Modeling X2 backhauling for LTE-Advanced and assessing its effect on CoMP Coordinated Scheduling

Giovanni Nardini, Antonio Virdis, Giovanni Stea
Dipartimento di Ingegneria dell'Informazione, University of Pisa
Largo Lucio Lazzarino 1, I-56122, Pisa, Italy
g.nardini@ing.unipi.it, a.virdis@iet.unipi.it, giovanni.stea@unipi.it

***Abstract*—Many LTE-Advanced algorithms and protocols rely on node coordination and cooperation to reduce power consumption, increase spectral efficiency and improve cell-edge performance. Functions such as Coordinated Multi Point, Network Assisted Handover, etc., require a standard connection among nodes to support their operations. The LTE X2 interface meets the above requirements and allows operators to connect nodes for both rel-8 and more advanced (e.g rel-13) functionalities. In this work we describe the modeling of X2 within the SimuLTE system-level simulator. Most research works assume an ideal X2 connection, with null delay and infinite bandwidth. However, the X2 delay and bandwidth do affect the behavior and performance of the aforementioned algorithms. Thus, using CoMP Coordinated Scheduling as a case-study to test X2 functionalities, we show how X2 round-trip delay affects the performance of the CoMP scheduler.**

Keywords—LTE, LTE-Advanced, X2, Coordinated Multi-Point, system-level simulation.

I. INTRODUCTION

The ever-increasing demand for data traffic from mobile smartphone users is one of the main drivers in the evolution of LTE and LTE-Advanced along their path to 5G. Higher bandwidths and lower latencies are not anymore *cell-average* requirements, but must be guaranteed to *all* the users including cell-edge ones and/or those in dense areas, in spite of high interference. These issues have been addressed by LTE through various approaches, e.g. Network-Assisted Handover or interference coordination. In the former the base stations, or evolved NodeB (eNB), assist the user equipment (UE) in the cell selection procedure, favoring the associations to eNBs with lower load and/or with higher expected performance. The latter approach instead aims at reducing the interference among eNB by coordinating their transmissions. This problem has been tackled since rel-8 through the so-called *enhanced Inter Cell Coordination* (eICIC), a technique focused on coordinating resource allocation in the time domain, i.e. selecting who is the "owner" of a given subframe. This approach, by its very nature, can only work at suitably large timescale. A more dynamic approach has been proposed in more recent releases, with the introduction of *Coordinated Multi-point* (CoMP), which instead works by deciding the ownership of *single RB*s (or groups thereof), hence being considerably more flexible. All the above techniques require communication among eNBs, a request that has been met by the LTE standard by defining the X2 communication interface. The latter is a logical interface between eNBs, which aims at connecting nodes from possibly different vendors in a transport-connection-agnostic manner. X2 interfaces can run on pre-existing physical connections and various topologies. Therefore, X2-level communications will be affected by a certain delay and/or bandwidth limitations, which might also arise from contention with non-X2 data sharing the same physical infrastructure. The functions of X2 are supported by the X2 Application Protocol (X2AP), specified in [16], which defines a set of signaling procedures and message formats.

In this paper we describe the integration process of X2 functionalities into SimuLTE [2], an OMNeT++-based system simulator for LTE/LTE-A networks, which is available for download at [17]. More in detail, we explain how to connect the elements for the LTE Radio Access, namely the LTE NIC card, with the X2 protocol layers in order to have eNBs communicate. Then we describe the modeling of X2AP messages, and especially how to define new ones to support new protocols. As a proof-of-concept evaluation, we then use the above to evaluate the performance of two CoMP CS algorithms from the literature [13], in order to assess how X2 delay affects them. Most of the research works on LTE coordination algorithms, in fact, abstract away the underlying X2 connections (see, e.g., [12]), assuming infinite bandwidth and null delays. However, the X2 delay affects the performance of the coordination algorithms, thus the overall system performance. The latter will then depend on the ability of the considered algorithm or protocol to absorb such impairments, which are expected to be non negligible [3]. Some preliminary evaluation [4] investigated the effects of delay on CoMP Joint Processing algorithms, highlighting that the impact of even small delays is non-negligible. To the best of our knowledge, no works on the impact of X2 communication on CoMP Coordinated Scheduling (CoMP CS) are available in the literature.

The rest of the paper is organized as follows: in Section II, we provides some background on LTE-Advanced. In Section III we describe the general architecture of SimuLTE. Section IV explains the integration process of X2 into the simulated system. In Section V we propose a validation of the simulator and a performance evaluation of two CoMP CS algorithms in presence of delay. Section VI concludes the paper.

II. BACKGROUND ON LTE-A

This section introduces background on LTE, focusing on its layering, scheduling functions and on the X2 communications, all of which will be referred to in the rest of the paper.

The LTE protocol stack is located at layer 2 of the OSI stack, and it includes several sublayers with different functions. Focusing on the downlink direction (i.e., from the eNB to the UE), and with reference to Fig. 1(a), IP packets arrive at the Packet Data Convergence Protocol (PDCP), where they are cyphered and numbered. Below that, there is the Radio Link Control (RLC) layer, where RLC SDUs are buffered. The MAC layer sits below, and includes the scheduling functions. The MAC-layer scheduler runs on each Time Transmission Interval (TTI, 1 ms), and composes a vector of Resource Blocks (RBs) destined to the various UEs. In order to do so, it dequeues from the RLC buffer of a UE as many bytes as can fit into the RBs that it allocates to that UE. A MAC Transport Block (TB) destined to a UE, which occupies some RBs, is transmitted using a given modulation and coding scheme. The latter is selected based on the Channel Quality Indicator reported (either periodically or on demand) by the UE, and determines the number of bytes per RB that can be transmitted. MAC-layer transmissions are protected by a Hybrid ARQ (H-ARQ) scheme. The UE sends an ACK/NACK in 4TTIs, and the eNB may retransmit the NACK-ed TB at any future TTI, for a configurable maximum number of times.

Uplink (UL) transmission follows the same paradigm, *mutatis mutandis*, with some key differences: first of all, the eNB needs to issue mutually exclusive *transmission grants* to the UEs, which then compose the UL subframe themselves by transmitting their traffic in the allocated RBs. Grants should only be given to backlogged UEs, hence UEs transmit a Backlog Status Report (BSR) as well, either alone or trailing data, to signal their backlog status to the eNB. When an empty UE becomes backlogged, it uses a Random Access procedure (RAC) to inform the eNB that it needs resources. RAC requests may collide, and are reiterated after a backoff period if unanswered. The eNB answers a RAC request by scheduling resources (usually one RB, enough for a BSR) to the requesting UE. UL transmissions are protected by a H-ARQ as well, but the standard mandates that a failed transmission must be repeated after eight TTI.

The X2 interface [15] provides both control and data plane for the communication among different eNBs. The protocol stack for the control plane is shown in Fig. 1(b). Signaling information are generated by the X2 Application Protocol (X2AP) [16], which defines a large set of procedures and messages for supporting inter-eNB operations, e.g. load management and inter-cell interference coordination. Layer-4 functionalities are provided by the Stream Control Transmission Protocol (SCTP), which establishes and maintains the association between two peering eNBs. Data plane is used for data PDUs (e.g., during Network-Assisted Handover), which relies on GPRS Tunneling Protocol (GTP) and UDP. In the latest releases of LTE, the X2 interface can also be used for communication between eNBs and a central coordinator, e.g. for CoMP centralized coordination [5].

CoMP Coordinated Scheduling addresses the problem of deciding which eNB in a coordinated set uses which RBs, so that interference is minimized. This is particularly important for cell-edge UEs, that may perceive comparable power from the serving cell and the neighboring one(s). CoMP CS is accomplished either statically or dynamically. Examples of the first class are Partial Frequency Reuse (PFR) [6] and Soft Frequency Reuse (SFR) [7]. The idea behind PFR is to partition the bandwidth so that only a limited amount of RBs can be used by all cells, while others are used with higher reuse factor. Cell-edge UEs can take advantage of lower interference in these sub-bands. In the SFR scheme, a cell can allocate the entire subframe, but different power levels are employed in cell-center and cell-edge RBs. Dynamic schemes, such as [8]-[13], can achieve better performance by leveraging time-averaged or even instant knowledge of the amount of traffic and UE location in a cluster of neighboring cells. Dynamic CoMP CS schemes, however, require communication among either the eNBs themselves, in a peer-to-peer fashion, or between eNBs and a central coordinator entity. Such communication could include UE load and positioning data, and requests/grants to use a certain pool of RBs, and it is expected to run on X2.

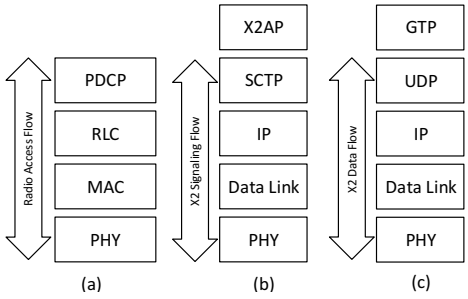

Fig. 1. Description of the Radio Access and X2 flows

III. SimuLTE Architecture

In this section we describe SimuLTE, with particular emphasis on those aspects that are related to the problem at hand. SimuLTE is a system-level simulator based on the OMNeT++ [1] framework. OMNeT++ revolves around the concept of *module*, which is the basic modeling unit. Modules communicate among themselves via message exchanges, and they can be organized in a hierarchy of *compound* modules. Modules have both a *structure*, defined via `.ned` files, and a *behavior*, implemented via `C++` classes. This allows one to change either of the two without affecting the other.

The core module of SimuLTE is the LTE Network Interface Card (NIC). Both UEs and eNBs incorporate one, as well as other modules from the INET framework. The latter is a library of OMNeT++-based modules developed by the community, that model standard Internet protocols, functions and entities. INET also models entities outside the LTE scope, e.g. application servers, that are used as traffic generators/receivers and communicate with the applications within the UEs. Fig. 2 provides a high-level view of the nodes. Defining NIC allows one to model nodes with multiple interfaces (e.g. LTE and Wi-FI), in full conformance to the modular paradigm of the OMNeT++ framework.

The NICs in the UE and eNB are organized in layers (PDCP, RLC, MAC and PHY), with a one-to-one correspondence with the LTE protocol stack. Leveraging inheritance of both the structure and behavior of modules, we model the *common* functionalities of each node in a base class, and then add *node-specific* functionalities when required. For

instance, the *MacUe* and *MacEnb* classes both extend the *MacBase* class. The eNB class includes resource scheduling as a node-specific function. The eNB includes an IP layer and a PPP interface to connect it to the Evolved Packet Core. The UE, instead, includes also transport layers and applications.

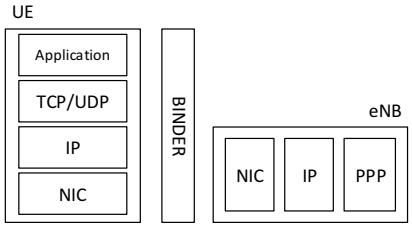

Fig. 2. High-level representation of the system.

Instead of modeling the airframe with all its symbols, SimuLTE separates *data transmission* from *resource accounting*. Resource accounting (i.e., keeping track of which RBs are used by whom) is done by a central module, called the Binder. The latter acts as an oracle, since it has the full visibility of all the nodes in the system, and can be queried by them to obtain shared information. This is useful for several purposes, e.g., it provides users with the ability to design and run ideal algorithms, leveraging full knowledge of the ongoing transmissions, and use them as optimal baselines for limited-scope, distributed ones. However, unless otherwise instructed, each LTE node only uses information that is supposed to be available to it. Data transmissions are instead modeled via message exchanges between modules. The Binder associates each message to the amount of RBs carrying it, based on the length of the MAC PDU and on the modulation and coding scheme employed by the transmitter. Control channels, such as the Physical Downlink Control Channel (PDCCH), that is used to carry scheduling assignments, are not directly modeled, rather they are abstracted using separated messages.

Interference management is guaranteed by endowing each NIC with a ChannelModel. The latter interacts with the PHY layer and models the status of the air channel as perceived by its NIC. The ChannelModel computes the SINR of signals received by the node, which in turn is used by the PHY layer to compute the CQIs and evaluate transmission errors. To compute the SINR, the ChannelModel queries the Binder to know who else is transmitting on the RBs occupied by the message directed to its NIC. SimuLTE defines the ChannelModel as an interface, i.e. a C++ abstract class with pure virtual functions only, and also provides an implementation of a realistic model, which accounts for path loss, fading and shadowing. If needed, such interface can be easily extended by implementing the two functions `getSINR()` and `error()`, used for the above functions.

IV. X2 Modeling and Implementation

This section details how we enhanced SimuLTE to support X2 communications. The original eNB architecture in SimuLTE provides a path between the LTE NIC and the Evolved Packet Core (EPC) network, where the two interfaces are interconnected through the IP layer. This allows one to simulate an LTE cell connected to an EPC, but not to have eNBs communicate among them.

Since X2 communications run on SCTP, which is a transport (i.e., layer-4) protocol, we need to model the X2 protocol as an *application*. Thus, we include all layer-4 protocols into the eNB, and the INET application classes that go with it. Moreover, we add PPP NICs to the eNB to model the layer 2 of X2 connections.

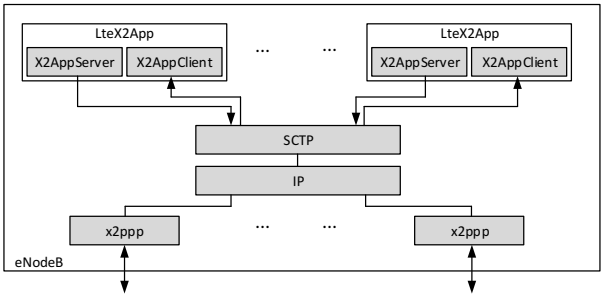

Fig. 3. Layering of the X2 interface

With reference to Fig. 3, we endow the eNB with a vector of X2 application modules, called *LteX2App*, running on top of the SCTP layer. Each LteX2App peers with *one* other eNB and is composed by *one pair* of modules called *X2AppServer* and *X2AppClient*, for sending and receiving messages, respectively. Such modules extend *SCTPServer* and *SCTPClient* modules provided by INET and take care of establishing and maintaining the connection between the two eNBs. The X2AppServer module receives messages from the LTE stack and commands the SCTP protocol to transmit them. On the other hand, the X2AppClient module receives messages from the SCTP and delivers them to the LTE stack. Transmitted messages traverse the existing IP layer and reach the network interface, called *x2ppp*. The latter is a vector of NIC cards, whose size depends on the network topology used to interconnect the eNBs. This allows the eNBs to be linked together using an arbitrary network topology. For example, eNBs can be connected using a *full mesh* or a *star* topology, as shown in Fig. 4. In the former case, the eNB has $N-1$ interfaces. In the latter case, only one interface connected to a central router is required. Alternative topologies are possible, by simply composing links and network elements provided by the INET framework. Depending on the chosen topology, X2AppClient modules must be configured with the destination IP address of the interface of the eNB where it has to connect. This is accomplished by setting the *connectAddress* parameter in the INI configuration file. Fig. 5 shows a snippet of the configuration file for the full-mesh topology in a scenario with three eNBs. Routing is carried out accordingly by INET functionalities.

Fig. 6 shows the interaction between the LTE protocol stack and the X2 interface described above, which is handled by the *LteX2Manager* module. The latter has, on one hand, one connection gate per LteX2App module, hence one gate per destination eNB. On the other hand, it has one connection gate per *X2User* module. An X2User module implements an entity (e.g., an algorithm or part of it) that exploits X2 communication to accomplish its task. It possibly interacts with the layers of the LTE protocol stack through direct method calls (dashed lines in Fig. 6) and sends information to the LteX2Manager, together with the list of destination eNBs. The

LteX2Manager creates a copy of the information for each destination and forwards it to the correct LteX2App. At the receiving side, the LteX2Manager takes data from the LteX2App modules and passes them to one X2User module. In other words, the LteX2Manager performs the (de)multiplexing of the information between X2User and LteX2App modules. In particular, messages coming from the X2 are multiplexed based on the message type. To do this, it is necessary that each X2User module informs the LteX2Manager about the type of messages it means to receive, thus a registration phase is carried out at the beginning of the simulation between them.

X2User and LteX2Manager exchange *LteX2Messages*, whose format is shown in Fig. 7. An LteX2Message contains the message type, length and a list of *X2InformationElements*, which are the basic units of information exchanged among the eNBs. The X2User communicates the list of destination eNBs through the *X2ControlInfo* attached to the LteX2Message. Each X2User module only needs to extend the LteX2Message and X2InformationElement classes so as to define its own message format.

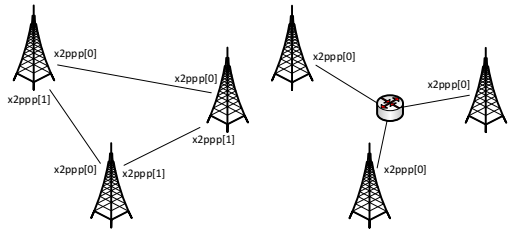

Fig. 4. Full mesh (left) and star (right) topologies

```
*.eNodeB1.x2App[0].client.connectAddress= "eNodeB2%x2ppp0"
*.eNodeB1.x2App[1].client.connectAddress= "eNodeB3%x2ppp0"
*.eNodeB2.x2App[0].client.connectAddress= "eNodeB1%x2ppp0"
*.eNodeB2.x2App[1].client.connectAddress= "eNodeB3%x2ppp1"
*.eNodeB3.x2App[0].client.connectAddress= "eNodeB1%x2ppp1"
*.eNodeB3.x2App[1].client.connectAddress= "eNodeB2%x2ppp1"
```

Fig. 5. Configuration for the full mesh topology

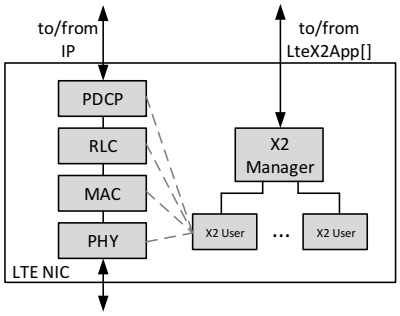

Fig. 6. Interface between LTE NIC and X2 in the eNB

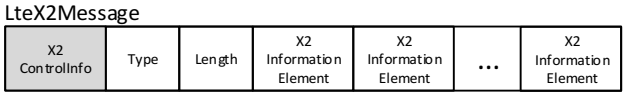

Fig. 7. Message format

A. A CoMP CS application running on X2

We now describe modeling and implementation of a CoMP Coordinated Scheduling algorithm running on X2.

Our CoMP CS model follows the *master-slave* paradigm, where a master node (i.e., either a central coordinator or a designated eNB) takes coordination decisions, relying on information sent from slave nodes (i.e., eNBs participating in the coordination algorithm) through X2. In particular, on each TTI, slave nodes fill in requests in terms of RBs required to support its downlink traffic load (obtaining relevant information from the MAC layer) and send them to the master node. The latter gathers all the requests from the cluster of slaves, partitions the bandwidth according to a configurable policy, then sends back the map of available RBs. Each eNB schedules transmissions in the allowed RBs only.

To do this, each eNB is endowed with a module, namely the *LteCompManager*, which performs CoMP-related operations (as either master or slave) and interacts with the X2 interface described above in a seamless way. The behavior of the LteCompManager can be easily redefined to include the desired coordination policy. The LteCompManager extends the X2User module and exchanges *X2CompMessages* with the LteX2Manager, as shown in Fig. 8. Those messages are obtained as extension of the LteX2Message class and may contain two types of X2InformationElements, the *X2CompRequestIE* and the *X2CompReplyIE*. The former is the request sent by slave nodes, whereas the latter is sent by the master node after the partitioning.

As test cases, we implemented two coordination policies, namely a dynamic reuse-n algorithm and the CoMP CS algorithm in [13]. In the dynamic reuse-n, each eNB gets exclusive use of a portion of the available bandwidth, which can be exploited to schedule UEs without interference. On each TTI, slave nodes compute the number of RBs required to transmit its backlog's worth of traffic. The X2CompRequestIE contains only one integer field. The coordination policy at the master node partitions the bandwidth proportionally to the slaves' requests. The X2CompReplyIE contains a bitmap, where the *i*-th bit is set if the eNB can use RB *i*. In the algorithm described in [13], the available bandwidth is partitioned into subbands, called *interference logical subbands* (ILSs), where only *subsets* of the eNBs can be active simultaneously. In a setting where three eNBs are coordinated, each eNB has a *shared* ILS, where all three eNBs transmit together, two *single-muting* ILSs, where it transmits together with another, and one *double-muting* ILS, where it transmits alone. On each TTI, slave nodes compute the number of required RBs for each ILS, based on both the channel conditions and the UE buffer status. In this case, the X2CompRequestIE contains one integer value per ILS. The master node then computes the size and offset of each ILS and sends them back to the slaves into an X2CompReplyIE message.

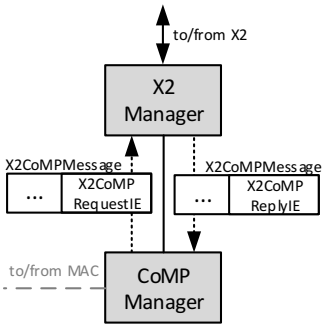

Fig 8. Message exchange between the CoMP Manager and the X2 Manager

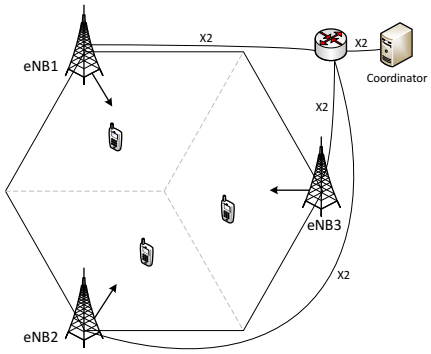

Fig. 9. Evaluation scenario

TABLE I. MAIN SIMULATION PARAMETERS

Parameter	*Value*
Carrier frequency	2 GHz
Bandwidth	5 MHz (25 RBs)
Path loss model	ITU Urban Macro [14]
Fading model	Jakes
eNB Tx Power	46 dB
Noise figure	5 dB
Cable loss	2 dB
Simulation time	50 s
UE mobility model	Stationary

V. PERFORMANCE EVALUATION

This section evaluates the effects of non-ideal X2 backhauling on the performance of interference coordination algorithms. The simulation scenario is reported in Fig. 9. We consider a hexagonal area, with three eNBs located at three vertices at a distance of 500 m from each other. eNBs radiate towards the center of the hexagon with a power of 46 dB and the attenuation pattern is $A(\theta) = \min\left\{12 \cdot (\theta / 70°)^2, 25\right\}$, θ being the relative angle between the eNB and the receiver. 5 MHz bandwidth is employed, resulting in 25 RBs per TTI. Channel is affected by shadowing and fading effects, modeled as in [14]. UEs are static, randomly deployed over the area. We consider downlink traffic only, originating from a remote server and forwarded to the serving eNB, which in turn schedules transmission to the destination. eNBs communicate with a *coordinator* through a router connected to X2 interfaces of each eNB. Simulation parameters are reported in Table I.

First, we assess the effect of traffic load variations when the dynamic *reuse-n* scheme discussed in Section IV is employed. Then, we consider the CoMP CS algorithm proposed in [13]. In this case, variations of interference conditions of UEs (e.g. due to mobility) may change the ILSs width, even if the traffic load is constant. Both algorithms partition the bandwidth at the coordinator on each TTI, relying on information coming from the eNBs via the X2 interface.

A. Variations of traffic load

We consider the scenario of Fig. 9, with five UEs per eNB. UEs served by eNB2 and eNB3 receive CBR traffic at 800 Kbps, whereas UEs served by eNB1 receive a burst of 1000 B every 500 ms. This results in sharp load peaks followed by relatively long inactivity periods. The reuse-3 algorithm is employed. Since the traffic load at eNB2 and eNB3 is constant, they request the same number of RBs to the coordinator during the entire simulation, except for variations due to fading. On the other hand, eNB1 requests no RBs during inactive periods and a burst of RBs at load peaks. Fig. 10 shows the temporal evolution of the number of requested RBs by eNB1 (solid line) and the corresponding number of RBs reserved by the coordinator (dashed line), for both 0ms (left) and 5ms (right) round-trip latency. Clearly, with an ideal X2 connection, the coordinator reacts immediately to the eNB1's request, whereas with non-ideal X2 the adaptation is delayed. This behavior affects the average application-level delay of UEs served by eNB1, as shown in Fig. 11. Since eNB1 has no reserved RBs, it has to wait for its request to reach the coordinator and for the reply to come back, before serving its UEs. Thus, the application-level delay increases with the X2 latency.

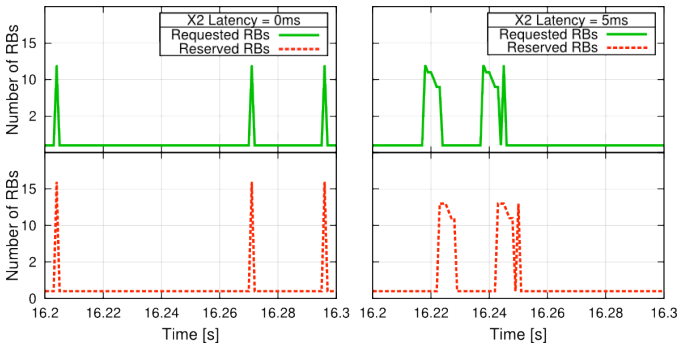

Fig. 10. Evolution of requested and reserved RBs, with ideal and non-ideal X2

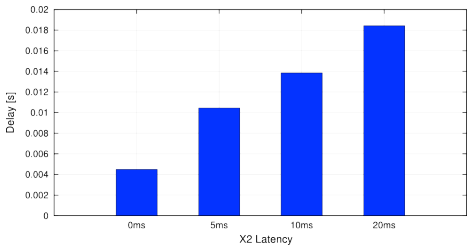

Fig. 11. Average application-level delay of UEs served by eNB1

B. Variations of interference conditions

We now evaluate the CoMP-CS algorithm in [13], in a scenario with 50 UEs per eNB. We deploy 25 UEs close to the eNB and 25 UEs in the center of the hexagon. This way, the former have good channel quality even if scheduled in the shared ILS, whereas the latter should be scheduled in the double-muting ILS to avoid high interference. To exacerbate the variation of interference conditions, we send traffic alternatively to either group of UEs, so that the eNB has to modify its requests from shared to mutually exclusive RBs and vice versa periodically, with a period of 50 ms. During activity

periods, which last for 20 ms, UEs receive CBR traffic at 80 Kbps. Fig. 12 shows that the average number of RBs allocated by one eNB increases with the X2 latency. This is because eNBs receive the updated RB masks later than expected and cannot schedule a UE in the most suitable ILS. For example, cell-edge UEs (i.e., those in the center of the hexagon) require more RBs if served in the shared ILS, given the high interference perceived from neighboring eNBs. In other words, latency on interference coordination causes an artificial increase of the consumed resources, even if the traffic load stays the same. This also affects the average application-level delay, as shown in Fig. 13, since an eNB may not have enough free RBs at a given TTI to serve some UEs, whose transmission must therefore be delayed. Considering cell-center and cell-edge UEs separately, Fig. 14 shows the respective cumulative distribution functions (CDFs) of the application-level delay. Although the delay increases with the X2 latency in both cases, cell-edge UEs suffer higher performance degradation, since they suffer strong interference if not scheduled in the double-muting ILS.

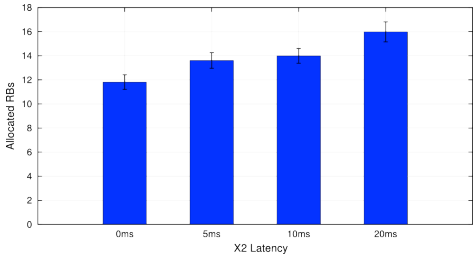

Fig. 12. Average allocated RBs

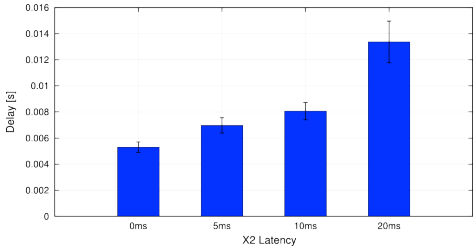

Fig. 13. Average application-level delay

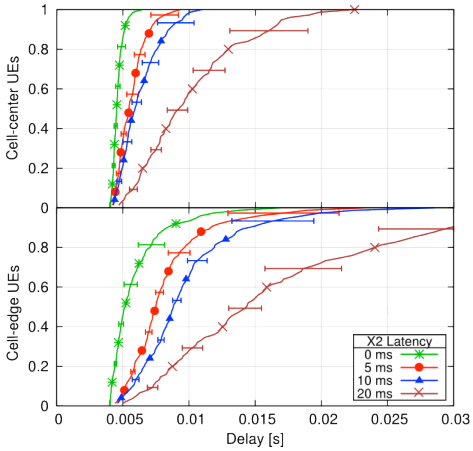

Fig. 14. CDF of the application-level delay for cell-center and cell-edge UEs

VI. Conclusions

In this work we have described the modeling of the X2 interface in the SimuLTE system-level simulator. Modeling the X2 prompted a general refactoring of the eNB structure, to allow transport protocols and related applications, X2 being in fact one such application running on SCTP. Our modeling of X2 is fairly general, in that it allows potential users to exploit it for any purpose, adding new X2 messages as required. As a test case, we used X2 as an infrastructure on which to run CoMP CS schemes, which requires communication among eNBs. Our modeling allows one to simulate CoMP CS algorithms in a more realistic setting, where the X2 limitations are taken into account. As a proof of concept, we have shown how two CoMP CS algorithms taken from the literature perform when subject to increasing X2 delays.

References

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

[2] A. Virdis, G. Stea, G. Nardini, "Simulating LTE/LTE-Advanced Networks with SimuLTE", DOI 10.1007/978-3-319-26470-7_5, in: Advances in Intelligent Systems and Computing, Vol. 402, pp. 83-105, Springer, 15 January 2016

[3] Backhauling X2, Cambridge Broadband Netwroks, White Paper 2011.

[4] S. Brueck, L. Zhao, J. Giese and M. A. Amin, "Centralized scheduling for joint transmission coordinated multi-point in LTE-Advanced," Proc. of the Workshop on Smart Antennas 2010, Bremen, pp. 177-184.

[5] Metsälä, E. M., Salmelin, J. (eds) (2015) Index, in "LTE Backhaul: Planning and Optimization," John Wiley & Sons, Ltd, Chichester, UK. doi: 10.1002/9781118924655.ch1

[6] Sternad, M., Ottosson, T., Ahlen, A., Svensson, A. (2003), Attaining both coverage and high spectral efficiency with adaptive OFDM downlinks, *Proc. of VTC 2003-Fall*, pp.2486-2490 6-9 Oct. 2003.

[7] 3GPP, "Soft Frequency Reuse Scheme for UTRAN LTE," 3rd Generation Partnership Project (3GPP), R1-050507, May 2005.

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

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

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

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

[12] M. Rahman, H. Yanikomeroglu, "Enhancing cell-edge performance: a downlink dynamic interference avoidance scheme with inter-cell coordination," IEEE Trans. on Wireless Communications, vol. 9, no. 4, pp. 1414-1425, April 2010.

[13] G. Nardini, G. Stea, A. Virdis, D. Sabella, M. Caretti, "Practical large-scale coordinated scheduling in LTE-Advanced networks", ResearchGate.net Springer Wireless Networks, DOI: 10.1007/s11276-015-0948-6, Volume 22, Issue 1, pp. 11-31, January 2016

[14] 3GPP TR 36.814 v9.0.0, "Further advancements for E-UTRA physical layer aspects (Release 9)," March 2010.

[15] 3GPP TS 36.420 v13.0.0, "Evolved Universal Terrestrial Radio Access Network (E-UTRAN); X2 general aspects and principles (Release 13)," December 2015.

[16] 3GPP TS 36.423 v13.2.0, "Evolved Universal Terrestrial Radio Access Network (E-UTRAN); X2 Application Protocol (X2AP) (Release 13)," December 2015.

[17] SimuLTE website: http://simulte.com