Presented at ICC - FlexNets 2017, Paris, FR, 20th May 2017 – full paper here

Flexible dynamic Coordinated Scheduling in Virtual-RAN deployments

N. Iardella(1,2), G. Nardini(2), G. Stea(2), A. Virdis(2), A. Frangioni(3), L. Galli(3), D. Sabella(4), F. Mauro(4), G. Dell'Aera(4), M. Caretti(4)

(1) DINFO, University of Florence, Italy

(2) Dip. Ing. dell'Informazione, University of Pisa, Italy

(3) Dip. di Informatica University of Pisa, Italy

(4) TIM (Telecom Italia Group), Turin, Italy

***Abstract*—Using Coordinated Scheduling (CS), eNodeBs in a cellular network dynamically agree on which Resource Blocks (not) to use, so as to reduce the interference, especially for cell-edge users. This paper describes a software framework that allows dynamic CS to occur among a relatively large number of nodes, as part of a more general framework of network management devised within the Flex5Gware project. The benefits of dynamic CS, in terms of spectrum efficiency and resource saving, are illustrated by means of simulation and with live measurements on a prototype implementation using virtualized eNodeBs.**

***Keywords*—Coordinated Scheduling, CoMP, Virtual RAN**

I. Introduction

Future cellular networks will be *dense* and *heterogeneous* ones, with macro base stations (eNodeBs or eNBs) used to achieve ubiquitous coverage, and possibly many *micro* eNBs used to increase the capacity per unit of surface. In such a system, the major performance-limiting factor is inter-cell interference, mainly *from* the macro(s) *to* the users attached to the micros, but possibly also from nearby micros. Coordinated Scheduling (CS) is a Coordinated MultiPoint (CoMP) techniques, by which eNBs dynamically agree on which frequencies (not) to use, so as to reduce the interference for cell-edge User Equipments (UEs). The progress in the coordination capabilities is also fostered by the emergence of flexible hardware-software infrastructures and new paradigms, such as the Virtual Radio Access Network (V-RAN). V-RAN employs general-purpose hardware and implement the eNBs protocol stack in a virtualized environment (thus by decoupling HW from SW) to pool a significant number (i.e., hundreds) of cells. This way, the baseband resources can be centralized in a so-called "cloud"-RAN (C-RAN) deployment. Operators recently dedicated an increased attention to C-Ran and V-RAN, especially because it abates the cost to manage, maintain, and expand the RAN. With reference to CS operation, a V-RAN architecture allows operators to have cells as virtual machines in a data center, and inter-cell communication as low-latency, high-bandwidth inter-VM local communication. Moreover, it allows them to leverage the abundance of computing power typical of a data center to run complex algorithms.

Most existing CS schemes (e.g., [2]-[4]) are *static*, meaning that they assume a *fixed*, long-term partitioning of resources among potentially interfering clusters of eNBs. A static partitioning is inflexible, hence unsuitable when the traffic varies at a fast pace: for instance, a UE with a high bandwidth demand moving from one micro cell to a coordinated nearby one, will be unable to reap the full capacity of both cells due to the fact that neither is allowed to use the entire spectrum at any time, even if the network is unloaded. Scenarios with unpredictable, localized traffic bursts are poorly supported by any static scheme. *Dynamic* CS schemes have been proposed, e.g., [5]-[8]. However, they require information which is unavailable in current cellular networks: for instance, they require UEs to measure and report their SINR with/without some of the major interferers. Moreover, some of these schemes (e.g., [7],[8]), request *per-UE* information to be conveyed to a central coordinator, which then allocates resources to single UEs in a multicell area. Such schemes scale poorly with the number of UEs.

The Flex5Gware EU-5GPPP project [1], belonging to phase 1 of the 5G Infrastructure Public Private Partnership (in short 5G PPP), aims at delivering reconfigurable HW platforms *and* HW-agnostic software to achieve higher capacity and increased energy efficiency and maximize flexibility in the transition to 5G wireless systems. The project encompasses a wide range of building blocks, from antennas to software architectures. One of the threads of this project aims at researching flexible, effective and efficient resource allocation mechanisms in a V-RAN environment, that allow an operator to enhance performance and save energy. This paper presents the software framework underlying the above-mentioned research, which implements a CS server that coordinates virtualized eNBs at a fast pace. We describe the framework, highlighting its flexibility, and the dynamic CS scheduling algorithm that lies at its core. The latter relies only on *standardized* reports, and makes decisions *per eNB*, as opposed to *per UE*, thus improving scalability. The algorithm relies on solving an optimization problem to optimality: however, thanks to a clever problem modeling, this takes few milliseconds for deployments of up to ten eNBs, which is consistent with the requirements of an environment where sudden traffic variations have to be accommodated. The evaluation of our framework takes place using two complementary techniques: simulation, to test for scalability, and prototyping, to validate the results in realistic settings. Our simulations show that CS allows an operator to save a considerable amount of radio resources to serve the same traffic: on one hand, this means a significant increase in the spectrum (hence energy) efficiency, since the energy consumed by the network depends on the number of occupied radio Resource Blocks (RBs). On the other hand, CS increases cell-edge user data rate, by contributing to increase the average capacity of the network, since more resources are available for new users. Moreover, having a *dynamic* CS allows you to deploy capacity *where and when* it is actually necessary. Measurements taken on our prototype confirm and validate our simulation results,

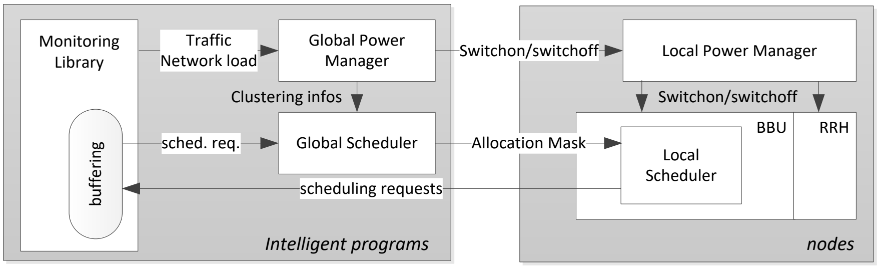

Fig. 1. Software framework for intelligent network management

$$\begin{array}{c|cccccc}
 & 1 & 2 & 3 & \dots & C & \\
\hline
1 & 0 & 1 & 1 & \dots & 0 & \\
2 & 1 & 1 & 0 & \dots & 1 & \mathbf{p} \\
3 & 1 & 0 & 0 & \dots & 1 & \\
\dots & \dots & \dots & \dots & \dots & \dots & \\
M & 0 & 1 & 1 & \dots & 0 & \\
 & & \mathbf{R}_2 & & & &
\end{array}$$

Fig. 2. Allocation masks and ownership vectors

albeit on a smaller scale, showing that the channel quality perceived by users connected to micros sensibly increases, despite the high interference from the macro. To the best of our knowledge, ours is the first live prototype to demonstrate this.

The rest of the paper is organized as follows: Section II describes the software framework and the CS algorithm, whereas Section III reports performance evaluation results obtained via simulation. Section IV describes our testbed implementation using virtualized eNBs. In Section V we conclude the paper and highlight directions for future work.

II. Flexible Coordinated Scheduling

We first introduce the components of our software framework, and then detail our CS algorithm.

A. *Software framework*

Our software framework, outlined in Fig. 1 and derived from the general Flex5Gware software architecture [1], adds an *intelligent program* layer on top of the *nodes* (i.e., eNBs, either macro or micro). That layer, whose perspective is network-wide, includes a *Monitoring Library* (ML), which stores the information to be used by the other components, a *Global Scheduler* (GS), which embodies the CS, and a *Global Power Manager* (GPM), whose role is to compute the most energy-efficient network configuration that allows the current load to be carried. The GPM makes decisions at a relatively slow pace (e.g., tens of minutes), whereas the GS works at subsecond timescales. A node is proxied by its own *Local Power Manager* (LPM), i.e. a process which is always on (e.g., in a cloud server). The LPM knows how to switch on/off both the Remote Radio Head (RRH) and the Base Band Unit (BBU) of its node, regardless of whether the latter resides on a physical or virtual machine. When the GPM wants to switch on/off a node, it contacts its LPM and instructs it to do so.

Nodes that are switched on register themselves as such on the ML, and start sending their *Scheduling Requests* (SR) to it. The latter are the number of Resource Blocks (RBs) required to clear their backlog in the current Transmission Time Interval (TTI). Periodically, each node *i* receives an *Allocation Mask* (AM) from the GS. As shown in Fig. 2, the latter is a binary *M*-vector, $\mathbf{R}_i$, where $\mathbf{R}_i[x]=1$ means that node *i can* include RB *x* in its schedule, and cannot use it otherwise. The fact that each node is sent a different AM enforces CS. Nodes check for a new AM on each TTI, just before scheduling their UEs.

A GS is a process started by the GPM, and coordinates a *cluster* of nodes. The clustering is done by the GPM itself, based on the position and radiation information about the nodes (all of which are static and stored in the ML). Therefore, there can be several GS instances active at any time, each one coordinating one cluster. The GS polls the ML periodically, requesting a *time average* of the SRs of the nodes in its cluster (e.g., on the last 100ms). The ML performs the time average and returns the reply. The GS is aware of the average inter-cell interference that node *i* does on node *j*'s users, and uses that information to produce a schedule for all the nodes in its cluster that minimizes the overlapping of RBs for the nodes with the highest mutual interference. When the schedule is completed, the GS prepares the AM for the nodes and sends it to them.

Note that the presence of a ML, interposed between the GS and the nodes, solves or alleviates (de)synchronization problems among the various entities involved: in fact, if TTI boundaries are not perfectly aligned at the various nodes, the worst that can happen is that the GS will receive as inputs time averages taken on slightly different intervals, whose relative offset is at most one TTI. Similarly, a node may acknowledge a new AM one TTI later than another. However, given that an AM is typically changed every 100ms or so, this has a negligible impact on the overall performance.

B. *Coordinated Scheduling algorithm*

The algorithm run by the GS minimizes the overall interference, i.e. the sum of the overlapping RBs between all pairs of cells *i,j*, weighted by the respective *interference coefficients* (*ICs*) $\alpha_{i,j}$. The latter can be inferred through on-field measurements or ray-tracing simulation, and represent the amount of interference that UEs of cell *j* will hear from cell *i*. Since cells may be anisotropic, it is in general $\alpha_{i,j} \neq \alpha_{j,i}$.

There are, of course, several different models that one may devise for this to happen. We first describe a simple, but inefficient model, and then present a more efficient one. Call $\mathbf{C}$ the cluster, with $C=|\mathbf{C}|$, and A_i the scheduling requests for cell i. The former is communicated to the GS by the GPM that instantiates it, and the latter are obtained periodically when the GS polls the ML. An elegant formulation of the optimization problem to be solved by the GS is the following:

$$\begin{aligned}
&\min \sum\nolimits_{i,j} \alpha_{i,j} \cdot \left\langle \mathbf{R}_i, \mathbf{R}_j \right\rangle \\
&s.t. \quad \sum_{x=1}^{M} \mathbf{R}_i[x] \geq A_i, \quad i \in \mathbf{C} \qquad (i) \qquad\qquad (1) \\
&\qquad\quad \mathbf{R}_i[x] \in \{0,1\}, \quad i \in \mathbf{C}, 1 \leq x \leq M \quad (ii)
\end{aligned}$$

The objective is to minimize the amount of overlapping RBs, weighted by the ICs. Notation $\left\langle \mathbf{R}_i, \mathbf{R}_j \right\rangle$ represents the inner product of AMs $\mathbf{R}_i$ and $\mathbf{R}_j$. Constraint (*i*) states that the sum of RBs allocated to cell *i* must be no fewer than its allocated ones A_i. Note that equality will hold in (*i*) at the optimum in any case, since this is a minimization problem. Constraint (*ii*) defines the problem variables to be binary. Problem (1) is a Quadratic Assignment Problem (QAP), which is notoriously hard to solve at optimality *and* nonlinear. It can be linearized by introducing *overlap vectors* $\mathbf{O}_{i,j}$, i.e. binary vectors such that $\mathbf{O}_{i,j}[x] = \mathbf{R}_i[x] \cdot \mathbf{R}_j[x]$. This yields the following:

$$
\begin{array}{lll}
\min \sum_{i,j} \alpha_{i,j} \cdot \sum_{x=1}^{M} \mathbf{O}_{i,j}[x] & & \\
s.t. & & \\
\mathbf{O}_{i,j}[x] \geq \mathbf{R}_i[x] + \mathbf{R}_j[x] - 1 & i,j \in \mathbf{C}, j \neq i, 1 \leq x \leq M & (i) \\
\sum_{x=1}^{M} \mathbf{R}_i[x] \geq A_i & i \in \mathbf{C} & (ii) \\
\mathbf{R}_i[x] \in \{0,1\} & i \in \mathbf{C}, 1 \leq x \leq M & (iii) \\
\mathbf{O}_{i,j}[x] \in \{0,1\} & i,j \in \mathbf{C}, j \neq i, 1 \leq x \leq M & (iv)
\end{array}
$$

Constraint (*i*) is the linear version of the product (or logical AND) between $\mathbf{R}_i[x]$ and $\mathbf{R}_j[x]$, and the rest remains unchanged. The above is a Mixed Integer-linear Problem (MILP). However, introducing overlap vectors makes the number of required binary variables scale as $O(M \cdot C^2)$: a cluster of 10 cells using 100 RBs requires as many as 10^4 variables. Moreover, a major disadvantage of the above model is *symmetry*: any optimal solution of the above model remains optimal if we apply the same permutation to all the AMs, i.e. there are $M!$ equivalent optimal solutions. A solver will have to find them all before establishing that any one of them is optimal, which makes solving this model at optimality very costly. This last problem can be dispensed with by recognizing that the *position* of a RB in an AM is immaterial. In fact, only the *ownership* of that RB (i.e., which cells are using it) determines the inter-cell interference. This can be leveraged to compute the optimal solution much more quickly.

Instead of focusing on AMs, let us switch the focus on RBs, and define the *ownerships* of a generic RB as a *C*-vector of binaries, i.e., $[0,1,1,0,...,0,1]$, meaning that this RB is allocated simultaneously in the AMs of cells 2, 3, and *C*. With reference to the example of Fig. 2, columns are the AMs, and rows are ownership vectors. Let $\mathbf{P}$ be the set of *possible* ownership vectors, hence $P = |\mathbf{P}| = 2^C$. Enumerate all vectors $\mathbf{p} \in \mathbf{P}$, and consider integer variable $x_{\mathbf{p}} \geq 0$, which counts the *number of occurrences* of ownership vector $\mathbf{p}$ in a matrix. The *interference cost* of adding one row $\mathbf{p}$ to the allocation can be computed *statically*, and it is $c_{\mathbf{p}} = \sum_{(i,j) \in \mathbf{C} \times \mathbf{C}} \alpha_{i,j} \cdot \mathbf{p}[i] \cdot \mathbf{p}[j]$. Once costs are computed, the model can be rewritten as follows:

$$
\begin{array}{llll}
\min \sum_{\mathbf{p} \in \mathbf{P}} c_{\mathbf{p}} \cdot x_{\mathbf{p}} & & & \\
s.t. \quad \sum_{\mathbf{p} \in \mathbf{P}} \mathbf{p}[i] \cdot x_{\mathbf{p}} = A_i & i \in \mathbf{C} & (i) & \\
\qquad \sum_{\mathbf{p} \in \mathbf{P}} x_{\mathbf{p}} \leq M & & (ii) & (2) \\
\qquad x_{\mathbf{p}} \in \mathbb{N} & \mathbf{p} \in \mathbf{P} & (iii) &
\end{array}
$$

Now, the objective function is still the same as the previous problem's. Constraint *(i)* states that the number of RBs given to a node must match its requirements, and constraint *(ii)* states that the number of allocated RBs must not exceed the available ones. The *only* variables are integers $x_{\mathbf{p}}$, which are $O(2^C)$. This model is solvable at optimality by a general-purpose solver (such as CPLEX, [9]) in split-second times for medium-sized clusters (e.g., up to 10 cells). Moreover, ad hoc solution *algorithms* can be devised for it to allow optimal solutions in split-second times for larger scales (i.e., a few tens of cells), always assuming that there *is* a performance return for larger-scale CS. Assessing this is the subject of ongoing work at the time of writing. Once (2) is solved, the AMs can be found by placing $x_{\mathbf{p}}$ instances of each row $\mathbf{p}$ in *any* order.

III. Performance Evaluation

A first evaluation has been carried out by simulation, using SimuLTE [10]. SimuLTE is a system-level simulator for LTE/LTE-Advanced networks, based on OMNeT++ [11] and INET frameworks [12]. The latter provide a simulation environment that comprises a considerable number of models for standard Internet protocols, entities and mobility models, for both wired and wireless networks.

With reference to Fig. 3, the main components of SimuLTE are the UE and the eNB. These include higher-level protocols (i.e. TCP/UDP, IP) taken from INET, as well as an LTE Network Interface Card (NIC) module. The latter implements the functionalities of all the four layers of the LTE/LTE-Advanced stack, in both the uplink (UL) and downlink (DL) directions. At the eNB side, the MAC layer also takes care of scheduling operations (SimuLTE comes with well-known algorithms like MaxC/I, Proportional Fairness and Round Robin). The underlying PHY module implements channel feedback computation/reporting and data transmission/reception. The radio channel is simulated through an extendible interface that allows one to define the most appropriate channel model. SimuLTE supports the simulation of heterogeneous eNBs (macro, micro, pico etc.), using omnidirectional and/or anisotropic antennas. Inter-eNB communications are made possible via the X2 interface [14]. Moreover, SimuLTE includes advanced functionalities like handover, device-to-device communications and inter-cell interference coordination algorithms.

We simulate the scenario of Fig. 4. We consider three macro eNBs, located on every second vertex of a hexagon. Their inter-site distance is 500m. For each macro eNB, two micro eNBs are located with relative angle of +30° and -30°, at a distance of 200m from the macro eNB. Micro eNBs are omnidirectional at a power of 26 dBm, whereas macro eNBs transmit at 40 dBm with an anisotropic pattern, whose attenuation is $A(\theta) = \min\{12 \cdot (\theta/70^{\circ}), 25\}$, where θ is the relative angle between the eNB and the receiver. Both macro and micro eNBs are connected to an entity called *coordinator*, which implements both the ML and GS of the framework presented in the previous sections. The connection is assumed to be ideal, i.e. infinite bandwidth and null latency. The GS runs algorithm (2) every $T = 100ms$, using the CPLEX solver. SRs are averaged over the last period and lower bounded to $b_{\min} = 5$ to ensure responsiveness in low-load conditions. For each couple of eNB (*i,j*), the interference coefficient $\alpha_{i,j}$ employed by the GS is computed offline as follows (the computation differs based on whether *i* is a macro or a micro). For a macro, we placed three UEs around the eNB *i* at a distance of 50m, with angles of -30°, 0° and +30° w.r.t. the antenna direction. For a micro, we placed four UEs at a distance of 20m, with relative angle of 90°

between them. This way, we took into account different transmission power and radiation patterns for macro and micro eNBs. Then, for each UE, we computed the SINR perceived by eNB *j* using the same channel model as in the simulations. $\alpha_{i,j}$ is the average of the three measured SINRs. We randomly deploy an increasing number of (static) UEs over the hexagonal area. They are associated to the eNB that provides the highest Signal-to-Interference-and-Noise Ratio (SINR). We simulate downlink traffic only, where an increasing number of UEs receive a Constant Bit Rate (CBR) traffic at 160 kbps. Simulation parameters are summarized in Table 1.

We first compare the CS algorithm of Section II.b against a baseline of uncoordinated eNBs. In both cases, the eNBs allocate resources according to the well-known MaxC/I policy, starting the allocation from a random position in the subframe. We assume that UEs reports per-RB CQIs and that the eNB selects the CQI to be used for a given UE as follows. First, it discards all the CQIs that are below the median value. Then, the CQI is selected as the minimum among the remaining ones. Obviously, RBs corresponding to discarded CQIs are not considered during the allocation of that UE. Fig. 5 shows the average CQI employed by the eNBs to serve their UEs. We observe that eNBs can exploit higher CQIs, since CS aims at avoiding overlaps among interfering cells, if possible, hence reducing interference. CQIs start to decrease at high loads (300 UEs), when cells request too many RBs. In that case, the coordinator cannot avoid overlap between cells, although the configuration with the minimum interference cost is applied. Higher modulation schemes allow the eNBs to achieve higher throughput or to consume fewer RBs. Fig. 6 reports the throughput measured at each eNB. With CS, eNBs are able to carry their entire offered load, hence the throughput increases with the UE density. On the other hand, in the uncoordinated scenario the throughput is impaired even at low load (i.e., 100 UEs) and caps around 5 Mbps. Fig. 7 shows the total RBs allocated by the eNBs on each TTI. The benefits of CS are evident: the number of allocated RBs is just over 60 RBs with 300 UEs, whereas uncoordinated allocation is already four times as much with 100 UEs, and requires up to 400 RBs with higher UE density.

The average time required to find the optimal solution to the CS problem by the CPLEX solver is reported in Fig. 8. These times are measured on an Intel(R) Core(TM) i7 CPU at 3.60 GHz, with 16 GB of RAM and a Linux Kubuntu 12.04 operating system. The solving time stays at about 5-6 ms to coordinate nine cells when the load is moderate. With 300 UEs, the time is higher since the SRs are such that overlapping cannot be avoided, hence the solver needs more time to find the least-interference overlap of AMs. Anyway, these times are

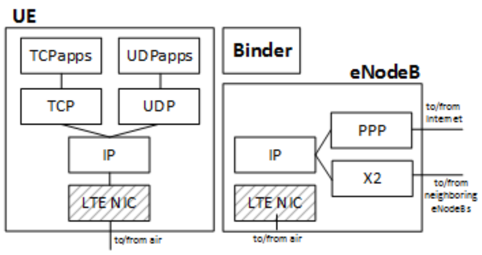

Fig. 3. High-level view of SimuLTE modeling

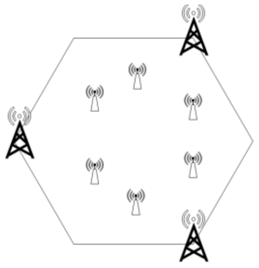

Fig. 4. Simulation scenario

Table 1 - Main simulation parameters

Parameter	***Value***
Carrier frequency	2 GHz
Bandwidth	10 MHz (50 RBs)
Path loss model	ITU Urban Macro [13]
Fading model	Jakes
eNB Tx Power	40 dB (macro), 26 dB (micro)
Antenna gain	18 dB (eNB), 0 dB (UE)
Noise figure	5 dB
Cable loss	2 dB
UE mobility model	Stationary
UE traffic model	CBR, 160 Kbps

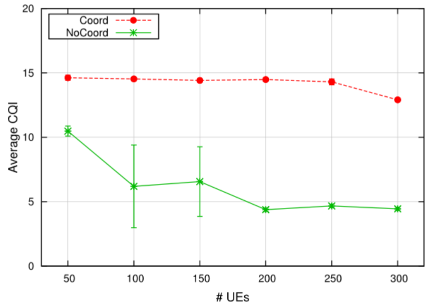

Fig. 5. Average downlink CQI

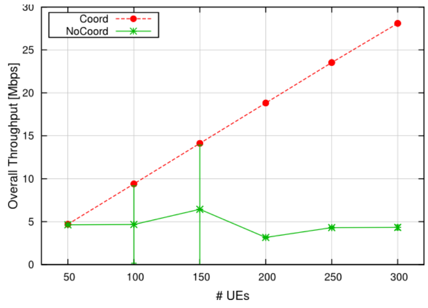

Fig. 6. Overall cell throughput

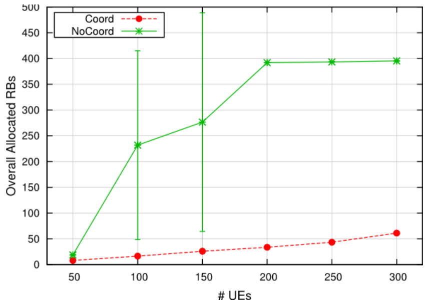

Fig. 7. RBs allocated on average by eNBs

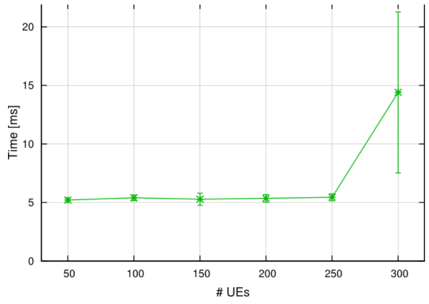

Fig. 8. Average solving time for the CS algorithm

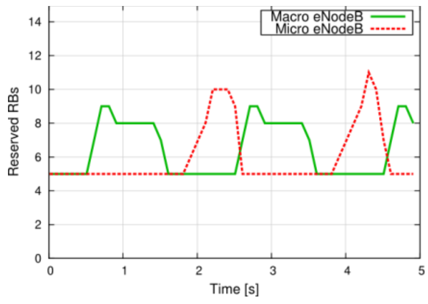

Fig. 9. Evolution of the RBs reserved by the GS

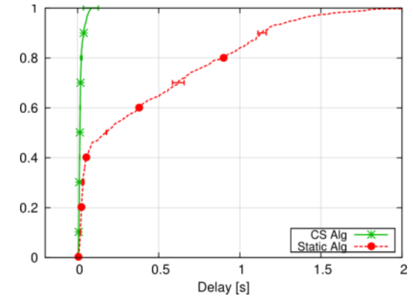

Fig. 10. CDF of application-layer delay

considerably smaller than the CS period, i.e. $T = 100ms$.

In order to show the benefits of *dynamic* CS, we now compare it against a *static* allocation scheme, where the available bandwidth is equally shared among the eNBs, independently of the cells' load. This approach allows the eNBs to exploit mutually exclusive RBs, i.e. with no interference. In the scenario of Fig. 4, we consider 300 UEs that are activated intermittently, with a duty cycle of 50% and period $T = 2s$. In particular, UEs served by macro and micro eNBs are activated alternatively. Fig. 9 reports the number of RBs reserved by dynamic CS to one macro eNB and one of its micros during five seconds of simulation. Allocations at other eNBs are omitted for ease of reading. We observe that during eNBs' inactive periods (i.e., $A_i = 0$) the GS reserves $b_{\min} = 5$ RBs. The GS starts to reserve more RBs during active periods. This means that our algorithm follows the cell's load, reserving the appropriate bandwidth to each eNB. On the other hand, static allocation reserves the same amount of RBs to all eNBs, ignoring their traffic load. This results in an increase of latencies for the application running at the UEs, as shown by the cumulative distribution function (CDF) of Fig. 10. In fact, when eNBs have not enough bandwidth to accommodate their UEs' traffic, the latter has to be buffered for longer time at the eNBs.

IV. Implementation of the V-RAN Prototype

The architecture presented in Section II has been implemented in the V-RAN prototype outlined in Fig. 11, which consists in three nodes implemented on general-purpose machines, each one hosting one vBBU realized with a customized version of the open source OpenAirInterface (OAI) framework [15], enhanced with our modifications [16]-[17]. Each vBBU is connected to a RRH, implemented using Ettus USRP B210 boards. The UEs are commercial Huawei E392u-12 dongles. Another machine hosts the ML, the GS, the GPM and the LPMs for the three nodes. Currently, the dynamic functionalities of the GPM and the LPMs are not used, and these components only manage the boot phase of the testbed. The GS can apply both a *static* CS, which shares the available RBs equally among the nodes in the cluster, and the dynamic CS of Section II. The static CS is load-unaware, hence does not use SRs and sends AMs that implement a crude mutual exclusion approach.

In the prototype, we implemented the different phases of the GS – the part that gets SRs, the part that sends AMs and the part that runs the algorithm – with different threads that run periodically at configurable periods. All the threads communicate through shared memory. The above architecture is shown in Fig. 12. This architecture allows one to plug in other algorithms with minor, localized modifications. The nodes send

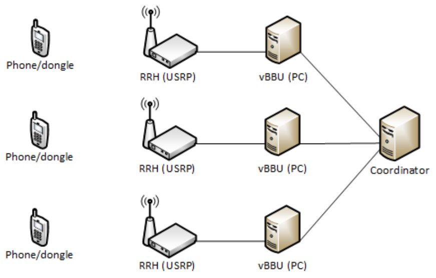

Fig. 11. Prototype overview

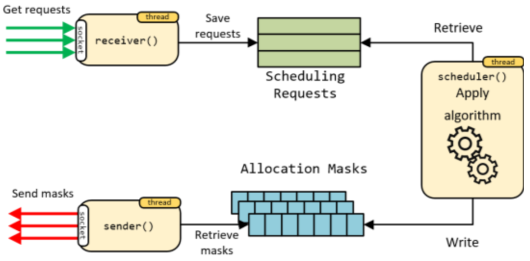

Fig. 12. Architecture of the GS

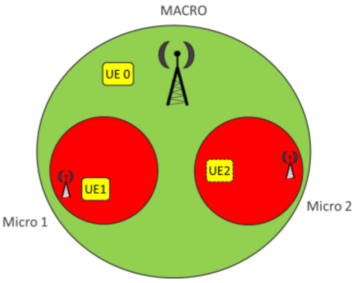

Fig. 13. Scenario reproduced by the prototype

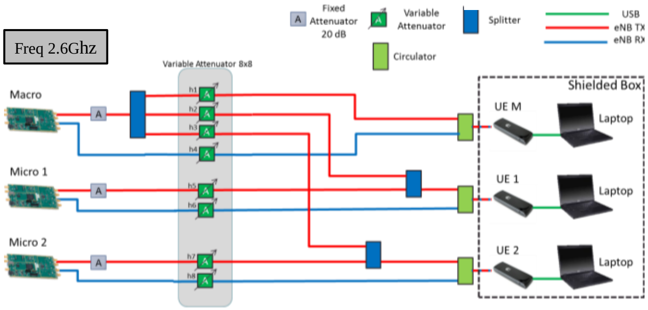

Fig. 14. Implementation of the prototype

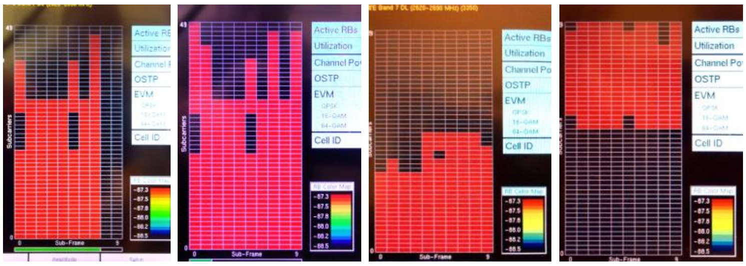

Fig. 15. DL allocation on two eNBs. Left: uncoordinated; right: using our CS

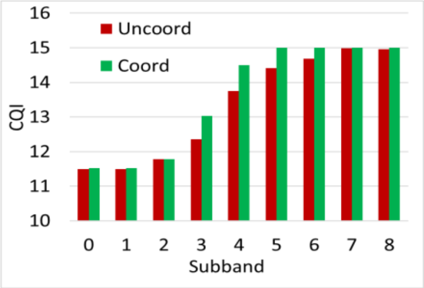

Fig. 16. Average narrowband CQIs reported by UEs connected to micro cells, for uncoordinated and coordinated scenarios

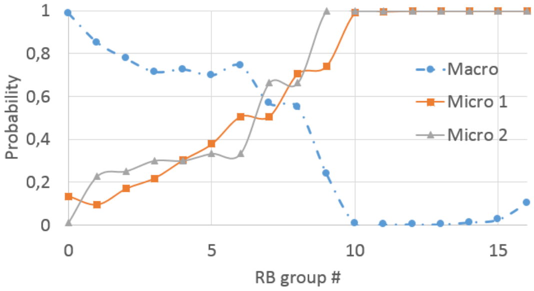

Fig. 17. Probability that the AM of the nodes includes the RB groups when coordinated scheduling is enforced

SRs and receive AMs via UDP sockets (they need high communication rates and packet loss is not an issue), while the GS and ML establish a TCP connection (SR polling is less frequent but needs to be reliable). The scenario is shown in Fig. 13.

Within the trials, to circumvent instability and synchronization problems between UEs and the eNB, the air interface has been channeled in a controlled environment by using wired connections. Variable attenuators, splitters and circulators were used to emulate inter-cell interference and reproduce a scenario where one of the nodes represents a Macro cell and the other two represent two Micro cells (Fig. 14). The Macro UE is used to generate DL interference for the two micros. UE1 is able to hear Macro and Micro 1 signal, but the attenuation on the Macro allows the UE 1 to attach to the Micro 1, and this connection is interfered by the Macro. The same behavior is used for UE 2, which is attached to the Micro 2, but the connection is interfered by the Macro. Micros do not interfere with each other.

We used a spectrum analyzer and XCAL (a commercially available benchmarking software running on the UEs), to analyzed inter-cell interference and its effect on CQIs. A preliminary test with two OAI+USRP nodes and two UEs allowed us to check the interworking among GS and vBBUs. Fig. 15 shows that, without CS, the two eNBs allocate RBs in the whole spectrum. When CS is enabled, instead each eNB uses only the portion of bandwidth allowed by the AM. Using the full prototype, we evaluated how dynamic CS affects CQIs reported by UEs when all nodes transmit DL traffic. Both the macro and the micros transmit DL traffic at 10 Mbps. The OAI MAC scheduler allocates *RB groups*, consisting of three consecutive RBs, starting from the lowest-index ones: when using CS, the set of eligible RB groups is restricted according to the AM of the node. Our UEs in the prototype report *nine* narrowband CQIs, each one roughly corresponding to 1/9th of the available RBs. The test is done with a system bandwidth of 10Mhz, hence $M = 50$.

With CS the algorithm allocates to the macro (roughly) the first half the spectrum, and overlaps the two micros on the second half. Fig. 16 shows the average CQIs reported by UEs attached to micro cells (UE 1 and UE 2), with and without CS. Those on the leftmost subbands are smaller, reflecting the fact that the macro is using those RBs for its own transmissions (both with and without CS). However, in the CS scenario, the macro uses fewer RBs (notably, none of those in subbands 4-8), hence the corresponding CQIs reported by micro users on the second half of the spectrum are maximum, i.e. 15. Fig. 17 shows the probability that the AM includes the RB groups for the macro and the micros. With CS, the micro AMs exploit the higher CQIs. Without CS (i.e., when every node allocates from the first RBs onward), the micros allocate RBs to the user in the subbands affected by macro interference (lower CQIs).

V. Conclusions

In this paper we described the implementation of a V-RAN prototype and in particular the software framework that achieves dynamic Coordinated Scheduling among a relatively large number of nodes, working at subsecond timescales and without requiring any modification to the 3GPP standard. We show that this is made possible by a combination of a well-engineered partitioning of functionalities and a clever modeling of the coordination problem as an optimization problem. Our results, in both a simulator and a prototype environment, show that dynamic CS is effective in increasing the efficiency of the network, using fewer RBs to serve the same traffic.

Since, thanks to CS, nodes are less loaded, then there are more opportunities to switch some of them off to save power. Our future work will include devising algorithms that compute the minimum number of nodes to be kept on to carry a given traffic demand, *assuming coordination is in place*.

Acknowledgment

The subject matter of this paper includes description of results of a joint research project carried out by Telecom Italia and the University of Pisa. Telecom Italia reserves all proprietary rights in any process, procedure, algorithm, article of manufacture, or other result of said project herein described. This work was partially supported by the European Commission in the framework of the H2020-ICT-2014-2 project Flex5Gware (grant agreement no.671563).

References

[1] Flex5Gware website: http://www.flex5gware.eu (accessed Dec. 2016)

[2] L. Fang, X. Zhang, "Optimal Fractional Frequency Reuse in OFDMA Based Wireless Networks", Proc. WiCOM '08, pp.1-4, 12-14 Oct. 2008.

[3] S.H. Ali, V.C.M. Leung, "Dynamic frequency allocation in fractional frequency reused OFDMA networks", IEEE Transactions on Wireless Communications, vol.8, no.8, pp. 4286-4295, Aug. 2009.

[4] K. Hoon, H. Youngnam, J. Jayong, "Optimal subchannel allocation scheme in multicell OFDMA systems", Proc. of VTC Spring'04 pp.1821-1825 Vol.3, 17-19 May 2004.

[5] G. Li, H. Liu, "Downlink Radio Resource Allocation for Multi-Cell OFDMA System", IEEE Transactions on Wireless Communications, vol.5, no.12, pp.3451-3459, Dec. 2006.

[6] C. Koutsimanis, G. Fodor, "A Dynamic Resource Allocation Scheme for Guaranteed Bit Rate Services in OFDMA Networks", Proc. ICC '08, pp.2524-2530, 19-23 May 2008.

[7] M.Y. Arslan, *et al.* "A Resource Management System for Interference Mitigation in Enterprise OFDMA Femtocells", IEEE/ACM Transactions on Networking, vol.21, no.5, pp.1447-1460, Oct. 2013

[8] G. Nardini, *et. al*, "Practical large-scale coordinated scheduling in LTE-Advanced networks", Wireless Networks, 22:(1), pp. 11-31, 2016

[9] ILOG CPLEX Software, http://www.ilog.com

[10] A. Virdis, *et. al*, "Simulating LTE/LTE-Advanced Networks with SimuLTE", DOI 10.1007/978-3-319-26470-7_5, Advances in Intelligent Systems and Computing, vol.402, pp.83-105, Springer, 15 Jan. 2016

[11] A. Varga, R. Hornig, "An overview of the OMNeT++ simulation environment", in Proc. SIMUTools '08, Marseille, France, Mar. 2008

[12] INET framework for OMNeT++: http://inet.omnetpp.org/

[13] 3GPP TR 36.814 v9.0.0, "Further advancements for E-UTRA physical layer aspects (Release 9) ", Mar. 2010

[14] 3GPP TS 36.420 v13.0.0, "Evolved Universal Terrestrial Radio Access Network (E-UTRAN); X2 general aspects and principles (Release 13)", Dec. 2015

[15] R. Wang, *et al.*, "OpenAirInterface - An effective emulation platform for LTE and LTE-Advanced", Proc. ICUFN 2014, Shanghai, pp. 127–132.

[16] N. Iardella, *et al.*, "Statistically sound experiments with OpenAirInterface Cloud-RAN prototypes", Proc. of CLEEN 2016, Grenoble, FR, May 2016

[17] A. Virdis, *et. al*, "Performance analysis of OpenAirInterface system emulation", Proc. of PMECT 2015, Rome, IT, Aug. 26, 2015