Base Band Unit Function Split Placement in Cloud Radio Access Networks: Mathematical Programming Approach

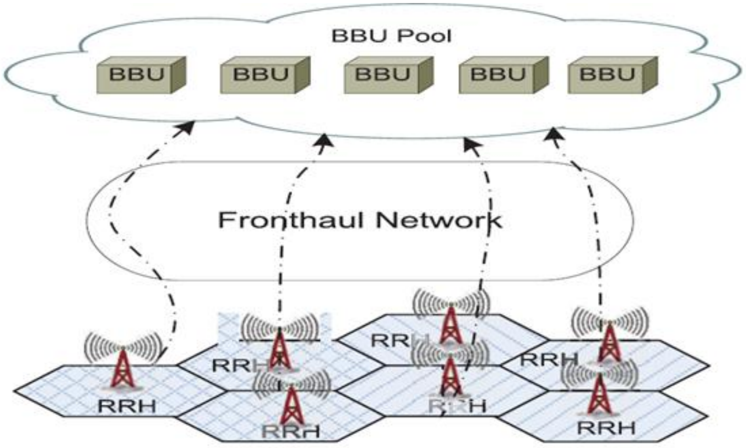
***Niezi Mharsi*[1,2]** *,Makhlouf Hadji[1], William Diego[3] and Ruby Krishnaswamy[3]*

[1] Institut de Recherche Technologique SystemX, Paris-Saclay, France
[2] Institut Mines Telecom – Telecom ParisTech, Paris, France
[3] Orange Labs, Chatillon, France

Outline

I. Motivation

Cloud Radio Access Network : Cloud-RAN



Cloud-RAN Architecture :

Three main components

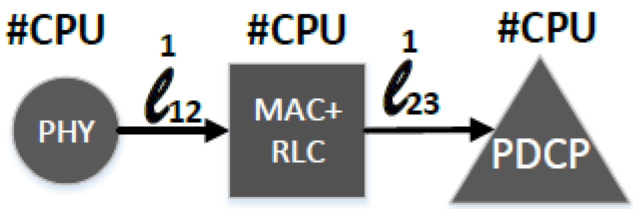
- ✓ BBU pool is composed of BBUs which operate as virtual base stations.
- ✓ RRHs is a set of antennas located at the remote sites.
- ✓ Fronthaul Network connects the RRHs to the BBU pool and requires high bandwidth and low-latency.

Cloud-RAN Benefits :

- ✓ Meeting the exponential growth in data traffic demand.
- ✓ Reduce the capital and operating expenditures.
- ✓ Improve the capacity and the coverage of mobile communication systems.

II. Problem Statement (1/3)

BBU Function Split & Placement in Cloud-RAN

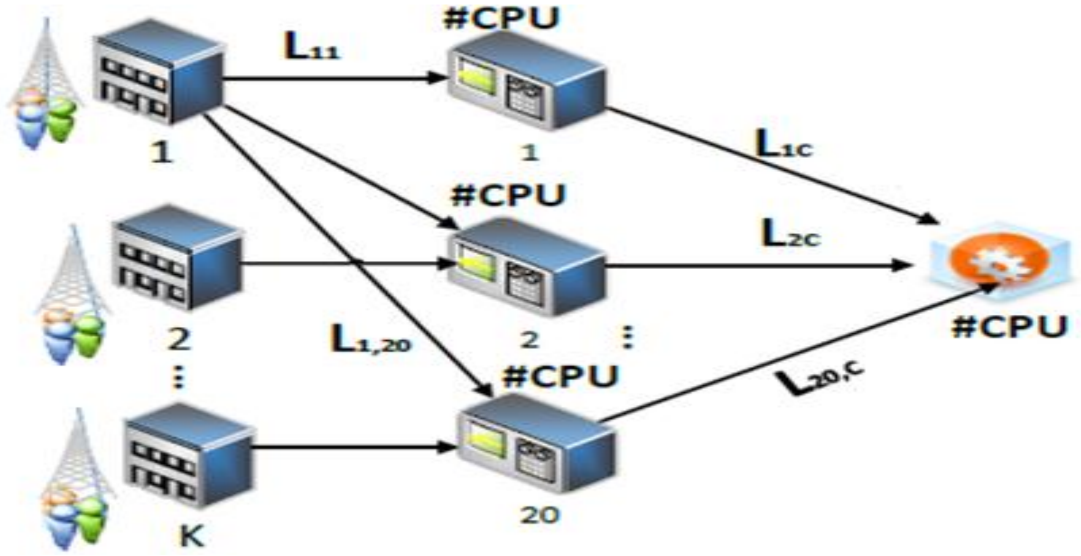


3GPP split option

- ✓ BBU functions are split into three (PHY, MAC+RLC, and PDCP) connected layers.
- ✓ This split is outlined as the best option in 3GPP meeting.

II. Problem Statement (2/3)

BBU Function Split & Placement in Cloud-RAN



Antennas & Embedded Clouds

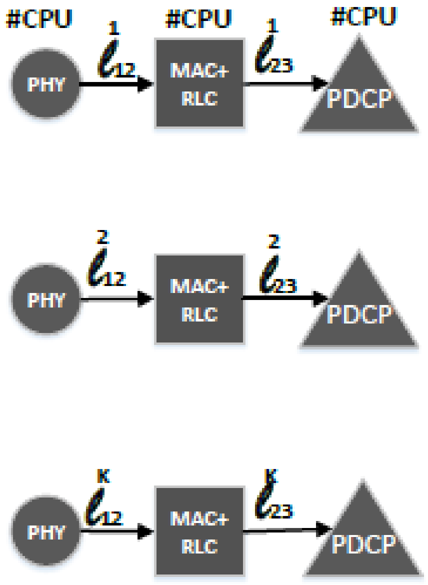
Edge Clouds

Centralized Core Cloud

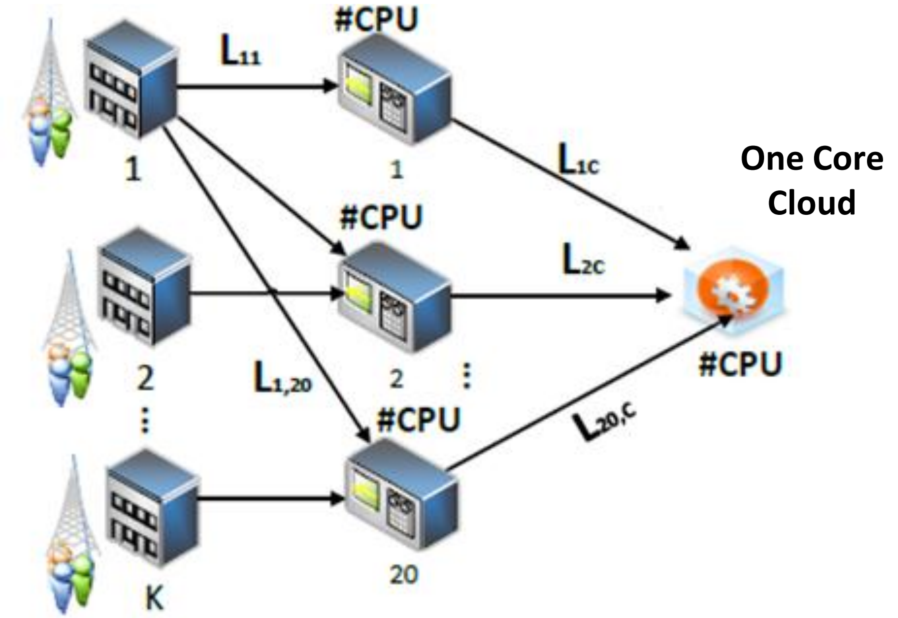
Network Infrastructure

II. Problem Statement (3/3)

BBU Function Split & Placement in Cloud-RAN



One aggregate demand per antenna



Antennas/Embedded Cloud (20 – 500)

Edge Cloud (10 – 20)

System Model

Problem Complexity

The BBU function split & placement problem has some similarities to the Virtual Network Embedding (VNE) problem : By relaxing sequencing order constraints between the layers of each chain, the problem becomes NP-hard, so does the BBU function split placement.

III. Mathematical Approach (2/3)

Integer Linear Program (for Small and Medium size network)

$$\min \sum_{k\in\mathscr{A}} \sum_{j\in\mathscr{V}} \sum_{j'\in\mathscr{P}(j)} \sum_{i\in\{1,2\}} L_{(j,j')} y^k_{(i,i+1);(j,j')} - \sum_{j\in\mathscr{V}} \left(C_j z_j - \sum_{k\in A} \sum_{i\in\mathscr{V}_v} c^k_i x^k_{i,j} \right).$$

$$\sum_{j\in\mathscr{V}_1} \mathbb{1}_{(k,j)} x^k_{1,j} = 0, \forall k \in \mathscr{A}$$

$$\sum_{j\in\mathscr{V}} x^k_{i,j} = 1, \forall k \in \mathscr{A}, \forall i \in \{1,2,3\}$$

$$\sum_{k\in\mathscr{A}} \sum_{i\in\{1,2,3\}} x^k_{i,j} \times c^k_i \leq C_j, \forall j \in \mathscr{V}$$

$$x^k_{i,j} \leq \sum_{j'\in\mathscr{P}(j)} x^k_{i+1,j'}, \forall k \in \mathscr{A}, \forall i \in \{1,2\}, \forall j \in \mathscr{V}$$

$$\sum_{j\in\mathscr{V}} y^k_{(i,i+1);(j,j')} = x^k_{i+1,j'}, \forall k \in \mathscr{A}, \forall i \in \{1,2\}, \forall j' \in \mathscr{P}(j)$$

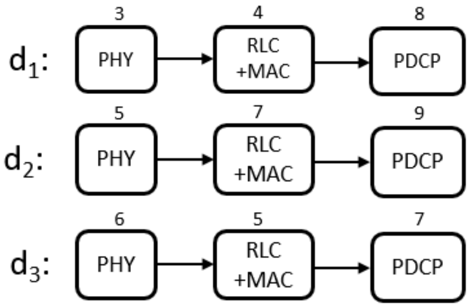
$$\sum_{j'\in\mathscr{P}(j)} y^k_{(i,i+1);(j,j')} = x^k_{i,j}, \forall k \in \mathscr{A}, \forall i \in \{1,2\}, \forall j \in \mathscr{V}$$

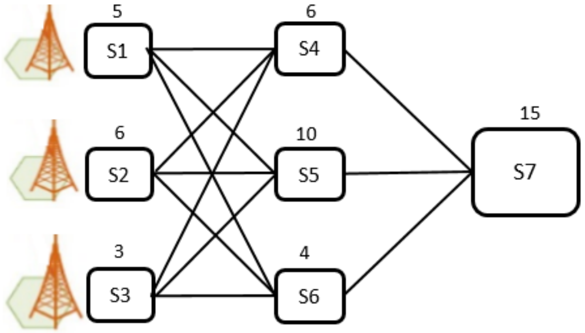
$$\sum_{j\in\mathscr{V}} \sum_{j'\in\mathscr{P}(j)} y^k_{(i,i+1);(j,j')} = 1, \forall k \in \mathscr{A}, \forall i \in \{1,2\}$$

$$L_{(j,j')} \times y^k_{(i,i+1);(j,j')} \leq l^k_{(i,i+1)},\ \forall k \in \mathscr{A}, \forall i \in \{1,2\}, \forall j \in \mathscr{V}, \forall j' \in \mathscr{P}(j)$$

$$x^k_{i,j} \leq z_j, \forall j \in \mathscr{V}, \forall k \in \mathscr{A}, \forall i \in \mathscr{V}_v$$

Heuristic Approach : Multi-stage Graph Algorithm



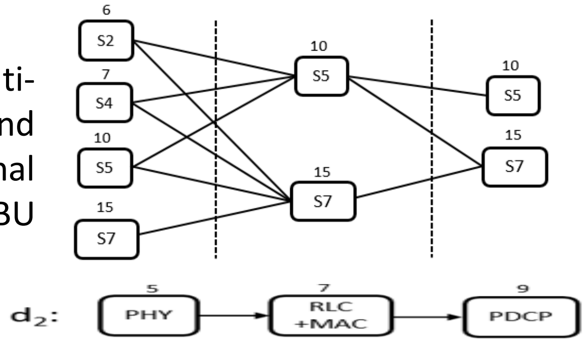


Step 1:

Sort all the requested demands according to the total amount of requested CPU(In this example, start by d_2)

Step 2 :

Create the multi-stage graph and find the optimal mapping of BBU functions



Step 3 :

- ✓ If the demand d_2 is deployed, update the number of CPU cores.
- ✓ If the demand is not deployed, then the demand (total graph of all the chains) is rejected.

A multi-stage approach example

IV. Numerical Results (1/3)

Algorithms Performance Comparison : ILP Vs Heuristic variants

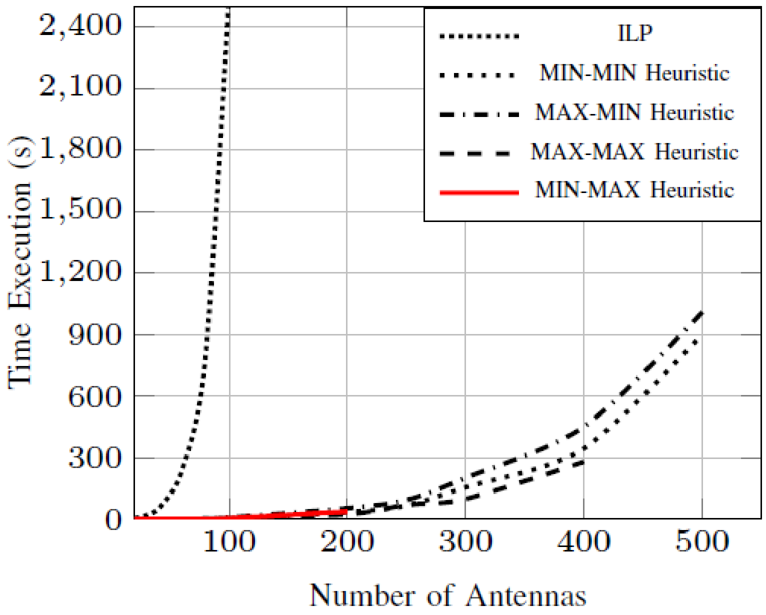
#Antennas	#Edge Clouds	Variant	Euclidean Graph Cost Gap (%)	Random Graph Cost Gap (%)
60	10	min-min	0	3.07
		min-max	0	0
		max-min	2.96	2.62
		max-max	0	0
	15	min-min	0	2.79
		min-max	0	0
		max-min	2.24	2.22
		max-max	0	0
	20	min-min	1.88	2.11
		min-max	0	0
		max-min	2.05	1.94
		max-max	0	0
80	10	min-min	2.49	3.08
		min-max	0	0
		max-min	2.64	2.56
		max-max	0	0
	15	min-min	2.92	2.76
		min-max	0	0
		max-min	2.3	2.32
		max-max	0	0
	20	min-min	2.55	2.28
		min-max	0	0
		max-min	1.87	2.0
		max-max	0	0

TABLE I
ALGORITHMS PERFORMANCE COMPARISON : ILP VS HEURISTIC VARIANTS

#Antennas	#Edge Clouds	Heuristic Variant	Heuristic Execution Time (s)	ILP Execution Time
100	10	min-min	2.25	29.11min
		min-max	2.49	
		max-min	4.35	
		max-max	2.87	
	15	min-min	2.98	33.47min
		min-max	3.26	
		max-min	6.17	
		max-max	4.00	
	20	min-min	4.19	42.85min
		min-max	3.98	
		max-min	7.99	
		max-max	4.86	
200	10	min-min	42.51	>12h
		min-max	-	
		max-min	82.15	
		max-max	35.43	
	15	min-min	<1min	>15h
		min-max	-	
		max-min	<2min	
		max-max	43.01s	
	20	min-min	<1min	>16h
		min-max	<1min	
		max-min	<2min	
		max-max	<1min	

TABLE II
SCALABILITY AND CONVERGENCE TIME COMPARISON USING EUCLIDEAN GRAPHS

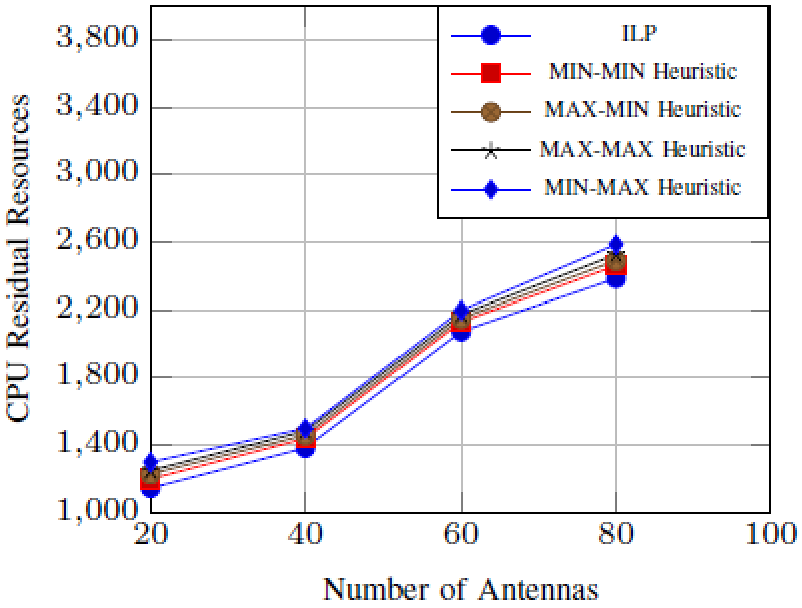
Algorithms Performance Comparison : ILP Vs Heuristic variants



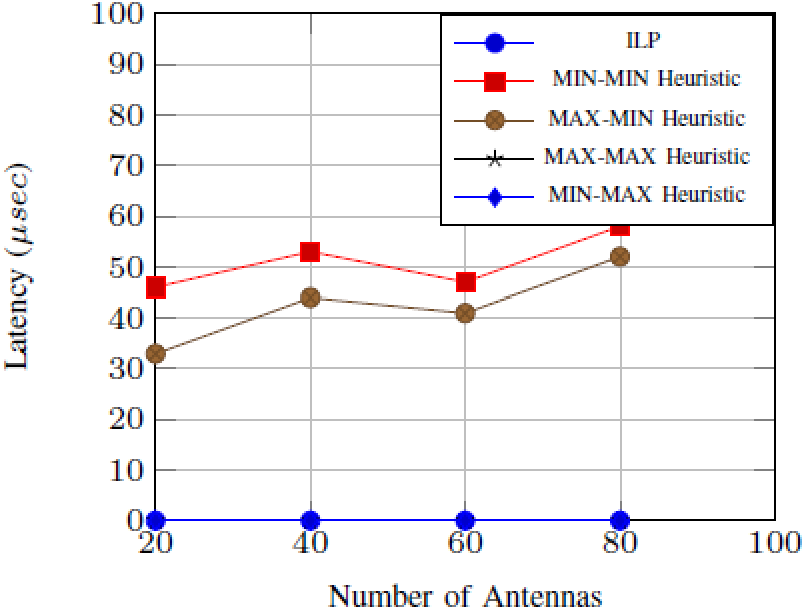
Algorithms' convergence time for random graphs with 20 edge nodes

IV. Numerical Results (3/3)

Algorithms Performance Comparison : ILP Vs Heuristic variants



ILP Vs Heuristic : CPU Residual resources behavior



ILP Vs Heuristic : Latency behavior

V. Conclusion

- ✓ For BBU function split placement problem, we have proposed an optimization which can be solved by ILP.
- ✓ For a large network size, we have proposed an heuristic algorithms based on the construction of an extended multi-stage graph.
- ✓ The heuristic approach provides a near optimal or optimal solution in a negligible time even for large instances.

THANKS FOR

YOUR ATTENTION

niezi.mharsi@irt-system.fr