	l_v
1	40
2	46
3	57
4	82
5	22

$t = 0$

CU	l_v
1	70
2	20
3	30
4	35
5	50

$t = 1$



$t = 0$ $t = 1$

1	0	0	0
1	0	0	0
0	0	1	0
0	1	0	0
0	0	1	0

A_0 $=$

1	0	0	0
1	0	0	0
0	0	1	0
0	1	0	0
0	0	1	0

A_1

→ No Relocations

Numerical illustration

Server Capacity $= 100\%$, $\delta t = 1$, $T = 4$

CU	l_v
1	40
2	46
3	57
4	82
5	22

$t = 0$

CU	l_v
1	70
2	20
3	30
4	35
5	50

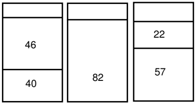
$t = 1$

CU	l_v
1	33
2	70
3	67
4	45
5	62

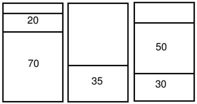
$t = 2$

$\} > 100\%$

$\rightarrow > 100\%$



$t = 0$



$t = 1$

→ Need 2 Relocations, with 1 more server.

→ Assume p_i's are same, relocate one with low $cost_{vm}$ $(\rightarrow l_v)$

1	0	0	0
1	0	0	0
0	0	1	0
0	1	0	0
0	0	1	0

A_0

1	0	0	0
1	0	0	0
0	0	1	0
0	1	0	0
0	0	1	0

A_1

Numerical illustration

Server Capacity $= 100\%$, $\delta t = 1$, $T = 4$

CU	l_v
1	40
2	46
3	57
4	82
5	22

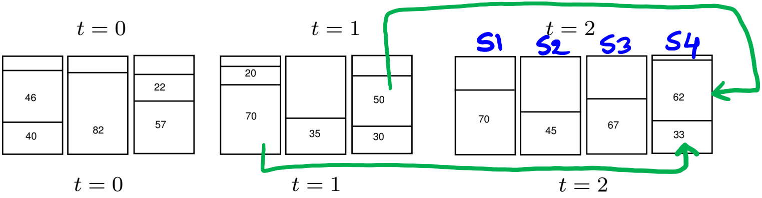
$t = 0$

CU	l_v
1	70
2	20
3	30
4	35
5	50

$t = 1$

CU	l_v	
1	33	✓
2	70	
3	67	
4	45	
5	62	✓

$t = 2$



1	0	0	0
1	0	0	0
0	0	1	0
0	1	0	0
0	0	1	0

A_0

1	0	0	0
1	0	0	0
0	0	1	0
0	1	0	0
0	0	1	0

A_1

Numerical illustration

Server Capacity $= 100\%$, $\delta t = 1$, $T = 4$

CU	l_v
1	40
2	46
3	57
4	82
5	22

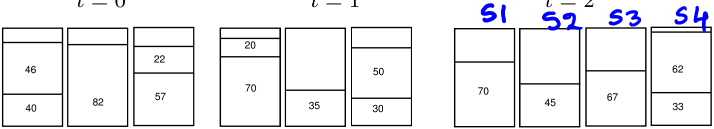
$t = 0$

CU	l_v
1	70
2	20
3	30
4	35
5	50

$t = 1$

CU	l_v	
1	33	✓
2	70	
3	67	
4	45	
5	62	✓

$t = 2$



$t = 0$ $t = 1$ $t = 2$

1	0	0	0
1	0	0	0
0	0	1	0
0	1	0	0
0	0	1	0

A_0

1	0	0	0
1	0	0	0
0	0	1	0
0	1	0	0
0	0	1	0

A_1

0	0	0	1
1	0	0	0
0	0	1	0
0	1	0	0
0	0	0	1

A_2

Numerical illustration

Server Capacity $= 100\%$, $\delta t = 1$, $T = 4$

CU	l_v
1	40
2	46
3	57
4	82
5	22

$t = 0$

CU	l_v
1	70
2	20
3	30
4	35
5	50

$t = 1$

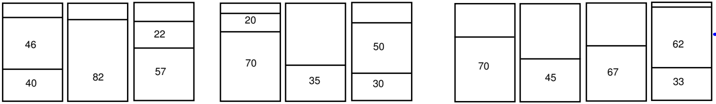
CU	l_v
1	33
2	70
3	67
4	45
5	62

$t = 2$

CU	l_v
1	60
2	30
3	27
4	25
5	32

$t = 3$

$\}< 100\%$



$t = 0$ $t = 1$ $t = 2$

→ chance for Consolidation, saves 2 servers

→ Relocate ones with low lvs

1	0	0	0
1	0	0	0
0	0	1	0
0	1	0	0
0	0	1	0

A_0

1	0	0	0
1	0	0	0
0	0	1	0
0	1	0	0
0	0	1	0

A_1

0	0	0	**1**
1	0	0	0
0	0	1	0
0	1	0	0
0	0	0	**1**

A_2

Numerical illustration

Server Capacity $= 100\%$, $\delta t = 1$, $T = 4$

CU	l_v
1	40
2	46
3	57
4	82
5	22

$t = 0$

CU	l_v
1	70
2	20
3	30
4	35
5	50

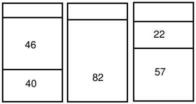
$t = 1$

CU	l_v
1	33
2	70
3	67
4	45
5	62

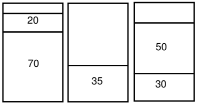
$t = 2$

CU	l_v
1	60
2	30
3	27
4	25
5	32

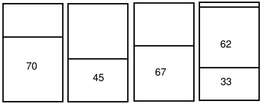
$t = 3$



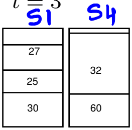
$t = 0$



$t = 1$



$t = 2$



$t = 3$

1	0	0	0
1	0	0	0
0	0	1	0
0	1	0	0
0	0	1	0

A_0

1	0	0	0
1	0	0	0
0	0	1	0
0	1	0	0
0	0	1	0

A_1

0	0	0	**1**
1	0	0	0
0	0	1	0
0	1	0	0
0	0	0	**1**

A_2

Numerical illustration

Server Capacity $= 100\%$, $\delta t = 1$, $T = 4$

CU	l_v
1	40
2	46
3	57
4	82
5	22

$t = 0$

CU	l_v
1	70
2	20
3	30
4	35
5	50

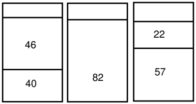
$t = 1$

CU	l_v
1	33
2	70
3	67
4	45
5	62

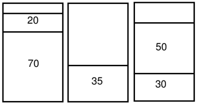
$t = 2$

CU	l_v
1	60
2	30
3	27
4	25
5	32

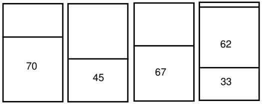
$t = 3$



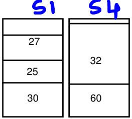
$t = 0$



$t = 1$



$t = 2$



$t = 3$

1	0	0	0
1	0	0	0
0	0	1	0
0	1	0	0
0	0	1	0

OFF

A_0

1	0	0	0
1	0	0	0
0	0	1	0
0	1	0	0
0	0	1	0

OFF

A_1

0	0	0	**1**
1	0	0	0
0	0	1	0
0	1	0	0
0	0	0	**1**

ON

A_2

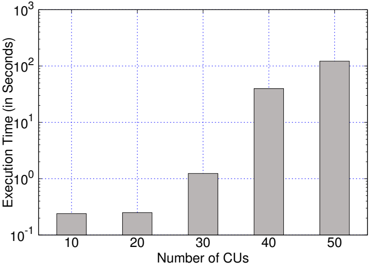
0	0	0	1
1	0	0	0
1	0	0	0
1	0	0	0
0	0	0	1

OFF OFF

A_3

ILP Execution Time

- Input size from 10 to 50 CUs
- Maximum execution time taken by the ILP model over 240 iterations
- 122 seconds ($\sim$ 2 minutes) to converge to the solution for 50 CUs.



Heuristics?

To alleviate the computational heaviness for larger input size, we can look forward to time-efficient heuristics approaches that produces solutions close to the optimal with acceptable service guarantees.

Relocation-Aware Heuristic Algorithm for KORA

Three distinct stages for every **overloaded** compute server.

1. **Candidate CU Selection :** Selecting a suitable candidate CU for relocation from an overloaded compute server (identified as source server). 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 $v \in \mathcal{V}$ and is a weighted average of ws_v and lv.

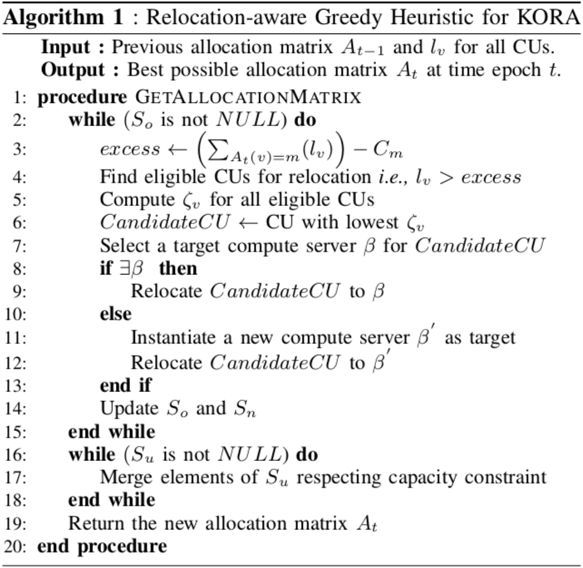
$$\zeta_v = (\alpha \times ws_v) + ((1-\alpha) \times l_v), \text{ such that } 0 \leq \alpha \leq 1 \tag{10}$$

2. **Determining Target Server :** 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. We use a variant of Best Fit (BF) bin packing approximation algorithm to identify a target for candidate CU. Instantiates a new compute server in case there are no existing non-overloaded compute server to accommodate candidate CU.

3. **Perform CU relocation :** Iteratively write the active memory pages/contexts of candidate CU from source to target compute server. (live migration)

Proposed Heuristic Algorithm

- Relocation of CUs from **overloaded** compute servers.
- Consolidation of CUs in **underloaded** compute servers.



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.

```
procedure GETALLOCATIONMATRIX
    while ($S_o$ is not $NULL$) do
        $excess \leftarrow \left(\sum_{A_t(v)=m}(l_v)\right) - C_m$
        Find eligible CUs for relocation i.e., $l_v > excess$
        Compute $\zeta_v$ for all eligible CUs
        $CandidateCU \leftarrow$ CU with lowest $\zeta_v$
        Select a target compute server $\beta$ for $CandidateCU$
        if $\exists\beta$ then
            Relocate $CandidateCU$ to $\beta$
        else
            Instantiate a new compute server $\beta'$ as target
            Relocate $CandidateCU$ to $\beta'$
        end if
        Update $S_o$ and $S_n$
    end while
    while ($S_u$ is not $NULL$) do
        Merge elements of $S_u$ respecting capacity constraint
    end while
    Return the new allocation matrix $A_t$
end procedure
```

Simulation Parameters

Parameter	**Value**
Network Area	10 KM × 10 KM
Number of RRUs	100
Users	1000
Tx Power of RRU	1 Watt
Sampling Interval	6 Minutes
Total Traffic Capture Window	24 Hours
Total Generated Samples	240
RRU workload Range	Normalized in [0,1]
Peak CU Compute Load	100%
$[w_1, w_2, w_3, w_4, w_5]$	$[0.10, 0.30, 0.20, 0.25, 0.15]$
Traffic generation process	Gaussian Mixture Model

Results .. 1

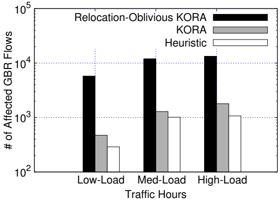


Figure: Total Number of Affected GBR Flows

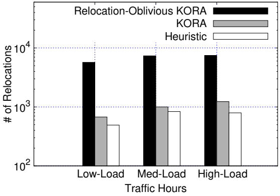


Figure: Total Number of CU Relocations.

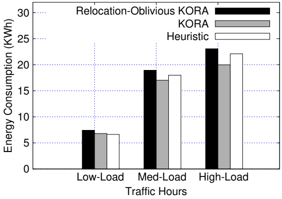
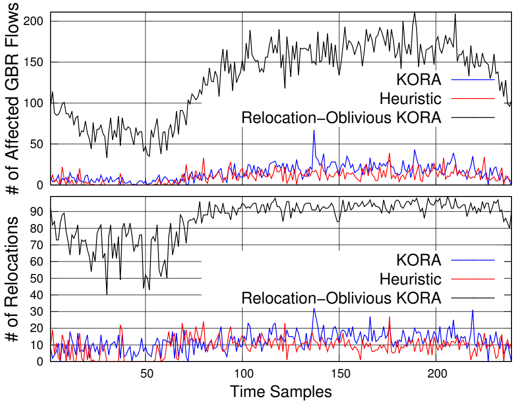


Figure: Total Energy Consumption (KWh).

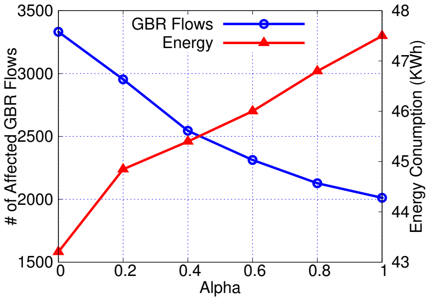
- *Relocation-Oblivious* KORA only focuses on minimizing the total energy consumption due to active compute servers and does not factor the relocation cost. Therefore, it incurs disruption to a large number of GBR flows ($\sim$ 13296 in High_Load) in all the three traffic scenarios.
- KORA is able to outperform *relocation-oblivious* scheme by saving 88.53% of affected GBR flows.
- The number of active relocations incurred are 85.74% less than that of relocation occurred with *relocation-oblivious* scheme.

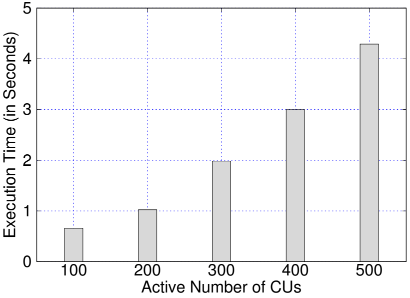
Results .. 2



- Linearly 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.

Results .. 3





- 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.
- In contrast to the execution time of ILP model, heuristic is light-weight and executes in order of few seconds.

Conclusions

- Different trade-offs involved in dynamic consolidation and relocation of CUs in C-RAN is studied in a novel and efficient resource management framework, KORA.

Conclusions

- Different trade-offs involved in dynamic consolidation and relocation of CUs in C-RAN is studied in a novel and efficient resource management framework, KORA.
- 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.

Conclusions

- Different trade-offs involved in dynamic consolidation and relocation of CUs in C-RAN is studied in a novel and efficient resource management framework, KORA.
- 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.
- A scalable, time-efficient relocation-aware heuristic algorithm is proposed.

Conclusions

- Different trade-offs involved in dynamic consolidation and relocation of CUs in C-RAN is studied in a novel and efficient resource management framework, KORA.
- 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.
- A scalable, time-efficient relocation-aware heuristic algorithm is proposed.
- The proposed heuristic algorithm saves 27% of relocations and 33% of GBR flows from disruption, but consumes 6.6% more energy than KORA.

Conclusions

- Different trade-offs involved in dynamic consolidation and relocation of CUs in C-RAN is studied in a novel and efficient resource management framework, KORA.
- 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.
- A scalable, time-efficient relocation-aware heuristic algorithm is proposed.
- The proposed heuristic algorithm saves 27% of relocations and 33% of GBR flows from disruption, but consumes 6.6% more energy than KORA.
- **Future work**: prototyping C-RAN system using OAI for different split options and factoring split-specific constraints in optimization models.

Acknowledgements

This work was supported by the project ”Converged Cloud Communication Technologies”

सत्यमेव जयते

Ministry of Electronics and Information Technology
Government of India

References I

China Mobile, *C-RAN - The road towards green RAN,*, ,*China Mobile White Paper*, 2011.

Manli Qian and Wibowo Hardjawana and Jinglin Shi, and Branka Vucetice, *Baseband Processing Units Virtualization for Cloud Radio Access Networks*, ,*IEEE WIRELESS COMMUNICATIONS LETTERS, VOL. 4, NO. 2, APRIL 2015*

Sourjya Bhaumik et al, *CloudIQ: A Framework for Processing Base Stations in a Data Center*, ,*IEEE Mobicom, 2012*

Nikaein, Navid, *"Processing Radio Access Network Functions in the Cloud: Critical Issues and Modeling"*, *Proceedings of the 6th International Workshop on Mobile Cloud Computing and Services, 3643*, 2004

White paper, *"Cisco Visual Networking Index, Global mobile data traffic forecast update, 2013–2018"*, *Online*, 2014.

Hong Xu and Baochun Li *Anchor: A Versatile and Efficient Framework for Resource Management in the Cloud*, ,*IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS,VOL. 24,NO. 6,JUNE 2013*

Gale, David, and Lloyd S. Shapley. *College admissions and the stability of marriage*, ,*The American Mathematical Monthly 69.1 (1962): 9-15.*

Navid Nikaein et al. *Demo: Closer to Cloud-RAN: RAN as a Service*, ,*International Conference on Mobile Computing and Networking, Mobicom 2015, Pages 193-195*

THANK YOU

QUERIES ?