Project Number:	Project Acronym:	Project Title:
248993	LOLA	Achieving Low-Latency in Wireless Communications

Instrument:	Thematic Priority
STREP	The network of the future

Workpackage:
WP5

Deliverable or Document Title:
Validation Results of WP4 Algorithms on Testbed 3

Keywords :
MESH, LTE, clusterhead, Mobile router, cooperative transmission, Relaying,

Organization name of lead contractor for this deliverable:	Document version:
EURE	V1.0

Partners (organisations) contributing to this document :
EURE, TCS, and LiU

Contractual Delivery Date:	Actual Delivery Date:
2013-03-31	2013-06-3

Start date of project:	Duration:
2010-01-01	39 months

Dissemination level (Project co-funded by the European Commission within the Seventh Framework Programme)		
PU	Public	X
PP	Restricted to other programme participants (including the Commission)	
RE	Restricted to a group defined by the consortium (including the Commission)	
CO	Confidential, only for members of the consortium (including the Commission)	

Revision History

The following table is a record of the main modifications done to the document since its creation.

Revision	**Date**	**Author (Organisation)**	**Description**
V0.0	2013-03-31	EURE	Creation and ToC
V0.7	2013-005-29	TCS / LiU	Contributions from TCS and LiU
V0.8	2013-005-29	EURE	Contributions from EURE
V0.9	2013-005-29	TCS / LiU	Contributions from TCS and LiU
V1.0	2013-06-03	EURE	Review and release

Table of Contents

1. EXECUTIVE SUMMARY

LOLA D5.7 deals with the validations results concerning LOLA testbed 3 for rapidly deployable wireless mesh network for safety applications. The previous network is also called Topology B in LOLA, in order to distinguish it from Topology A dealing with cellular architectures. Testbed 3 is implemented through the OpenAirInterface (OAI), and it implements the most important features of the LOLA mesh network, which is a wireless static mesh network built on top of 3GPP LTE, which we refer as LTE mesh extension.

In this document, the LOLA consortium provides results of the integration and validation of some of WP4 algorithms into the LOLA mesh. The main innovation tested here is the implementation of the Virtual Link (VL or VLink) at the MAC level. The LOLA mesh is a static clusterized wireless network. Cluster Heads (CHs) cannot directly communicate, so routing through other connecting nodes, called bridging Mesh Routers (MRs), is necessary. The idea is to create virtual links connecting directly CHs of adjacent clusters such that routing decisions are taken only at the CH level and not at the MR level. Moreover, in order to reduce the latency of the communications through the MRs, MAC forwarding is implemented at the MRs. When more than one MR is connecting the same two CHs, cooperation among the MRs can be activated in order to increase the robustness of the communication.

The structure of the deliverable is as follows. Section 2 deals with testbed 3 setup. The section about validation scenarios (Sect 2.1) explains the methodological approach in order to prove the interest and validity of the concepts related to the VLink for the static LOLA mesh network. First, a validation scenario at link level was used in order both to implement and to check concepts related to cooperative transmission from multiple bridging Mesh Routers (MRs) combined with Hybrid Automatic Repeat and Request (HARQ), which were also studied in WP4 (see LOLA D4.3 M36 Release). The second step is the so-called in-lab system validation, in which LOLA innovations and MAC and higher level were implemented based on the LTE protocol stack and tested both on the control plane and the data plane. Finally the Radio Frequency (RF) validation scenario aims at validating the integration on the hardware and of some basic functionalities of the PHY, which are necessary for the operation of the network. We stress that this deliverable does not contain results related to the RF scenario, those results will be shown and recorded during the final project demonstration.

Section 2.2 briefly presents the overall LOLA mesh solution (more details are included in LOLA D4.2 and D5.6). Sections 2.3 and 2.4 respectively contain an updated description of the specifications of the LOLA mesh which were presented in LOLA D5.6. In fact, during the software development and integration phase, certain specifications evolved, so we decided to collect the material in this last deliverable for the sake of clarity and completeness. Sections with “update” in the title contain the modified specifications. Sections without main updates were shortly reported here, with a reference to previous deliverables in which they are described in more details.

Section 3 reports integration and validation results for the three scenarios mentioned above. Section 3.1 deals with the link-level scenario. Notice that a link level simulator was developed in OAI and was used to derive results for WP4. Here we report part of the work done on OAI software at PHY level, more linked to fundamental procedures, namely an interference cancellation receiver for decoding the control channel coming from different CHs on the same resources.

Section 3.2 reports more details of the implementation of all the modifications from MAC and up to RRC level. We stressed some parts of this implementation namely the virtual link building through the establishment of cooperative Data Radio Bearers (DRB), the filling-up of the MAC forwarding tables, the queuing/buffering at MAC, the MAC forwarding functionality in itself, and others. Latency and throughput results for the virtual link when MAC forwarding is activated is presented and compared with the case where the forwarding is performed at the higher layer.

Section 3.3 is currently empty and will contain the results of the RF validation scenario, which will be inserted after the final demonstration.

Finally Section 4 concludes the document.

2. TESTBED 3 SETUP

2.1. Validation scenario

Testbed 3 aims at validating some of the concepts presented in LOLA for Topology B or, in other words, the static LOLA mesh network.

For clarity's sake, we briefly recall that the LOLA mesh network is a wireless mesh network built on top of LTE. Each cluster is mapped onto an LTE cell; the Cluster Head (CH) plays the role of the eNodeB, while the Mesh Router (MR) inherits procedures from the User Equiment (UE). Hence, the primary roles of a CH are the following: (i) to coordinate user traffics, (ii) to manage physical (i.e. radio) resources within a cluster and (iii) to route user traffic across the network. The main roles of an MR are: (i) to route user traffic using allocated physical resource and (ii) to relay control information between CHs or between MRs. An MR or CH can take also the role of edge router, which acts as a gateway and assures the interface with mobile nodes. Hence, even if an MR inherits procedures from the UE, it still makes part of the mesh network, which is mainly an infrastructure network to be used in absence of the core network. The LOLA mesh network is based on IP and typically routing at IP level can be done in any node, MR or CH (see Figure 5). For more details, please refer to Sect. 2.2 and following.

The aim of testbed 3 is to validate the following three concepts:

1. Virtual link at IP level.
2. MAC forwarding.
3. Cooperation at MAC/PHY layer.

A virtual link is composed of 2 CHs connected by one or more MRs belonging to both clusters and called bridging MRs. When many MRs are used to connect the CHs, they can cooperate. The aim of the virtual link is to build a direct link at IP level in between two adjacent CHs, so that when the packets' route passes through the two CHs, the additional step of routing through the bridging MR can be spared, and the corresponding latency as well. The virtual link is implemented thanks to MAC forwarding at the bridging MRs.

A virtual link is composed of a broadcast phase and of a relaying phase. During those two phases, all the MRs members of a virtual link share the same resources in UL and DL. If cooperation exists in the relaying phase, in LOLA a distributed Alamouti scheme is implemented, combined with a decoding-and-forward (DF) approach. Of course, this is only a specific choice lead by simplicity of both the algorithmic complexity and the standard implementation, and by robustness; other cooperation schemes can be used in principle. DF is done at MAC level and it smoothly combines with the concept of MAC forwarding.

Figure 1 presents the general validation scheme for testbed 3 and the three concepts which are under study.

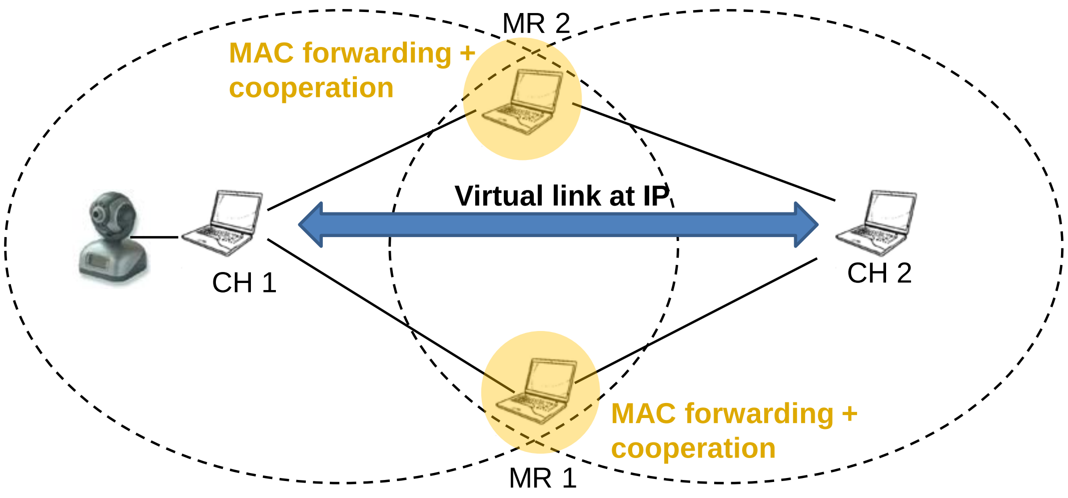


Figure 1: General validation scenario

To reach the global validation, WP5 has proceeded through the following steps:

- Validation at link-level.
- In-lab system validation.
- RF validation.

The previous steps are represented from Figure 2 to Figure 4.

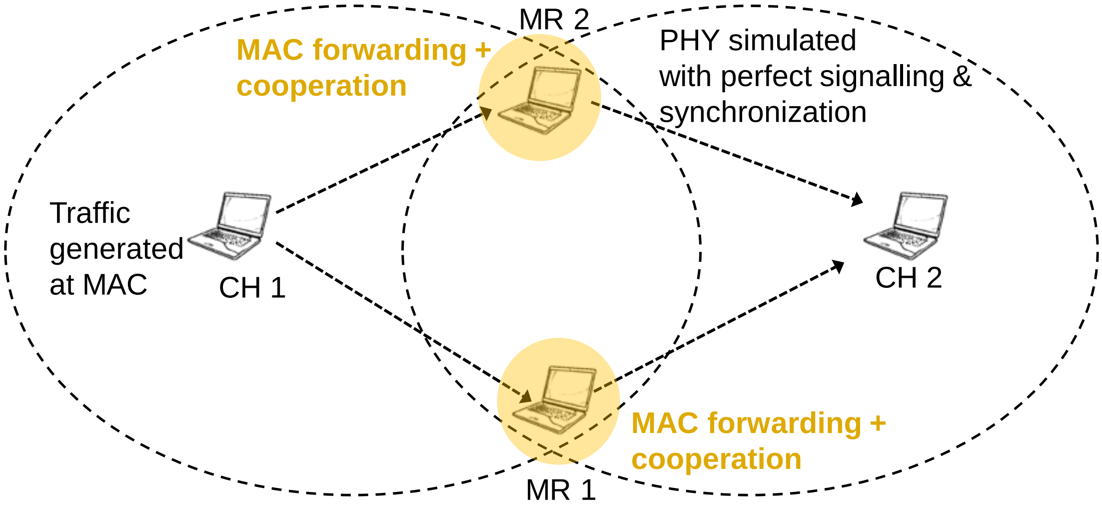


Figure 2: Link-layer validation scenario, used in WP4

Figure 2 presents the validation at link-level: in this step we developed code on OAI concerning mainly the cooperation with distributed Alamouti combined with DF and, in minor way, the MAC forwarding operation. This code was included in a link-level simulator which was used in WP4 to give simulation results on the behaviour of the HARQ strategy selected in LOLA [4]. In this step the channel was simulated as well as the PHY data transmission. Signalling, synchronization and random access procedures were considered ideal. Thanks to the modularity of OAI, most of the code can be used also for the RF simulator. This simulator was used to validate the data plane transmission at PHY layer mainly (control plan is ideal).

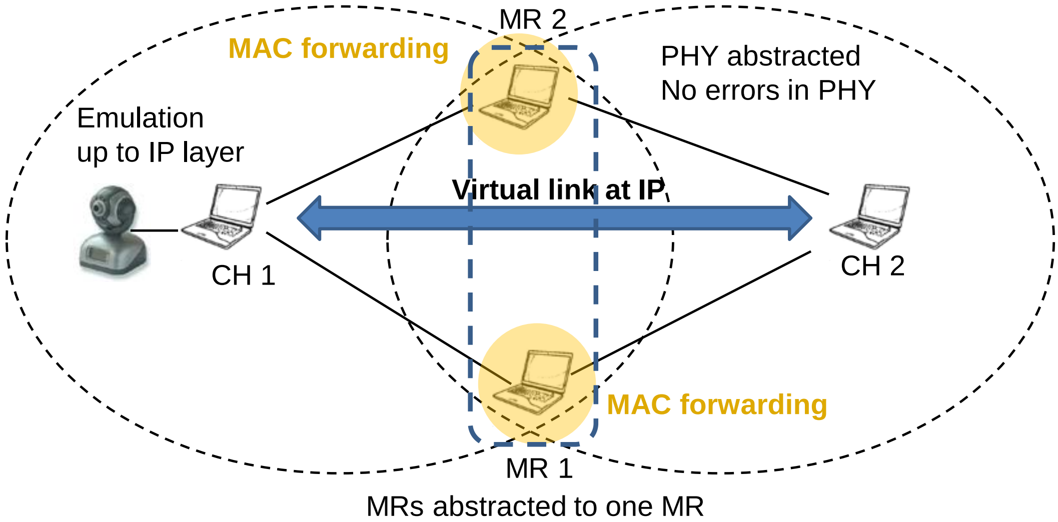


Figure 3: In-lab system validation scenario

Figure 3 presents the validation scenario of the part related to MAC and higher layers. In this scenario PHY behaviour is completely abstracted and it is ideal: all the packets arrive without error and without loss. Under this assumption, simulating one or more bridging MRs is exactly the same, in terms of latency performance. For visualizing this fact a dashed box around all the bridging MRs is drawn in the figure. Of course, the fact that there is more than one bridging MR is different from the point of view of the procedures. For example, more messages must be exchanged to establish a virtual link.

In this set-up, all protocols from MAC level and up are simulated. The only assumption here is that the network is already on, it is clustered and the two-hop neighbourhood information is available to the CHs. Notice that the previous steps, even if important in themselves, are not in the focus of LOLA.

This scenario allows validating:

- the virtual link building through the establishment of cooperative Data Radio Bearers (DRB);
- the new RRC messages, which are exchanged;
- the filling-up of the MAC forwarding tables;
- the queuing/buffering at MAC, for MAC forwarding;
- the MAC forwarding functionality in itself;
- the new DCI, BSR fields;
- and in general the whole protocol stack related to MAC and higher layers.

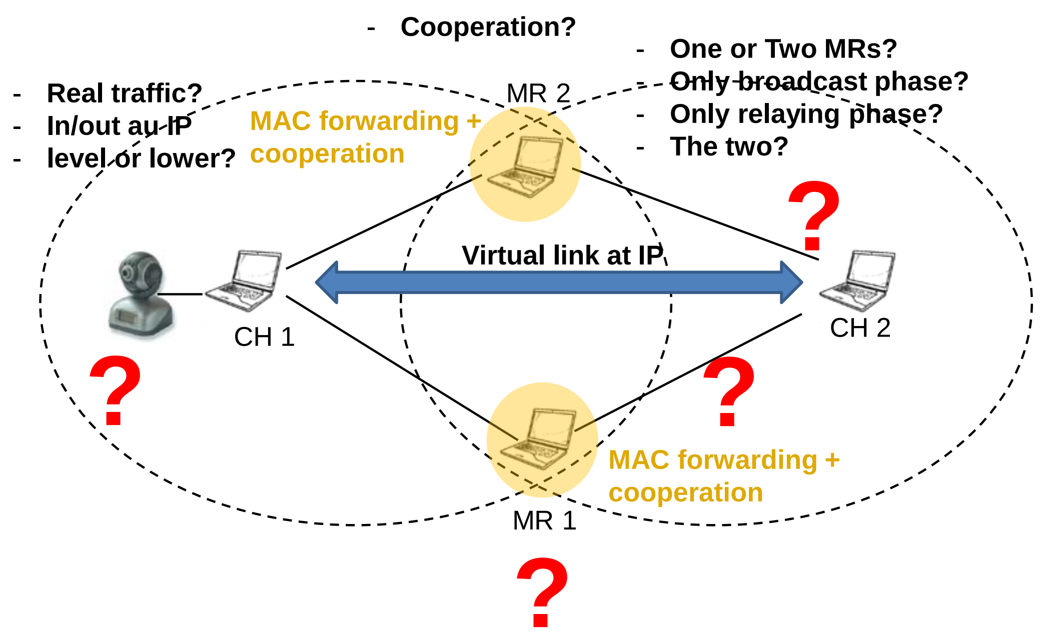


Figure 4: RF validation scenario

Figure 4 presents the realtime RF validation scenario. In this scenario the we will mainly focus on the distributed synchronisation, UE RRC connection procedure with respect to tow CHs, and MAC forwarding based on the OAI RF platforms. The scenario will be 2 clusterheads and one MR demonstrating the relay operation of the MR in support of latency as the RLC, PDCP, and IP layers will be bypassed when relaying is activated.

2.1.1. WP3 Application Traffic

For the application traffic, we have implemented the video streaming traffic model designed in WP3 [7][8] into OpenAirInterface traffic generator (OTG). It allows the generation of multiple random processes with arbitrary but jointly characterized distributions, auto-correlation functions and cross-correlations.

The following parameters is used for this model, where different video streaming data rate, i.e. 10Mbps, 4Mbps, 2Mbps, 768kpbs, 384kpbs, 192kpbs, can be achieved through reconfigurable compression ratio. The setup for the video in the final demo will be 2Mbps (compression 5) using the following configuration.

```
video->tarma_size.inputWeight[0]=1;
video->tarma_size.maWeight[0]=0.47;
video->tarma_size.maWeight[1]=-0.829;
video->tarma_size.maWeight[2]=0.358;
video->tarma_size.arWeight[0]=1;
video->tarma_size.arWeight[1]=-1.984;
video->tarma_size.arWeight[2]=0.984;
video->tarmaVideoGopStructure[0]=0; /*i frame*/
video->tarmaVideoGopStructure[1]=2; /*b frame*/
video->tarmaVideoGopStructure[2]=2;
video->tarmaVideoGopStructure[3]=1; /*p frame*/
video->tarmaVideoGopStructure[4]=2;
video->tarmaVideoGopStructure[5]=2;
video->tarmaVideoGopStructure[6]=1;
video->tarmaVideoGopStructure[7]=2;
video->tarmaVideoGopStructure[8]=2;
video->tarmaVideoGopStructure[9]=1;
video->tarmaVideoGopStructure[10]=2;
video->tarmaVideoGopStructure[11]=2;
video->polyWeightFrame[0][0]=55400/compression; /*i frame*/
video->polyWeightFrame[0][1]=32300/compression;
video->polyWeightFrame[0][2]=10400/compression;
video->polyWeightFrame[0][3]=-100/compression;
video->polyWeightFrame[0][4]=-800/compression;
video->polyWeightFrame[0][5]=74/compression;
video->polyWeightFrame[1][0]=20900/compression; /*p frame*/
video->polyWeightFrame[1][1]=19100/compression;
video->polyWeightFrame[1][2]=7150/compression;
video->polyWeightFrame[1][3]=-740/compression;
video->polyWeightFrame[1][4]=-130/compression;
video->polyWeightFrame[1][5]=20/compression;
video->polyWeightFrame[2][0]=11700/compression; /*b frame*/
video->polyWeightFrame[2][1]=10300/compression;
video->polyWeightFrame[2][2]=4320/compression;
video->polyWeightFrame[2][3]=700/compression;
video->polyWeightFrame[2][4]=-90/compression;
video->polyWeightFrame[2][5]=-2/compression;
```

2.1.2. Key Performance Indicators

In the LOLA project we focus on delay and in special on the one way delay in the network. The considered KPI for this testbed is

- Average One Way Delay OWD (Direction, Datarate, Users, Traffic Pattern). The delay is extended for the number of users of the application, the datarate and the traffic pattern used.

2.2. LTE MESH extension: overall solution revisited

In this and the following sections we present an overview of the overall LOLA solution concerning static mesh networks based on LTE.
Most of the material here is closely related to LOLA D5.6 v1.1 and is reported here for two main reasons:

1. During the development phase some specifications has evolved and these updates are reported here.
2. For sake of clarity in order to present in a self-contained way the LOLA mesh solution.

The general setup of a LOLA mesh network can be represented as in Figure 1.

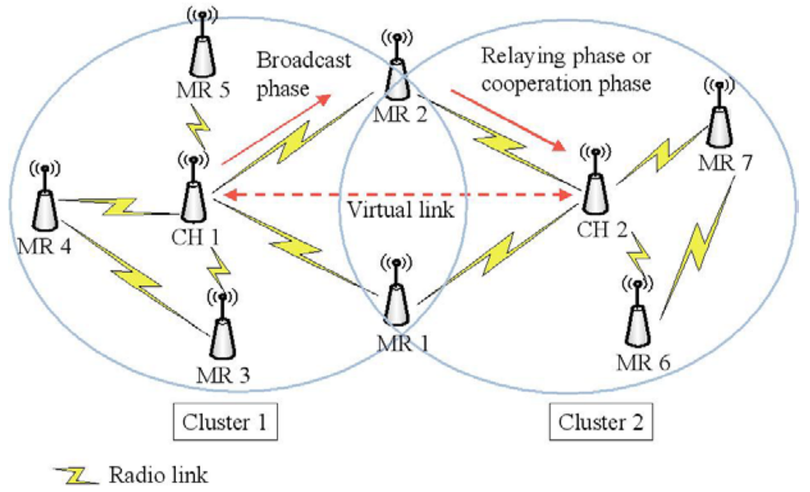


Figure 5: Topology B with a virtual link

In this section we briefly present the retained solution in terms of structure of signalling and fundamental procedures.
The transmission of a transport block from the source CH to the destination CH is briefly presented in this section. In this brief presentation it is assumed that the forwarding

tables and the virtual link have already been configured. Further details on how this can be achieved are described in section 2.4.1.

2.2.1. Exchanges in the source cluster

2.2.1.1. Transmission in the broadcast phase

When the source CH wants to send a MAC PDUs over the cooperative virtual link, i.e. in the DL towards the bridging MRs, it must add a MAC PDU sequence number in the MAC header (in order to identify the MAC PDU at the MR side). Then:

- The source CH schedules it for transmission over the PCHSCH (analogous of the PDSCH), through a scheduling instruction sent over the CHCCH (analogous to the PDCCH), where the only difference if that the DCI is created with the good CO-RNTI.
- The timing of scheduling messages and data transmission follows the standard procedures of LTE in TDD.
- When the packet is sent, if HARQ is active, a HARQ process is created at the source CH, and it is managed in the standard way.
- The ACK/NACK messages are transmitted in subframes which are determined in the standard way according to the synchronous ACK/NACK feedback process of LTE.

2.2.1.2. ACK/NACK signalling in the broadcast phase

The HARQ process in the CH and in MRs follows the standard timing in TDD, and the ACK/NAC messages as well.

For each virtual link in the source cluster, the LOLA solution defines the so-called implicit ACK/NACK. For each CO-RNTI, all the bridging MRs in the virtual link *always* send their ACK/NACK over a common PMRCCH (analogous to PUCCH), by using a PUCCH format 1, the one for the Scheduling Request (SR), in the following way:

- The presence of the signal is a NACK.
- The absence of the signal is a ACK.

Notice that the CH cannot know which MR is sending an ACK, but will continue to send retransmissions until all the MRs in the virtual link send an implicit ACK. As already mentioned in LOLA D4.3, this mechanism can produce useless retransmissions in the source cluster.

In order to reduce the possible resource wastage on the first hop, the LOLA solutions defined also the so-called explicit ACK/NACK, which is sent over the same resources of the implicit ACK/NACK.

The explicit ACK/NACK must be sent by the MRs which receive from the second CH (the destination one) an ACK of the correct reception of the MAC PDU in the relaying phase.

All the MRs belonging to the same virtual link use the same code for the explicit ACK/NACK.

For more details on the specifications please refer to Sect. 2.4.7.

2.2.2. MAC in the bridging MRs of a virtual link

Since forwarding is done at MAC level, the MAC in the bridging MRs of a virtual link must:

- Check and discard multiple MAC PDUs from the source CH.
- Maintain HARQ processes for the virtual link for the MAC PDUs which have been scheduled by the destination CH, even if the MR has not yet sent them (because, for instance, it has not yet received them).
- Insert the MAC PDU in forwarding queues or buffers.
- Generate Buffer Status Report (BSR) or SR when a new MAC PDU (or some new MAC PDUs) arrives in the forwarding queues.

We notice that the synchronization of the MAC buffers of the bridging MRs for cooperative transmission is done in a distributed way thanks to modified scheduling instructions (DCI) from the destination CH.
In order to understand the operation of the MAC in the MRs, we give a simple example. Assume that two MRs belong to the virtual link and that the first one correctly receives a MAC PDU at the first transmission, while the second MR correctly receives the MAC PDU at the third transmission. The source CH continues to send the packet up to the third transmission, such that the first MR must discard the multiple copies. Moreover, the first MR will ask the destination CH for resources independently of the behavior of MR2, which is unknown to it (MRs do not exchange control information). When the destination CH schedules the resources for the first MR, the second MRs shall overhear the scheduling information and create a HARQ process, even if it has no MAC PDU to be sent. This is required because, if the second MR correctly decodes the MAC PDU from the first hop before that the HARQ process in the second hop is ended, then it can cooperate to the transmission toward the destination CH starting at the good HARQ round. Notice that, at this aim, the HARQ state machine for the management of the process in the second MR must be created not only with the good HARQ idendifier, but must also linked to the MAC PDU Sequence Number (SN).

2.2.3. Exchanges in the destination cluster

The bridging MR(s) wanting to send a MAC PDU will do the following:

- If there is no PMRSCH allocated (with the RNTI of each bridging MR) for BSR transmission, then a SR must be sent:
  - The MRs send a SR over their standard PMRCCH (analogous to the PUCCH) located by the code of the MR. All the MRs send the SR with the standard PUCCH format 1.
  - When the destination CH receives a SR it schedules *separate* resources for all the bridging MRs having sent a SR, in the standard way. These resources will enable the MRs having sent the SR for the virtual link to send a modified BSR (called CO-BSR in the following).
- If the interested MRs have already been allocated some PMRSCH resource, they can send their CO-BSR with the CO-RNTI (for localizing the virtual link), the MAC PDU sequence number, as well as the buffer size.
- Upon reception of the CO-BSRs, the destination CH decides which MAC PDU to schedule and sends the scheduling information through a modified DCI containing also the MAC PDU sequence number, which is used by the MRs for buffer synchronization. This new DCI format is sent over a CHCCH (analogous to PDDCH) by using the CO-RNTI and it is decoded by all the MRs of the virtual link.

  Moreover, the destination CH schedules other separate resources (i.e. with their specific RNTI) for the next CO-BSR for all the MRs, if necessary. CO-BSR must be scheduled on separate channels because the BSR information may depend on the MR.
- Upon reception of the scheduling information over the CHCCH with the right CO-RNTI, all the MRs (even those not having the MAC PDU with the right sequence number in their queues) create a (virtual) HARQ process associated to the SN contained in the scheduling information.
  - The MRs having the MAC PDU with the right SN in the queue send the MAC PDU to the (virtual) HARQ process, for transmission.
  - The MRs not having the MAC PDU with the right SN do not send anything.
  - The virtual HARQ processes in the MRs are in fact different instances of a unique HARQ process shared among the group of MRs participating to the virtual link, and the destination CH. These processes are all synchronized and follow the timing and procedures described in the standard.

- The MRs with the right SN will send the information on the scheduled resources for the PMRSCH with the CO-RNTI.
  - Each MR uses full power.
  - Each MR sends the data according to one virtual antenna of the Alamouti distributed space-time code, which is supposed to be determined during the virtual link creation.
- The destination CH tries to decode the UL information and sends ACK/NACK messaging over the PHICH as described in the standard.
- All MRs overhear the PHICH with PHICH index corresponding to the virtual links, i.e. CO-RNTI, they belong to, and update accordingly their HARQ processes.

2.2.4. Exchanges for inter-cluster CQI

In the LOLA mesh, it is assumed that the channel is reciprocal, hence the MR can deduce the CQI of the UL in the destination cluster.

Inter-cluster CQI is dealt with in LOLA by sending aperiodic CQI reporting on the PMRSCH of each MR (i.e. associated to their specific RNTI).

When a CH wants to have a CQI report on the other cluster, for a specific MR, it schedules PMRSCH resources for that MR, by using its RNTI and a modified DCI, in which there are two bits (instead of one) for the request for aperiodic CQI.

When the MR receives this new DCI format 0 with the reporting of the CQI for the other cluster, it encodes CQI information as per the LTE standard, the only difference is that it will send two (or more) such messages, the first one referring to the link between the CH having done the request and the MR, and the second one referring to the link between the MR and the CH of the other cluster of the virtual link, and so on for all the virtual links which the MR belongs to.

2.3. LOLA Mesh PHY Layer

This section is a reminder of the main parameters used by the LOLA mesh PHY and it gives some updates on the specifications of the mechanisms which are present in the LOLA mesh.

2.3.1. PHY layer parameters

LOLA mesh PHY and MAC layers are built on top of 3GPP LTE. All the communications are based on OFDMA and the selected duplex mode is TDD.

In the following it is assumed that nodes are equipped with a unique transmit and receive chain (no multi-radio equipment, which would simplify the job here). Additionally, DL is used to indicate CH-to-MR direction and UL is used to indicate MR-to-CH or MR-to-MR direction.

In the LOLA mesh, all cells operate with the same bandwidth and at the same frequency. Therefore, frequency planning is not needed when deploying a network. However, the use of a single frequency for the cells results in two main issues that must be considered: frame alignment and handling of interference between cells. Frame alignment is achieved through distributed synchronization and is explained in Sec. 2.3.2. Interference is handled by a variety of techniques and is discussed in Sec. 2.3.6.

The LOLA mesh can use the normal and the extended CP (16.7 μs), with a preference for the latest in order to have more robustness to network synchronization errors.

The role of the CP in a mesh network may change due to the fact that, contrary to a standard cellular network, there can be more than one receiver when multiple transmitters are active in the same subframe (which may happens in UL subframes, for inter-cluster communications for instance). In a standard cellular network the CP must cover only the delay spread, thanks to the time-advance technique. The propagation delay is managed by the guard times present in the frame when switching from transmission to reception and vice versa. In the LOLA mesh standard time advance (TA) cannot always be used, for example in the case of inter-cluster communications with more than one bridging MRs cooperating together. For other LOLA mesh configurations in which the standard LTE TA may not work, please refer to LOLA D5.6 v1.1 [6] or LOLA D4.2 [5].

In these cases, the CP can have also a role for limiting interference by absorbing relative propagation delays.

Parameter	**Unit**	**Value**
Duplex mode		TDD
PHY layer technique		OFDMA for all the communications
Basic time unit T_s	s	1/(15000*2048) = $3.255 \cdot 10^{-8}$
Subcarrier spacing Δf	kHz	15
Cyclic prefix length $N_{CP,l}$	samples	512 for 15 kHz subcarrier
Cyclic prefix length	μs	16.7 for 15 kHz subcarrier

Table 1: PHY parameters of the LOLA mesh

2.3.2. Synchronization (updates)

In the LOLA mesh, network synchronization is done in-band using distributed procedures and dedicated channels. In a network, one CH is designated as master and provides a time reference for the network. MRs that connect to the master CH start transmitting in the mesh router PHY synchronization channel (MRPSCH) (defined in Sec. 2.3.3.2). CHs other than the master are silent until they initially receive the MRPSCH, at which point they activate. MRs in its range then connect to it and propagate the synchronization further.

2.3.2.1. Cluster head PHY synchronization channel

The synchronization in the DL, i.e. from the CH to the MR, is done as in the standard using the Primary and Secondary Synchronization Sequences. An MR can synchronize at the same time with at most 3 CHs. For more details, please refer to LOLA D5.6 v1.1 [6].

2.3.2.2. Mesh router PHY synchronization channel

Mesh router PHY synchronization channel (MRPSCH) is used:

- For continuous network synchronization due to time drift of the equipment clocks
- For network synchronization during network building / entry.

For possible uses of the MRPSCH for direct MR-to-MR communications, please refer to LOLA D5.6 v1.1.

The MRPSCH is sent in the first S-subframe of the frame, in the last symbol of the UpPTS. Currently in the second S-subframe of the frame there is no MRPSCH (space is free for possible future evolutions, for instance).

MRPSCH uses UpPTS, but please notice that:

- the UpPTS can also be used for the random access preamble number 4 for the PRACH (see Sect. 5.7.1-1 in [9], or Section 8.2 in [4]). This means that for LTE mesh, PRACH will use configured subframes, or, if possible, only the UpPTS left free by the MRPSCH.
- The UpPTS can also be used to SRS, so in the LOLA mesh SRS must not be used in the UpPTS part used by the MRPSCH.

All the MRs in the network send the same signal inside the MRPSCH.

The MRPSCH is inherited from the PSS, in the sense that it occupies the central 62 subcarriers (excluding DC). The parameter of the PSS is different from the one of the CH, in particular today we use the root index u = 23 (the other root indexes used in standard PSS are 25, 29, 34).

2.3.3. Broadcast messages (updates)

Broadcast messages are of fundamental importance for the mesh operation. The CH broadcast channel (CHBCH) is strongly inherited from LTE.

The MRBCH relays small amounts of signalling data to nodes entering the network in the network set-up phase and for distributed network synchronization.

2.3.3.1. Cluster head broadcast channel

The CHBCH is inherited from LTE. It consists of the PBCH, PCFICH, PHICH, PDCCH, and the PDSCH transmissions containing System Information Blocks. For more details, please refer to LOLA D5.6 v1.1 [6]. The channels have been adapted to enable suppression of interference in the single-frequency scenario of LOLA. These changes are further discussed in Sec. 2.3.6.

2.3.3.2. Mesh router broadcast channel

In the LOLA mesh the MRBCH can be implemented in two ways depending on the size of the information to be conveyed, and the type of mesh procedure.

The MRBCH is sent in the UpPTS part of the S-subframes. It implies that it can be either absent or be one or two OFDM symbols long. Depending on the presence of the MRPSCH, certain RE of the last OFDM symbols may not be available to the MRBCH. For instance, in the channel configuration at 1.4 MHz, the PMRSCH occupies the whole last OFDM symbol of the UpPTS part, so the MRBCH can use only the previous OFDM symbol. As for the MRPSCH, a cooperative MRBCH is sent in the UpPTS part of the first S-subframe. All the MRs send the same information, it is a cooperative MRBCH. The MRBCH can be used during network setup and in principle a few bytes shall be sent.

A second type of MRBCH is specified which can also convey information related to the specific MR. However, this MR-specific MRBCH will not be implemented. The allocation of the cooperative MRBCH and of the MRBCH to a given MR and their periodicity are done either in a static way or by the CH during the connection establishment. For this aim, a function in the RRC block must be created.

The decoding of the MRBCH can be eased by the presence of an MR pilot pattern (if any), e.g. for channel estimation, or by the use of the same PHY layer format of PDCCH with reference signal equal to one pilot per antenna every six REs. Scrambling according to the identity of the MR can also be applied. The amount of information that can be reasonably expected to be sent is similar to that which can be sent in a PDCCH with one or two OFDM symbols, hence probably 72 or 144 information bits. The size of the band has also an influence: for a band with a number of PRBs greater than 10, one OFDM symbol should be sufficient for conveying the information. For a band with less than 10 PRBs, it will probably be necessary to use more than one OFDM symbol for the MRBCH. The exact coding and pilot design is still to be specified.

The second option which will probably not be implemented is that the MRBCH is sent either in subframe 2 (it is always an MR-to-CH subframe for all TDD frame configurations) or in subframe 7 (it is always an MR-to-CH subframe for all TDD frame configurations with S-subframe periodicity of 5 ms), or both of them. In this case the MRBCH will follow the specifications of an LTE control channel like PDDCH (exact specification is to be done), see Section 3.3.5.1.

2.3.4. Direct mesh router communication

The LOLA mesh has included the possibility for the MRs to directly communicate between them inside the same cluster, according to the scheduling decision of the CH.

This feature has not been implemented in the final testbed and its specifications have not evolved since the last report. The interested reader is invited to refer to Sect. 3.2.5 of LOLA D5.6 v1.1.

2.3.5. Frame parameters

Sect 3.2.6 of LOLA D5.6 v1.1 presents the general final frame design; please refer to it for a general presentation. Here, we present the updates and specifications retained for the particular implementation of testbed3.

The preferential TDD configurations for the LOLA mesh are shown in Table 2. It is worth recalling that a 'D' means a CH-to-MR subframe (CH TxOps), while a 'U' means a MR-to-CH or MR-to-MR subframe (MR TxOps).

Uplink-downlink configuration	Downlink-to-Uplink Switch-point periodicity	Subframe number									
		0	1	2	3	4	5	6	7	8	9
0	5 ms	D	S	U	U	U	D	S	U	U	U
1	5 ms	D	S	U	U	D	D	S	U	U	D
2	5 ms	D	S	U	D	D	D	S	U	D	D
6	5 ms	D	S	U	U	U	D	S	U	U	D

Table 2: Uplink-downlink configurations with switch-point periodicity of 5 ms

In the testbed, uplink-downlink configuration 1 has been used, and different resource blocks have been equally allocated to each clusterhead.

2.3.6. PHY layer procedures (updates)

Several of the physical channels in LTE have been adapted for LOLA in order to deal with interference between the cells. In fact, for instance, control channels are common for adjacent clusters while data channels are separated in the PHY (see Sect. 2.3.6.4. In most cases, the main issue is that pilots of one cell overlaps with resource elements used for data in another cell, and this degrades the quality of the channel estimations and significantly decreases the performance. Therefore, the aim has been to puncture the bits in the pilot symbols whenever possible.

2.3.6.1. Synchronization and broadcast channels

The synchronization and broadcast channels consist of the Primary Synchronization Signal (PSS), the Secondary Synchronization Signal (SSS) and the Physical Broadcast Channel (PBCH). In LTE, these are designed to be detected in neighboring cells (with strong interference from the serving eNodeB), and thus no changes are required in LOLA.

2.3.6.2. Downlink control channel

In LTE, the PDCCH consists of one to three symbols in the beginning of each DL subframe. In LOLA, since the first symbol is a pilot symbol, the PDCCH is always three symbols, with the bits in the first symbol punctured. In addition, the aggregation level of the user-specific DCIs is increased. To receive the PDCCH, an interference cancelling receiver is required at the MRs.

The restriction of the PDCCH length to three symbols makes the PCFICH superfluous. However, it is retained in LOLA in order to keep compatibility with LTE.

2.3.6.3. Hybrid ARQ indicator channel

The PHICH consists of a number of BPSK-modulated resource elements in the first symbol in each DL subframe. The spreading of the coded bits is sufficient for successful decoding in the presence of heavy interference, and thus no changes have been made to this channel.

2.3.6.4. Traffic channels

In principle, an interference cancelling receiver could be used for the traffic channels (PDSCH and PUSCH). However, since this is peripheral to the interests of LOLA, the final testbed uses a static partition of the spectrum between the cells.

2.4. Higher layers and MAC

This section contains updates with respect to the corresponding Sect. 4 of LOLA D5.6 v1.1.

2.4.1. RRC layer / Virtual link set up (updates)

2.4.1.1. Input requirements

The network clustering shall have been performed before establishing a virtual link. Each node in the network shall know its role in the clusterized network (i.e. CH or MR). Each CH in the network shall know the sets of MRs connecting it to the other CHs two radio hops away.

Each virtual link between CHs shall be identified by a virtualLinkID.

A virtual link shall be composed of two CHs and at least one MR acting as a relay between them. The number of relays of a virtual link shall be comprised between 1 and MAX_MR_PER_VLINK. The number of virtual link for a CH is comprised between 1 and MAX_VLINK_PER_CH.

In order to satisfy the requirements above without a clustering function, a static table describing all the virtual links shall be available for a CH scheduler. Each 100 frames, on subframe 1, the scheduler of a CH checks all the virtual links described and tries to set them up.

The table struct virtual_links virtualLinksTable[NB_MAX_CH] gathers all the virtual links for all the CHs of the networks. NB_MAX_CH is number maximum of CHs in the LOLA topology B network. The indexes of the table represent the eNB indexes in the OpenAirInterface simulation.

The structure struct virtual_links gathers all the virtual links for a specific CH.

```
struct virtual_links {
        int count;
        struct virtualLink array[MAX_VLINK_PER_CH];
}
```

The structure “struct virtualLink” describes a virtual link and it gathers the virtual link ID, the source CH, the destination CH and the list of MRs part of the virtual link.

```
struct virtualLink {
        unsigned char virtualLinkID; // Virtual link ID
        unsigned short PCellIdsourceCH; // Source CH (not used)
        unsigned short PCellIddestCH; // Destination CH (not used)
        struct mr_array MRarray; // MRs of the virtual link
        unsigned char status;  // Connected or Not connected
}
```

The structure struct mr_array gathers all the MRs members of a virtual link. The indexes of the array represent the UE indexes in the OpenAirInterface simulation. These indexes are assigned per eNB. The indexes shall be common to all eNBs (i.e UE_index 0 shall represent the same UE on eNB 0 and eNB 1).

```
struct mr_arrays {
        int count;
        char array[MAX_MR_PER_VLINK];
}
```

2.4.1.2. Functional requirements

Each CH of a virtual link shall establish a virtual DRB with each MR of the MR list of the virtual link.

A CH shall use the same virtual DRB id with all the MRs of a virtual link.

A virtual DRB establishment shall be done through a RRCConnectionReconfiguration message generated by the RRC layer of a CH.

In order to establish a virtual link, two new parameters shall be added in the RRCConnectionReconfiguration message:

- The virtual link ID
- The CO-RNTI

The CO-RNTI is assigned by a CH and shall identify uniquely all the MRs members of a virtual link.

A CH establishing a virtual DRB shall be in RRC_CONNECTED mode with the MRs members of the virtual link.

Upon reception of a RRCConnectionReconfiguration message, a MR shall configure its protocol stack:

- RLC layer in UM
- Fill the MAC layer forwarding table (with the received virtual link ID and CO-RNTI)

A MR network entry and a virtual link establishment are illustrated in the figures below (what follows the sign "=>" indicates the result(s) obtained by the previous exchanges).

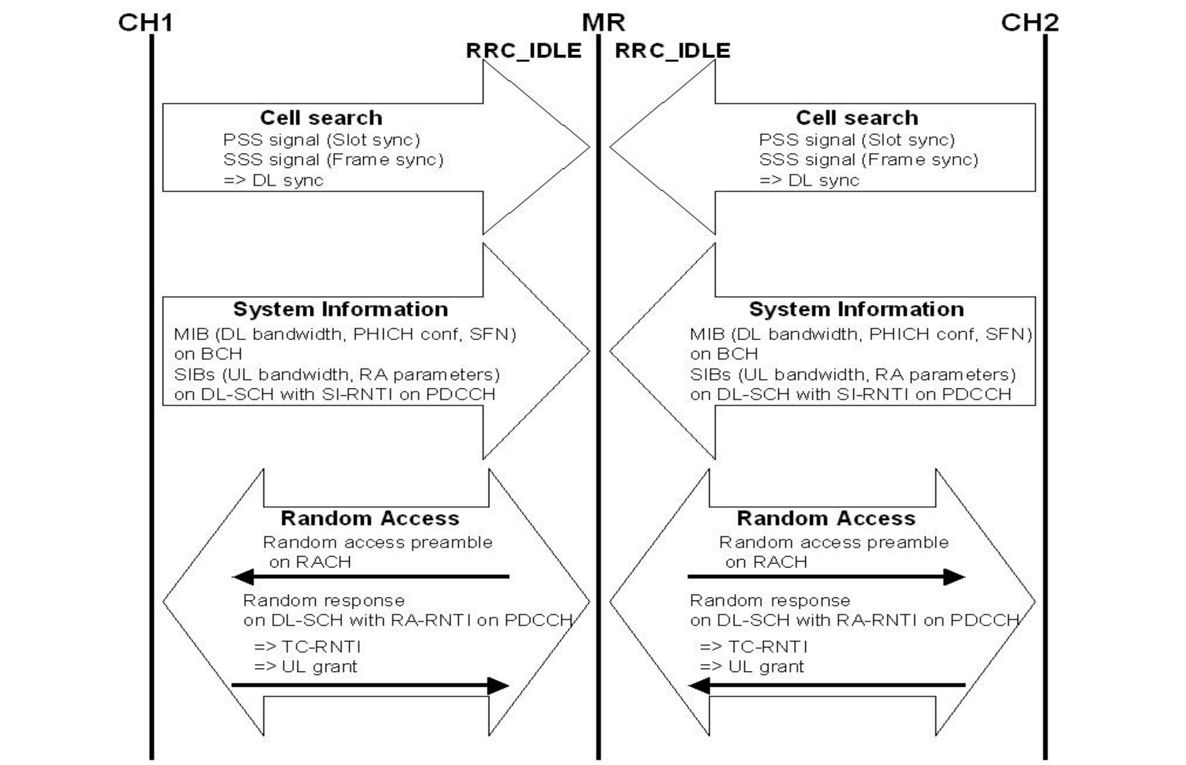


Figure 6 - Network entry 1

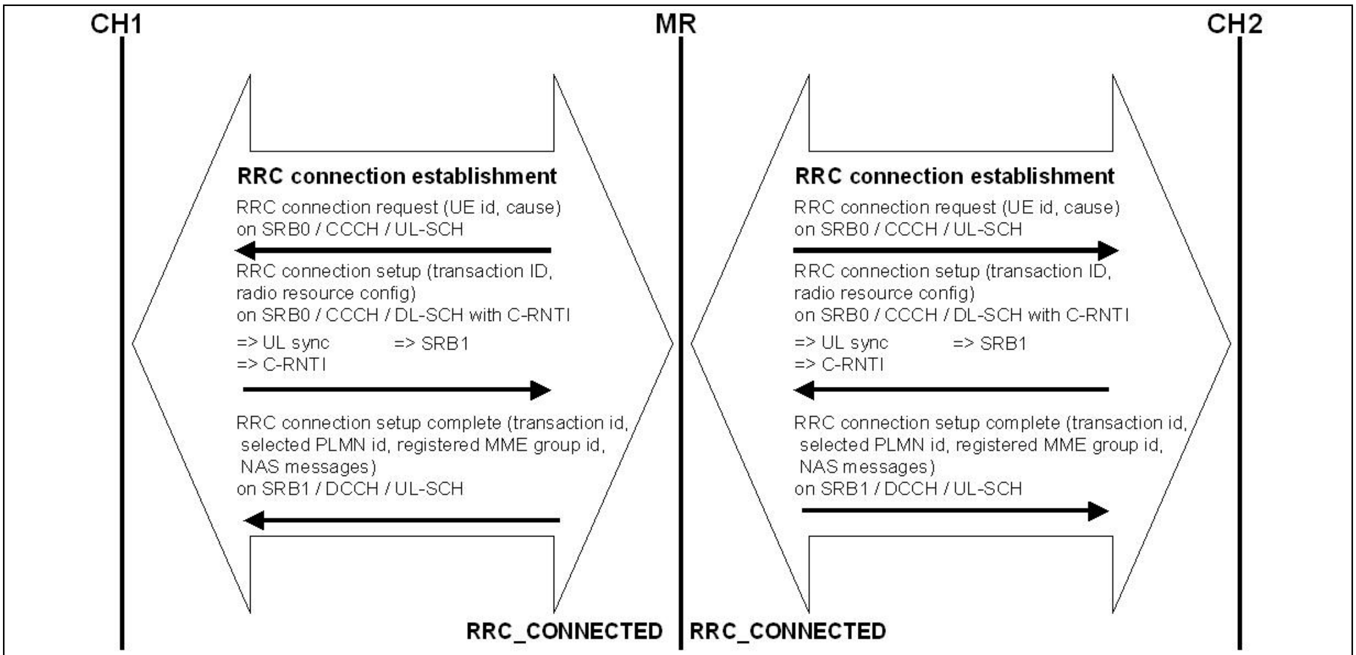


Figure 7 - Network entry 2

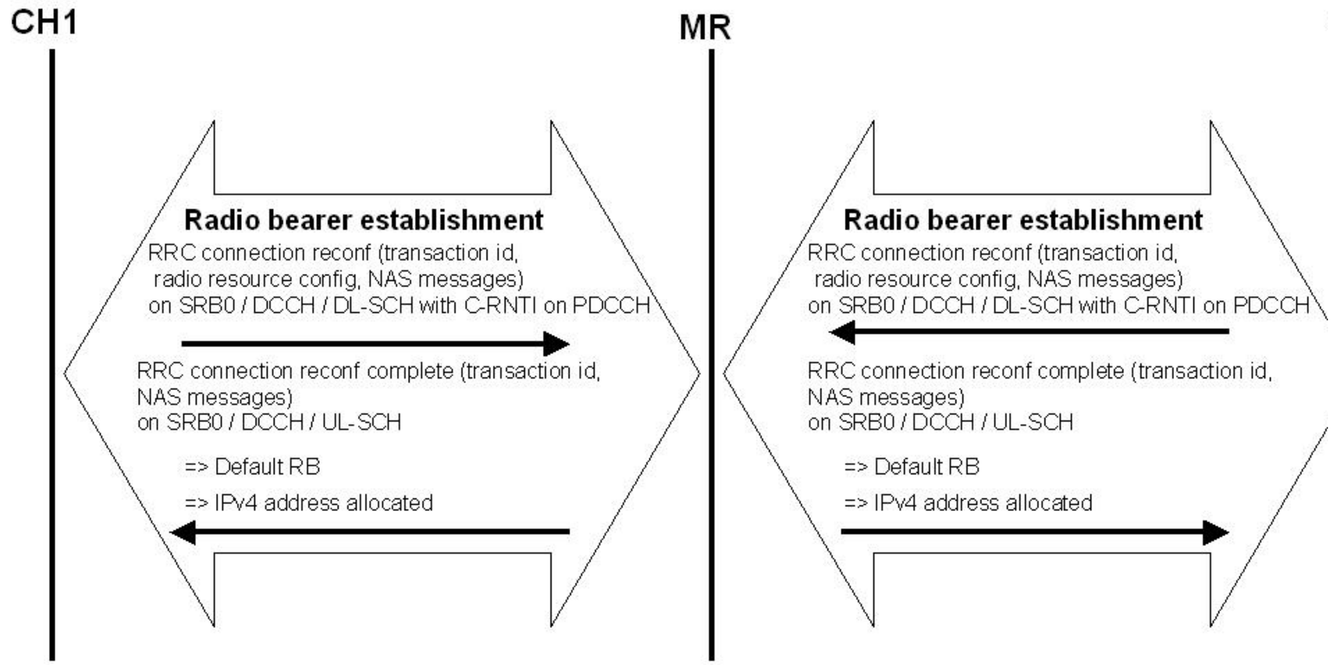


Figure 8 - Network entry 3

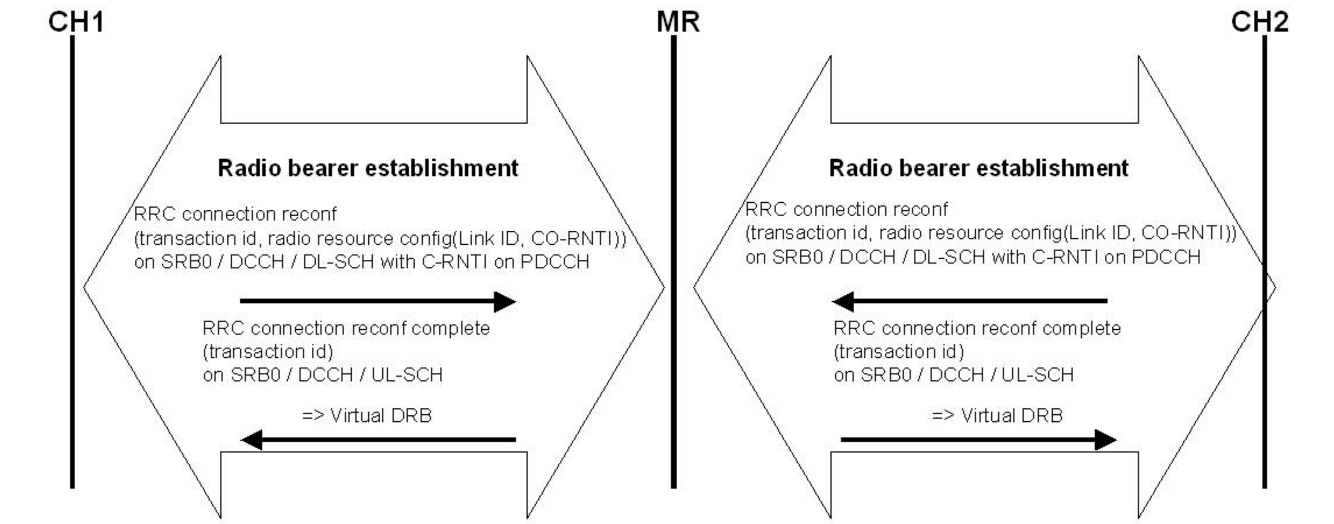


Figure 9 – Virtual DRB establishment

2.4.1.3. PDU requirements

The RRCConnectionReconfiguration message to be generated by the CH in order to set up a virtual DRB shall be as follows (the new RRC message field are highlighted in red):

```
RRCConnectionReconfiguration ::=        SEQUENCE {
        rrc-TransactionIdentifier                   RRC-TransactionIdentifier,
```

```
        criticalExtensions                    CHOICE {
                c1                                    CHOICE{
                        rrcConnectionReconfiguration-r8       RRCConnectionReconfiguration-r8-Ies,
                        spare7 NULL,
                        spare6 NULL, spare5 NULL, spare4 NULL,
                        spare3 NULL, spare2 NULL, spare1 NULL
                },
                criticalExtensionsFuture                      SEQUENCE {}
        }
}

RRCConnectionReconfiguration-r8-Ies ::= SEQUENCE {
        MeasConfig                            MeasConfig                    OPTIONAL,       -- Need ON
        mobilityControlInfo           MobilityControlInfo       OPTIONAL,       -- Cond HO
        dedicatedInfoNASList                  SEQUENCE (SIZE(1..maxDRB)) OF
                                                      DedicatedInfoNAS      OPTIONAL,       -- Cond nonHO
        RadioResourceConfigDedicated          RadioResourceConfigDedicated  OPTIONAL,       -- Cond HO-toEUTRA
        securityConfigHO                      SecurityConfigHO              OPTIONAL,       -- Cond HO
        nonCriticalExtension          SEQUENCE {}                   OPTIONAL        -- Need OP
}

RadioResourceConfigDedicated ::=              SEQUENCE {
        srb-ToAddModList              SRB-ToAddModList                      OPTIONAL,       -- Cond HO-Conn
        drb-ToAddModList              DRB-ToAddModList              OPTIONAL,       -- Cond HO-toEUTRA
        drb-ToReleaseList             DRB-ToReleaseList             OPTIONAL,       -- Need ON
        mac-MainConfig        CHOICE {
                explicitValue         MAC-MainConfig,
                defaultValue          NULL
         }                                                          OPTIONAL,       -- Cond HO-toEUTRA2
        sps-Config                            SPS-Config                            OPTIONAL,       -- Need ON
        physicalConfigDedicated               PhysicalConfigDedicated       OPTIONAL,       -- Need ON
         rlf-TimersAndConstants-r9            RLF-TimersAndConstants-r9     OPTIONAL,       -- Need ON
         measSubframePatternPCell-r10         MeasSubframePatternPCell-r10  OPTIONAL,       -- Need ON
        sps-RA-ConfigList-rlola               SPS-RA-ConfigList-rlola       OPTIONAL        -- NEED ON
}

DRB-ToAddModList ::=                          SEQUENCE(SIZE(1..maxDRB)) OF DRB-ToAddMod

DRB-ToAddMod::=                       SEQUENCE {
        eps-BearerIdentity                    INTEGER (0..15)       OPTIONAL,       -- Cond DRB-Setup
        vdrb-Identity                         DRB-Identity,
        pdcp-Config                           PDCP-Config           OPTIONAL,       -- Cond PDCP
        rlc-Config                            RLC-Config            OPTIONAL,       -- Cond Setup
        logicalChannelIdentity                INTEGER (3..10)       OPTIONAL,       -- Cond DRB-Setup
        logicalChannelConfig                  LogicalChannelConfig  OPTIONAL,       -- Cond Setup
        co-RNTI                               INTEGER (0..255)
        virtualLinkID                         INTEGER (0..255)
}

DRB-ToReleaseList ::=                         SEQUENCE(SIZE(1..maxDRB)) OF DRB-ToAddMod
```

2.4.2. General MAC requirements

2.4.2.1. Input requirements

N/A

2.4.2.2. Functional requirements

In order to perform MAC PDU forwarding for cooperative transmissions, the MAC layer shall integrate new functions:

- An RLC PDUs concatenation function
- A forwarding function
- A queuing (or buffering) function

The new MAC layer architecture is illustrated in the figure below.

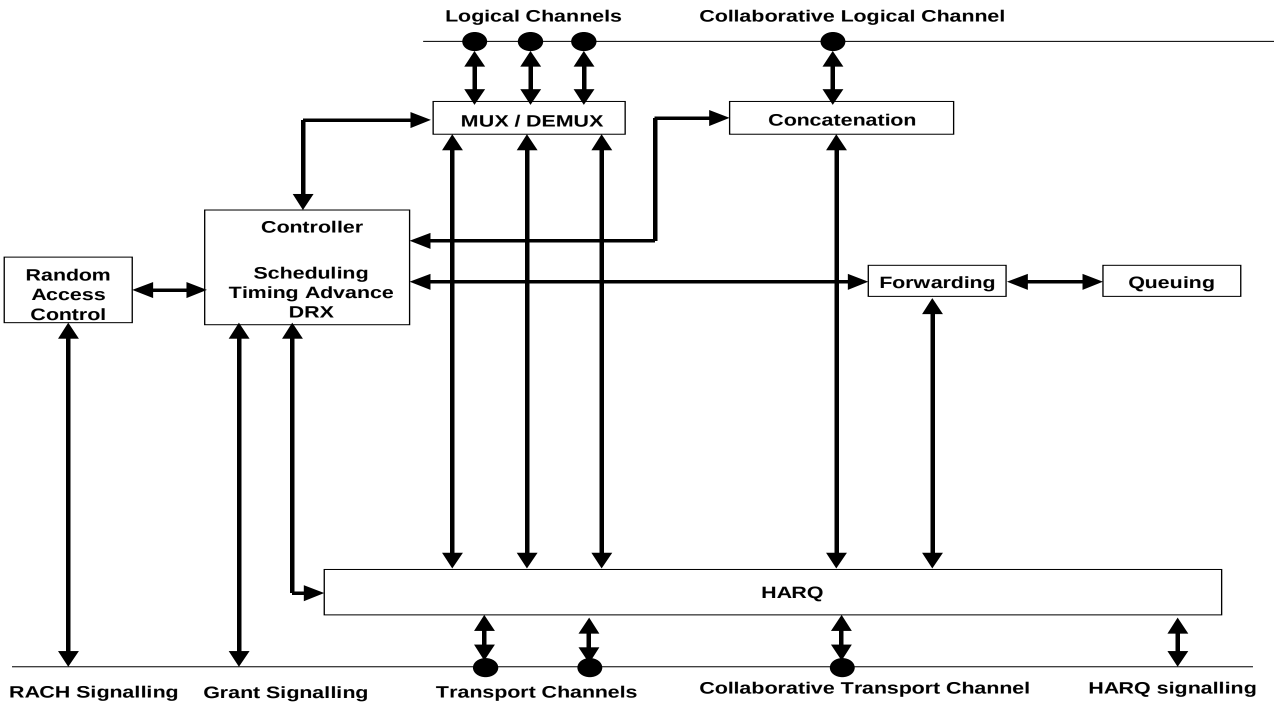


Figure 10 – MAC layer architecture

2.4.2.3. PDU requirements

N/A

2.4.3. Concatenation / MAC PDU building (updates)

2.4.3.1. Input requirements

The concatenation function shall be implemented on CHs only.

2.4.3.2. Functional requirements

Downlink

Upon reception of a Transport Block Size and informed by the MAC controller of a subframe scheduled for a TX broadcast phase of cooperative data, the concatenation function shall request to the RLC layer an amount of RLC PDUs fitting the MAC Transport Block Size.

The concatenation function shall perform the MAC PDUs building including the MAC PDU sequence number.

Uplink

Upon reception of a MAC PDU from the HARQ function, the concatenation function shall extract the RLC PDUs and give them to the RLC layer.

2.4.3.3. PDU requirements

The MAC header shall integrate a MAC Control Element, called sequence number (see Figure 11). The length of the field is 16 bits. A new LCID is used to refer to this control element.

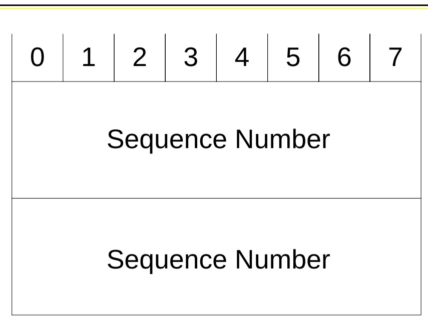


Figure 11 – new MAC Control Element

We make use of the MAC header long to convey the SN through the length field (L). The MAC header long structure is shown below.

```
typedef struct {
  u8 LCID:5;   // octet 1 LSB
  u8 E:1;
  u8 R:2;      // octet 1 MSB
  u16 L:15;      // octet 3/2 LSB
  u16 F:1;      // octet 3 MSB
} __attribute__((__packed__)) SCH_SUBHEADER_LONG;
```

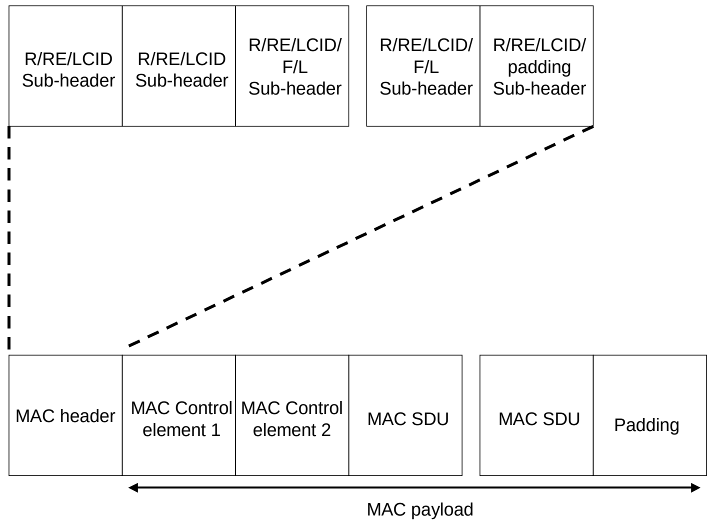


Figure 12 – MAC PDU format

2.4.4. Forwarding (updates)

2.4.4.1. Input requirements

The forwarding function is implemented on MRs only, since incoming packets received by MRs are locally stored in the queues where each MR maintain per associated ingress and egress CO-RNTI, in order to be transmitted later to the destination CH when a scheduling request is decided.

The forwarding table shall have been correctly configured with link IDs and the corresponding CO-RNTI by the RRC layer.

2.4.4.2. Functional requirements

The forwarding function shall implement a forwarding table. The forwarding table shall associate a virtual link ID with an ingress and an egress CO-RNTI.. An example of forwarding table is shown in the table below.

Virtual link id	CO-RNTI (ingress)	CO-RNTI (egress)
VL #1	CO-RNTI #18	CO-RNTI #2
	CO-RNTI #2	CO-RNTI #18
VL #2	CO-RNTI #7	CO-RNTI #24
	CO-RNTI #24	CO-RNTI #7
VL #...	...	...

Table 3: Example of forwarding table

The forwarding function shall maintain a table (named receivedTBTable) of MAX_SEQUENCE_NUMBER_SIZE elements. This table shall be flushed each REFRESH_TIME seconds.

The receivedBTtable table keeps track of the received MAC PDU (i.e SN) in the broadcast phase by the MR as well as the correctly or not correctly MAC PDU sent by the MR or other MRs in the virtual link.

The receivedBTtable table shall be updated on the events that follow:

- Upon the reception of a newly received MAC PDU (avoid duplicate data reception in the broadcast phase), the corresponding index in the table is marked as received
- Upon the ending of a transmit HARQ process related to a virtual link (i.e. related to a specific SN) in the relaying phase. If the HARQ process associated to a MAC PDU with SN *N* successfully terminates, then at the index *N*, the table is updated with a good reception. If the HARQ process terminates unsuccessfully, then at the index *N*, the table is updated with a bad reception.

Downlink

On a subframe scheduled for a RX broadcast phase of cooperative data, at the MR side, the HARQ process shall send the received cooperative MAC PDU to the forwarding function.

The forwarding function shall check the sequence number of a newly received MAC PDU against receivedTBTable and:

- Discard the MAC PDU if the sequence number has already been correctly received (either in the broadcast phase or in the relaying phase).
- Accept the MAC PDU if the sequence number corresponds to an active virtual link HARQ transmit process for the received CO-RNTI and send the MAC PDU to that HARQ process
- Accept the MAC PDU if there is a valid entry in the forwarding table for the received MAC PDU based on the CO-RNTI used in the DCI for the considered cooperative DL resource grant. Then the forwarding function shall send the MAC PDU along with the considered CO-RNTI to the queuing function. Otherwise the forwarding function shall discard the MAC PDU.

Uplink

Upon notification from the MAC controller that the MR is scheduled in UL for a cooperative transmission, the forwarding function shall retrieve a MAC PDU from the queuing function corresponding to the scheduled CO-RNTI and send it to the HARQ function.

Upon request from the MAC controller for MAC PDU with a given SN and a given CO-RNTI the forwarding function shall request the MAC PDU corresponding to this SN and CO-RNTI and:

- Sent it to the virtual link HARQ transmit process associated to the SN if the MAC PDU is queued
- Inform the MAC controller if the MAC PDU is not queued.
- The new header information on the MAC PDU SN may be suppressed from the MAC header since the destination CH does not need it.

2.4.4.3. PDU requirements

N/A

2.4.5. Queuing (updates)

Each MR maintains queues associated to its CO-RNTIs in the MAC layer. Each buffer located in a MR is initiated when the RRC procedures among the respective MR and the CH on connection take place. The buffer is in general a double-linked list under a FIFO operation. However, different or enhanced implementations of a buffer considering hash functionality or even a more sophisticated data structure that can be used to boost performance and reduce latency is acceptable.

2.4.5.1. Input requirements

The queuing function is implemented on MRs only. An API for handling the queuing operations is given in Table 4.

```
 mac_buffer_nb_elements( MR_ID,  CO_RNTI);
mac_buffer_stat_ind(MR_ID, dst_CH, dst_CO_RNTI,payload_ptr, SEQ_NUM,SIZE,HARQ_PID);
mac_buffer_data_req(MR_ID, CO-RNTI, PDU_SIZE, PDU_SN, PDU_HARQ_ID );
mac_buffer_data_ind(MR_ID, CO-RNTI, PDU_SIZE, PDU_SN, PDU_HARQ_ID);
```

Table 4. MAC forwardinfg buffer API

- **mac_buffer_nb_elements(…)** returns the number of PDUs that are currently stored on the buffer that MR maintains for a specific CO-RNTI
- **mac_buffer_stat_ind(…)** this function is used in order to assist the update of the CO-BSR information. A number of packets is requested for transmission and this function updates the CO-BSR providing a short report that includes the sequence number and the pdu size for a sole PDU or a long report for a bunch of PDUs.
- **mac_buffer_data_req(…)** This function is a wrapper of the function that dequeues a PDU element from the MAC buffer queue. For the buffer in MR

identified by the MR_ID and CO-RNTI number, a PDU element is requested for transmission from the UE to the destination CH. A search in the buffer is done for a PDU to identify whether is stored or not using the sequence number and the HARQ id or by a requested size. The latter operation returns any PDU with size less than or equal to the requested size.

- **mac_buffer_data_ind(…)** This function is a wrapper of the function that inserts a PDU element at the tail of the queue. For the buffer in MR identified by the MR_ID and CO-RNTI number, the PDU element is stored along with the PDU_SIZE, PDU_SN and HARQ information.

In Figure 13 the MR interfaces are shown. The interaction of the MAC buffer data indicate and request functions of the MAC layer with the buffer and with the functions for sending and getting a collaborative SDU that interact with the Physical procedures are illustrated. The respective functions are contoured on red boxes.

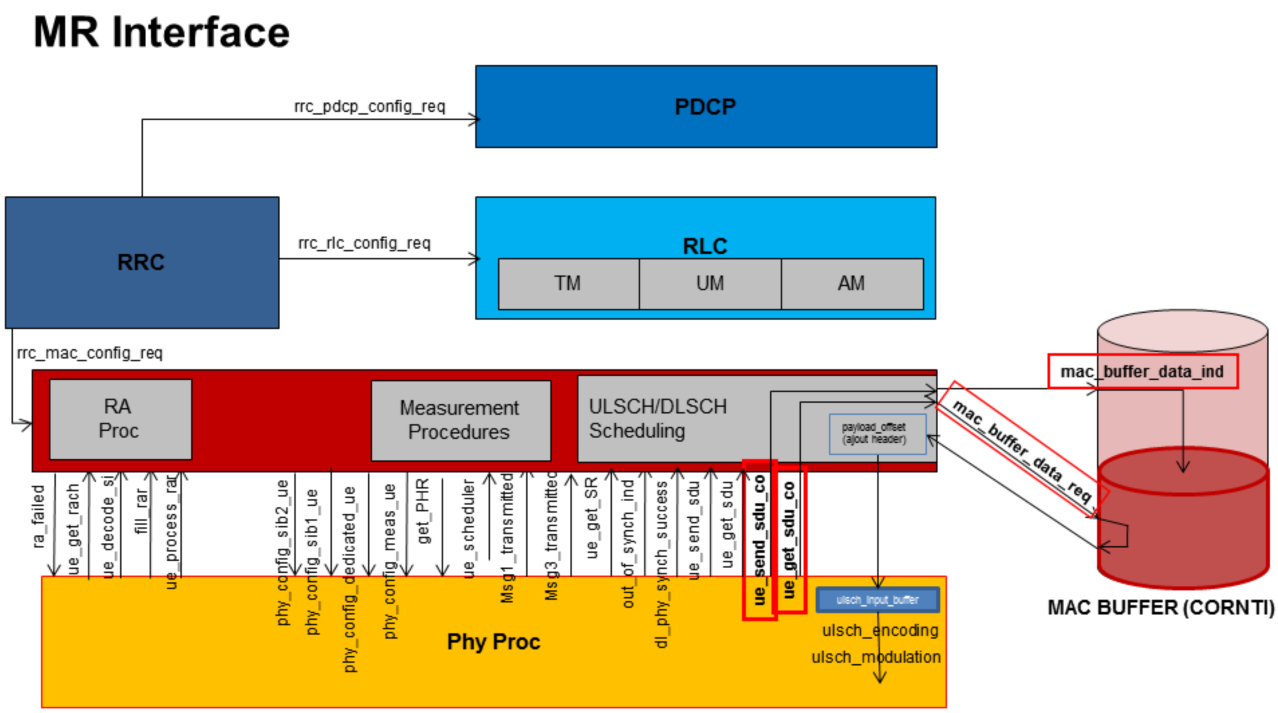


Figure 13. **MR interface functions**

2.4.5.2. Functional requirements

The queuing function implements MAX_CORNTI_PER_MR queues in one MR. The aforementioned value can be interpreted as an upper bound on the virtual links that the respective MR can participate.

Downlink

Each MAC PDU received from the forwarding function shall be queued according to the egress CO-RNTI given by the forwarding function.

The basic role of the queuing functionality in the MR is to prevent collaborative packets from entering the layers after the MAC in the MR's protocol. Once those collaborative packets are identified, they are stored in the respective MR's queue identified by the associated CO-RNTI. Figure 14 illustrates the transmission from source CH to MR that is the downlink from the CH transmission perspective.

Uplink

Upon request from the forwarding function, the queuing function shall retrieve a MAC PDU from queue corresponding to the CO-RNTI given by the forwarding function and give it to the forwarding function. The forwarding transmission is being done when the destination CH schedules a collaborative transmission. Then, in the relaying phase, the MR transmits to destination CH the packets that were requested and were already stored in the queuing buffers. Figure 15 shows an illustration of the aforementioned relaying phase and is the uplink from the destination CH perspective.

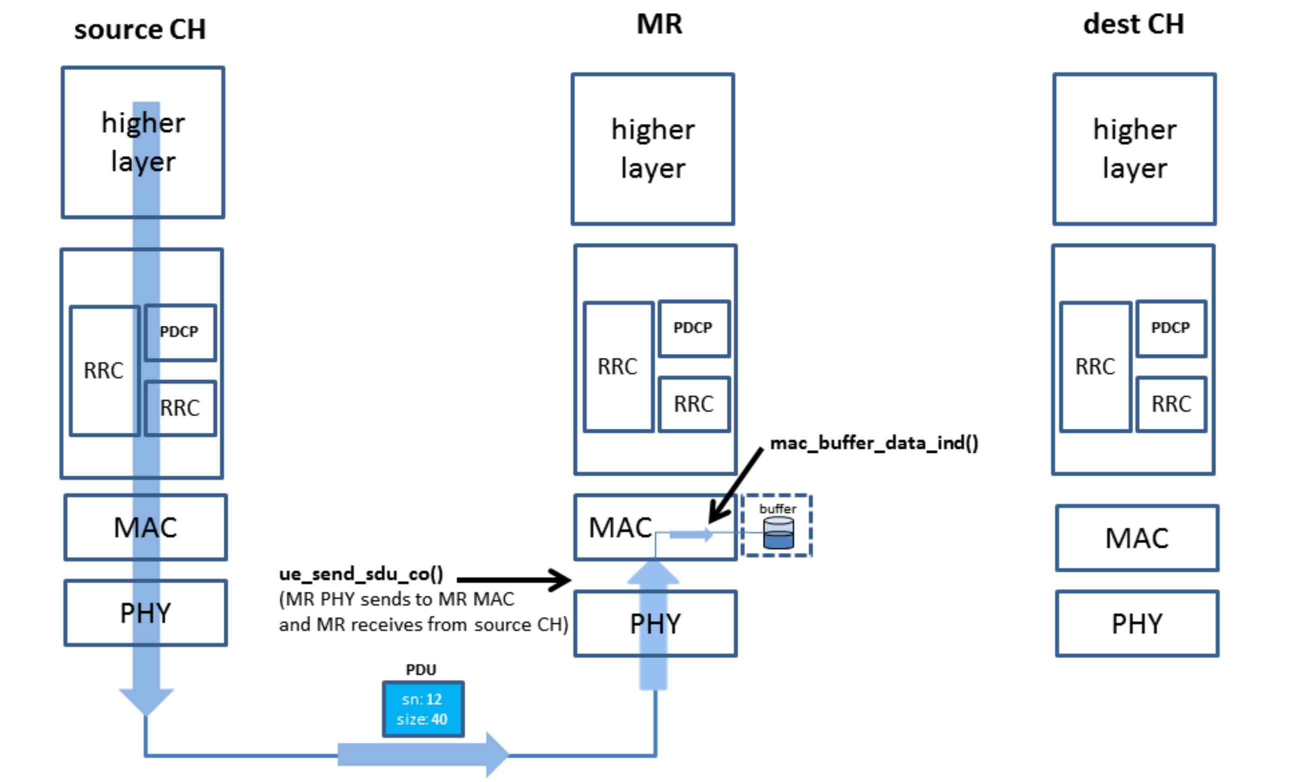


Figure 14. Communication from source CH to MR. Collaborative packets are prevented from entering upwards the higher layers of MR protocol stack and are stored in the MAC buffer.

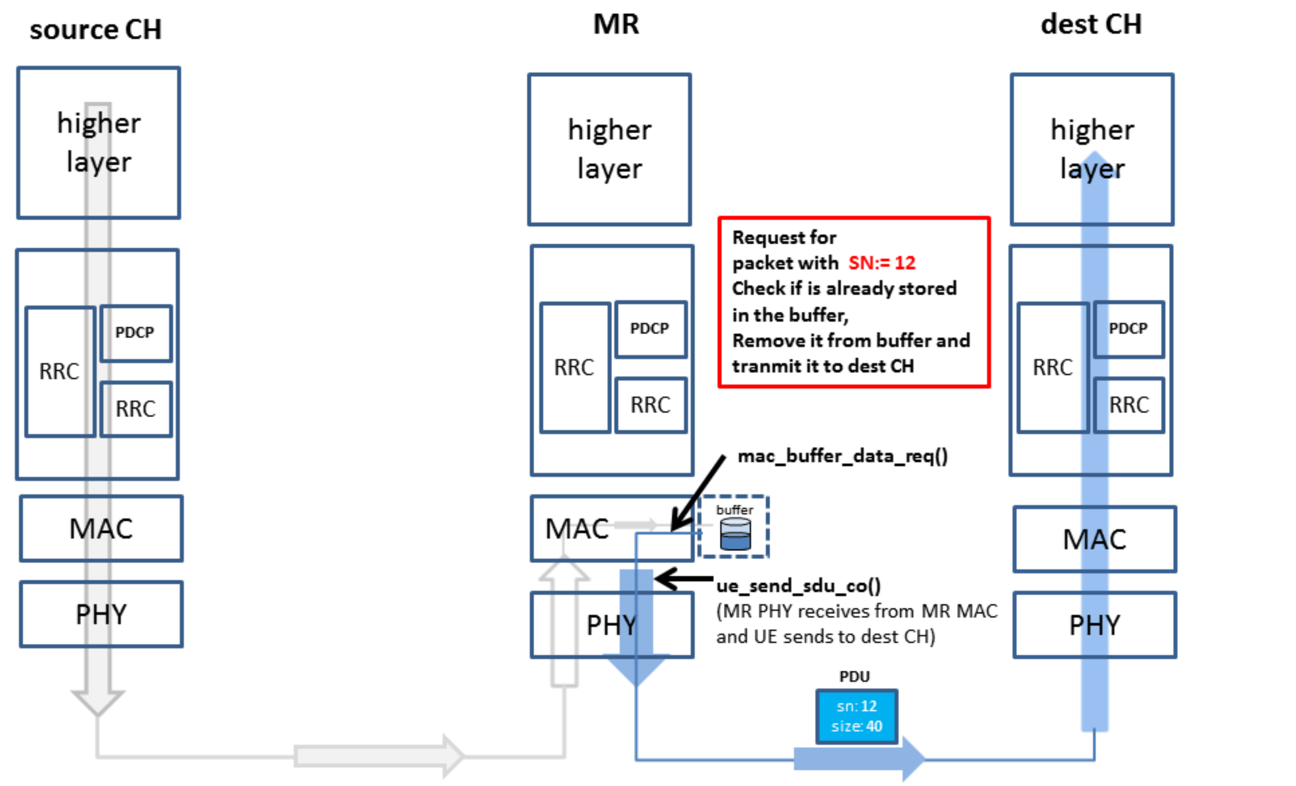


Figure 15. Collaborative packets that are waiting for transmission in the MR buffer are transmitted to the destination CH.

2.4.5.3. PDU requirements

N/A

2.4.6. Controller

2.4.6.1. Input requirements

N/A

2.4.6.2. Functional requirements

The MAC controller shall maintain, for a given CO-RNTI, a table associating a virtual link HARQ transmit process ID to the corresponding MAC PDU SN.

An example of this table is illustrated in the table below.

CO-RNTI #0		
	HARQ process #5	SN#12540
	HARQ process #6	SN#12541
CO-RNTI #1		
	HARQ process #0	SN#5421

	HARQ process #1	SN#5422

Table 5: Table with the association between HARQ process ID and SN number of the MAC PDU which is currently dealt with by the HARQ process

Downlink at the CH side

Upon the scheduling of cooperative data in a DL subframe with a TX broadcast phase, the scheduling function of the MAC controller, based on the virtual DRB RLC buffers and the channel quality indicators, shall send to the concatenation function the Transport Block Size to be used for this transmission.

Uplink at the MR side

When cooperative MAC PDUs are queued in the queuing function, a modified BSR must be sent (called also CO-BSR in the following).

If there is no PMRSCH allocated (with the RNTI of the bridging MR), over which to send a CO-BSR, then the scheduling function of the MAC controller of a MR sends a SR:

- The SR is sent over its PMRCCH (analogous to the PUCCH) located by its specific code.
- The MR sends the SR with the standard PUCCH format 1, according to the standard. The presence of the signal indicates the presence of SR.
- The SR is sent following the periodicity of the PMRCCH (PUCCH), according to the standard.

When the destination CH receives the SR of a bridging MR, it cannot distinguish whether it is for the virtual link or a standard communication, hence, it schedules *separate* resources for all the bridging MRs having sent the SR, on their PMRSCHs (i.e. through DCI with their RNTI). These resources will enable the MRs to send their CO-BSR. Notice that it is only upon reception of a CO-BSR that the destination CH understands that there is a packet to be sent on the virtual link. For a better clarification, refer to Figure 16.

Upon reception of a new DCI with CO-RNTI, which schedules a MAC PDU with a given SN, the MAC layer controller shall:

- Ask the forwarding function if the received SN is queued for the received CO-RNTI
    - If the requested MAC PDU is queued, the forwarding function shall send it to the HARQ function
    - If the requested MAC PDU is not queued, the MAC controller shall instantiate a virtual link HARQ transmit process. The MR and this HARQ

process will keep track of the evolution of the HARQ process by listening to the PHICH, even if it has not the MAC PDU with the right SN.

Uplink at the CH side

The scheduling function of the MAC controller of a CH granting UL resources for a cooperative transmission shall grant also additional UL resources for each MR (if UL resources were not already planned). In those additional UL resources for each MR of the cooperative transmission, each MR shall send their cooperative BSR in case there are further data to be sent in the virtual link. The size of the granted UL resources for each MR shall be enough to send a cooperative BSR or a classical BSR.

The MAC controller of the destination CH shall grant resources for the data of a virtual link only by using a DCI with the CO-RNTI of the virtual link (see PDU requirements for the definition of the new DCI).

Synchronization at the MRs forwarding queues is obtained by explicitly scheduling the sequence numbers in the DCI. It is up to the destination CH to decide which MAC PDUs to schedule, based on the BSR information and CQI / QoS information.

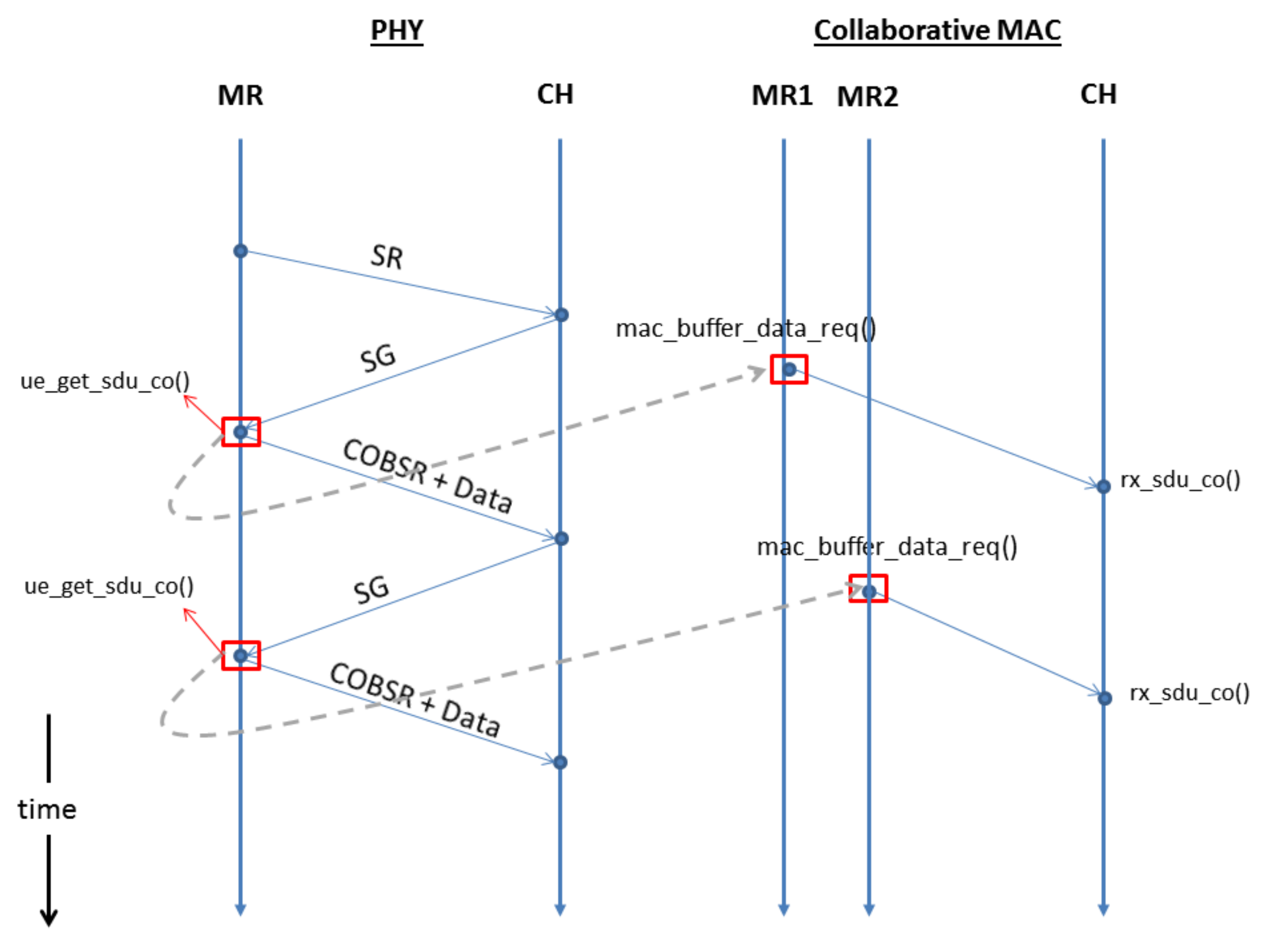


Figure 16. Collaborative MAC forwarding.

2.4.6.3. PDU requirements

The MAC payload shall integrate a new Control Element. This Control Element is called Cooperative Buffer Status Report (CO-BSR) and is composed of:

- CO-RNTI reflecting the virtual link
- One or a list of MAC PDUs Ids and the associated size to be transmitted.

Two types of CO-BSR Control Element are considered: short and LONG. The short CO-BSR is illustrated in the figure below. The long BSR sends the SN and size for four elements.

0	1	2	3	4	5	6	7
CO-RNTI							
CO-RNTI							
SN							
SN							
Buffer size				Padding			

Figure 17 – CO-BSR Control Element message format

This new CO-BSR must be signalled in the MAC header. A new value of the LCID field of MAC header for UL-SCH must be inserted in Table 6.2.1-2 in the LTE MAC description [5]. For instance, we can say that the value 11010, which is currently reserved, denotes a CO-BSR MAC CE. It is through the value of this field that the destination CH understands that there is a request for data over a virtual link.

The MAC controller shall generate a new DCI messages which is an extension of the DCI format 0 [6] called format DCI format 0A including the SN of the MAC PDU to be transmitted.

The DCI format 0A message is composed of the fields that follow:

- For non-hopping PUSCH:
  - $\left(\left\lceil \log_2(N_{\mathrm{RB}}^{\mathrm{UL}}(N_{\mathrm{RB}}^{\mathrm{UL}}+1)/2)\right\rceil\right)$ bits provide the resource allocation in the UL subframe
- Modulation and coding scheme and redundancy version – 5 bits
- New data indicator – 1 bit
- TPC command for scheduled PUSCH – 2 bits
- Cyclic shift for DM RS – 3 bits
- UL index – 2 bits (this field is present only for TDD operation with uplink-downlink configuration 0)
- Downlink Assignment Index (DAI) – 2 bits
- CQI request – 1 bit
- MAC PDU SN – 9 bits (this value can be shortened during the development phase if a stricter range is sufficient).

Detection of the format 0A is identical to the standard PDCCH detection procedure, namely blind detection based on hypothesized length of 32 + $\left\lceil \log_2(N_{\mathrm{RB}}^{\mathrm{UL}}(N_{\mathrm{RB}}^{\mathrm{UL}}+1)/2)\right\rceil$ bits and by CRC mask of CO-RNTI.

2.4.7. HARQ (updates)

2.4.7.1. Input requirements

N/A

2.4.7.2. Functional requirements

Downlink (broadcast phase data transmission)

HARQ behavior for a virtual link in the broadcast phase, as far as data transmission is concerned, is the same as the one specified in the standard.

Downlink (broadcast phase acknowledgments)

The HARQ process in the CH and in MRs follows the standard timing in TDD.

For each virtual link in the source cluster, i.e. for each CO-RNTI, all the bridging MRs in the virtual link always send their ACK/NACK over a common PMRCCH (analogous to PUCCH), by using a PUCCH format 1, the one for the scheduling request (SR), in the following way:

- The presence of the signal is a NACK.
- The absence of the signal is an ACK.

The previous message is call implicit ACK/NACK.

In this way, the CH cannot know which MR is sending an ACK, but will continue to send retransmissions until all the MRs in the virtual link send an implicit ACK, thus implementing the HARQ strategy proposed in WP4 and in the LOLA solution package (see LOLA D4.3 final release).

PUCCH format 1/1a/1b allows multiplexing from 8 to 24 users in case of extended CP and from 12 to 36 users in case of normal CP (see pag. 361 in [3]).

For each virtual link in the source cluster, i.e. for each CO-RNTI, a PUCCH resource index (nRSPUCCH, noc) will be attributed (in the PMRCCH), so that the CH knows which couple of cyclic time-shift domain and orthogonal time spreading code to use in order to detect the implicit ACK/NACK.

The PUCCH region is determined as in the standard. The explicit ACK/NACK message is possibly sent over the same resources, i.e. following the ACK/NACK synchronous process for stopping the source CH sending useless retransmissions.

The explicit ACK/NACK must be used by the MRs which receive from the second CH (the destination one) an ACK of the correct reception of the MAC PDU in the relaying phase. The MR shall send an explicit ACK/NACK to the source CH over the same resources allocated for the implicit ACK/NACK. The explicit ACK/NACK has higher priority than the implicit ACK/NACK.

For each virtual link in the source cluster, i.e. for each CO-RNTI, a PUCCH resource index (nRSPUCCH, noc,ex) will be attributed (in the PMRCCH), so that the CH knows which couple of cyclic time-shift domain and orthogonal time spreading code to use in order to detect the explicit ACK/NACK. This code is different from the one of the implicit ACK/NACK.

All the MRs belonging to the same virtual link use the same code for the explicit ACK/NACK.

Notice that the implicit and explicit ACK/NACK were implemented in the link level simulator, but in an abstracted way.

Uplink (cooperative relaying phase data transmission)

A virtual link HARQ transmit process is initiated by the MAC controller upon the reception of a valid scheduling grant from the destination CH. This HARQ process will be running even if the scheduled MAC PDU is not present in the MR buffer. In this case (absence of the scheduled MAC PDU in the buffer) no data shall be sent to the PHY layer and no signal must be generated on the air by the PHY, including reference symbols or reference signals.

Each HARQ transmit process related to a virtual link must be associated to the MAC PDU sequence number it is serving[1].

A virtual link HARQ transmit process shall keep track of the evolution of the process (e.g. current HARQ round) by monitoring PHICH despite not sending data.

Upon reception of data from the forwarding function on a virtual link, the HARQ transmit process shall send data on the next scheduled opportunity with the right redundancy version.

Uplink (cooperative relaying phase acknowledgments)

PHICH for the cooperative relaying phase is done substantially as in the standard Rel 8. We recall that in Rel 8:

- Multiple PHICHs (i.e. a PHICH group) are mapped to the same set of resource elements.
- Different PHICHs inside the same group are differentiated by use of complex orthogonal Walsh sequences.
- Each PHICH is uniquely identified by a *PHICH index* which is given by the couple (PHICH group, Walsh sequences): $(n_{PHICH}^{group}, n_{PHICH}^{seq})$.
- The number of PHICHs in one group (i.e. the number of UE that can be multiplexed in a PHICH group) is twice the Walsh sequence length. In total we can multiplex 8 users for normal CP and 4 users for extended CP.

[1] This association is necessary to avoid ambiguities when a MR decodes the MAC PDU later than the other MRs. In this way, the MR is able to associate it to the right HARQ process.

PHICH for a virtual link is allocated as in the standard, i.e. in an implicit way, described in the following. The PHICH index is associated to the lowest UL RB used for the corresponding PMRSCH (i.e. PUSCH transmission), where the correspondence is defined by the timing of the ACK/NACK synchronous process. Hence, the PHICH index will be determined by the association described in the standard based on the position of the RB of the corresponding CO-PMRSCH (i.e. the PMRSCH scheduled with the CO-RNTI). The procedure of the assignment of the PHICH index is explained in section 9.1.2 in [4].

2.4.7.3. PDU requirements

The PUCCH resource (code) depends on the position of the CCE in the PDCCH. There is no change w.r.t. the standard mechanism.

2.4.8. Inter-cluster CQI reporting

This section is substantially unmodified; hence the interested reader is invited to check Sect 4.3 in D5.6 v1.1. Moreover, this feature has not been implemented in testbed 3.

3. INTEGRATION AND VALIDATION RESULTS OF LTE MESH EXTENSION

3.1. Link-level simulator

3.1.1. Interference-cancelling receiver for PDCCH

An interference cancelling receiver for the PDCCH, as described in Sec. 2.3.6.2, has been simulated and verified in a link-level simulator. Assume a transmitted desired symbol s_0 and a transmitted interfering symbol s_1, where both transmitters have one antenna. Assume further flat-fading Rayleigh channels and a two-antenna receiver and denote the received signal **y**. Thus, **y** can be written $\mathbf{y} = \mathbf{h}_0 s_0 + \mathbf{h}_1 s_1 + \mathbf{n}$, where $\mathbf{h}_0$ and $\mathbf{h}_1$ are the channel vectors from the transmitters to the receiver, and **n** is AWGN. A distance metric can be defined as

$$D(\mathbf{y}|\mathbf{Hs}) = \left\|\mathbf{y} - \bar{\mathbf{h}}_0 s_0 - \bar{\mathbf{h}}_1 s_1\right\|^2$$

where $\bar{\mathbf{h}}_0$ and $\bar{\mathbf{h}}_1$ are the estimates of $\mathbf{h}_0$ and $\mathbf{h}_1$, respectively. Using this distance metric, the interference cancelling receiver computes the optimal desired and interfering symbols as

$$s_0, s_1 = \arg\min_{s_0, s_1}(D(\mathbf{y}|\mathbf{Hs}))$$

To achieve good performance, $\bar{\mathbf{h}}_0$ and $\bar{\mathbf{h}}_1$ must be good estimates of the channels. The simulations in the next section show that the data must be punctured in the OFDM symbols used for pilots for the interference cancelling receiver to work reliably.

3.1.2. Results

The simulation setup consists of a cluster head CH1 transmitting a desired signal, an

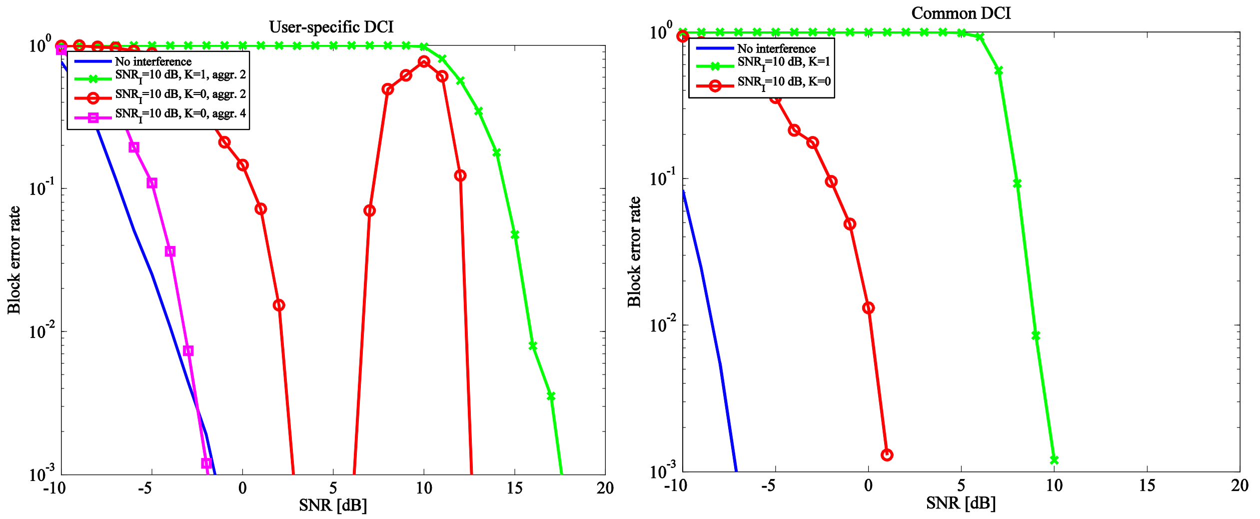


Figure 18 – PDCCH decoding performance

interfering cluster head CH2, and a receiving MR. The setup has been simulated over an AWGN channel for two interference cases, with the SNR of the CH2-MR link at 10 dB and

0 dB, respectively. In the simulations, K is a scaling factor for the data resource elements in the symbols occupied by pilots (the first symbol). The performance is shown separately for common and user-specific DCIs, as the code rates are different.
The figures below show the PDCCH decoding performance with an interfering signal of 10 dB. As can be seen, when data is transmitted in the pilot symbols (K=1) the quality of the channel estimations is deteriorated, and the desired signal cannot be decoded when it is weaker than the interfering signal. As seen in the user-specific DCI plot, an aggregation level of two does not offer sufficient protection when the SNR of the desired signal is close to the SNR of the interfering signal. Increasing the aggregation level to four, a performance similar to the interference free case is achieved.

Similar results can be seen for the case with an interfering signal with an SNR of 0 dB shown below. The performance of the interference cancelling receiver is significantly weakened due to the channel estimations of the interfering signal being affected by noise. However, the desired signal is still decodable at levels several dB weaker than the interfering signal.

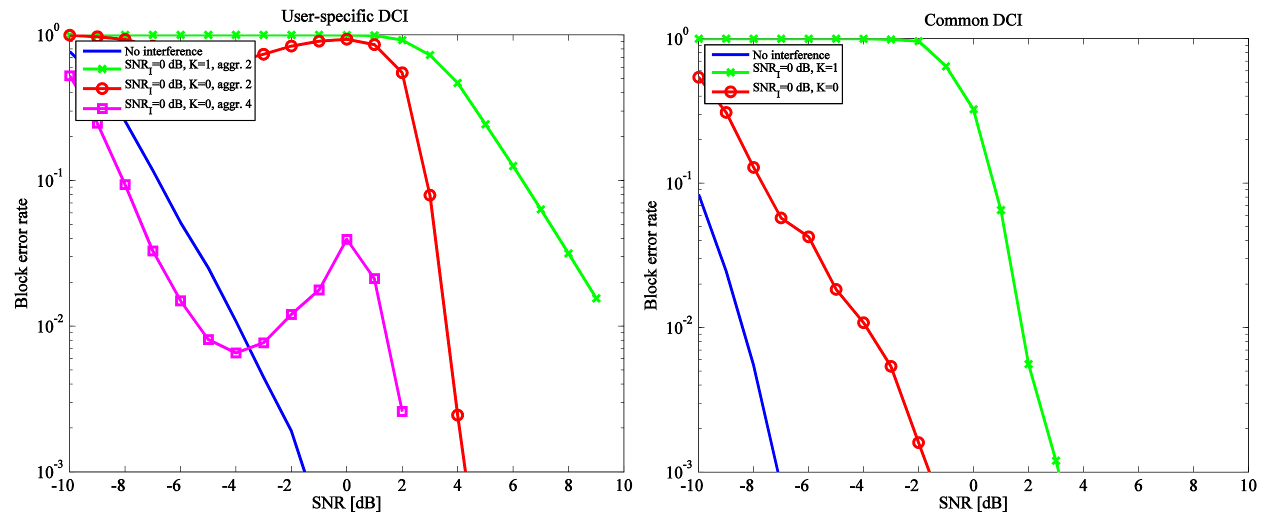


Figure 19 – PDCCH decoding performance with interference

3.2. In-lab system validation platform

3.2.1. WP4 MAC Layer Techniques

3.2.1.1. Virtual Links and MAC Layer Forwarding

Our design in packet level cooperation suggests a new type of virtual links that can be decomposed into classic point-to-point links. A virtual link between two CHs is composed of one or more collaborating MRs that belong to both CHs. Each virtual link is composed of two phases:

- the broadcast phase (downlink from the perspective of the source CH). Here the source CH initiates a collaborative scheduling transmission to the MRs that compose the Vlink.
- the relaying or forwarding phase (uplink from the perspective of destination CH). Here the destination CH schedules a collaborative packet reception from the MRs that compose the Vlink.

In LOLA, we envision a way to enable a multiple access scheme at the higher layer so that to perceive the lower PHY level of the protocol stack still as a packet erasure link even though it may be decomposed into several point to point links. Therefore, we elaborate a mechanism to support MAC layer packet forwarding so that each MR participating in a collaborative transmission could be able to transmit a packet at the MAC layer as soon as the transmitted signal originating from the source CH is received by the MAC layer of the forwarding MR. The same mechanism identifies if a collaborative transmission is on to be processed, and prevents received packets on each MR entering upwards the OSI layers after the MAC. The packets, instead, are queued in buffers maintained on the MAC layer of each MR and wait for transmission when a collaborative packet request is scheduled by the destination CH.

3.2.1.1.1. Collaborative Radio Network Temporary Identifier - CO-RNTI

An RNTI is a term that stands for Radio Network Temporary Identifier and it is a kind of an identification number. In LTE there are many types of RNTIs used for example to identify paging, system information, cell or random access procedures. We introduce the concept of the COllaborative-RNTI as an identification number to differentiate a regular transmission from a collaborative one.

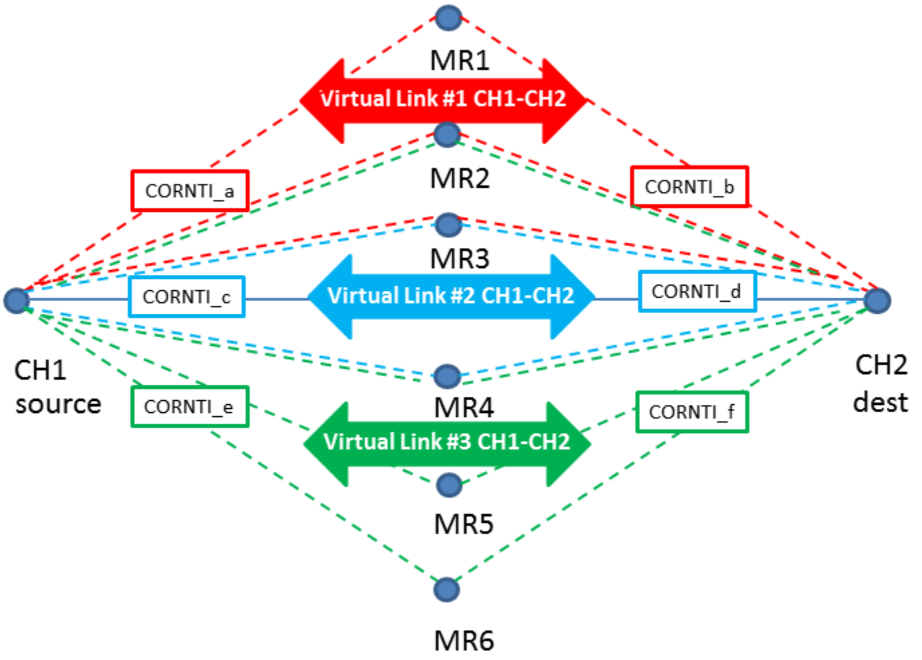
Specifically, this RNTI type, is used for indicating that a certain packet is on a collaborative transmission via a virtual link. The CO-RNTI is carried as a part of the MAC header in the Control Packets that are transmitted from CH to MR in order to establish the VLink.

A collaborative transmission over a virtual link needs at least one MR acting as packet forwarder and two CO-RNTIs that describe the point-to-point transmission on the (CH-MR) physical link. Two CO-RNTIs (an ingress and an egress) can participate to form a Virtual Link setup. The ingress CO-RNTI is used by the destination CH to identify the appropriate buffer - queue in the respective MR so that to be able to schedule the packets stored (which were previously indicated to the destination CH by the same MR). From the perspective of the destination CH that needs to communicate back to the source CH over a collaborative transmission, this design is symmetric and another CO-RNTI is used that describes the transmission being done from the destination CH in this point-to-point link towards source CH via a MR. That CO-RNTI refers to the egress CO-RNTI. Therefore, the least number of CO-RNTIs at a given MR that is essential to compose a bidirectional Virtual link is two.

3.2.1.1.2. MAC - Virtual Link

A virtual link provides an abstraction to the cooperative transmission at the MAC layer. It is used as a means of hiding the information to higher layers: that is, a virtual link between two points is composed of several point-to-point links formed with the aid of relaying MRs. A MR can participate at the same time to multiple VLinks. The MAC layer is responsible for managing the virtual/logical links. More specifically the MAC layer is responsible for identifying the links that will be created in order to complete a single packet transmission between two specific endpoints. Moreover, it is responsible for the identification and scheduling of collaborative transmissions both in downlink and uplink direction.

In Figure 20, virtual links between two CHs are illustrated. Several MRs act as forwarders to assist in a collaborative packet transmission. We use the same color to show the CO-RNTIs that belong on the same Virtual Link.



Virtual Link	CORNTI ingress	CORNTI egress
Vlink #1 (CH1-CH2)	a	b
Vlink #2 (CH1-CH2)	c	d
Vlink #3 (CH1-CH2)	e	f

Figure 20. A virtual link is composed with the aid of several MR acting as forwarders. Each MR is associated with two CO-RNTIs, one for ingress and one for egress communication direction.

A virtual link definition is given in Table 7, as it is defined in OpenAirInterface software. A virtual link structure is composed by a vLinkID that is used as an identifier for the link, by a Cell Id for the source and one for the destination CH, and an array of the participating MRs in that Virtual Link.

3.2.1.1.3. Forwarding Table

An example of a forwarding table has already been illustrated in Figure 20 presenting Virtual Links between two CHs. In a larger LTE network, each CH is aware of the forwarding tables and Virtual Links in the neighboring CH, and for the respective CO-RNTIs and MRs that participate in each virtual link. An example of forwarding table is given in the table below.

Virtual Link Id	CORNTI (ingress)	CORNTI (egress)
VLink #1	CORNTI #a	CORNTI #b

VLink #2	CORNTI #e	CORNTI #f
Vlink #...	...	...

Table 6. A forwarding table example.

In Table 7, we give the definition of a forwarding table as it is used in OpenAirInterface software. A forwarding_Table structure is composed by an array of entries where each entry is consisted of a pair of the participating CHs (eNBs), the ingress and the egress CORNTIs, and the virtual link ID.

```
//openair2/LAYER2/MAC/virtual_link.h
// MAC structure
/* virtual link and collaborative RNTI definition */
struct mr_array {
        int count; /* Nb of MR currently in the list */
        unsigned char array[MAX_MR_PER_VLINK]; /* List of UE_index */
};

typedef struct  {
  unsigned char vlinkID;
  unsigned short PCellIdsourceCH;
  unsigned short PCellIddestCH;
  struct mr_array MRarray;
  unsigned char status;

  /* For now M-TMSI replaced by UE_indexes */
  //unsigned int MRMTMSIlist[MAX_MR_PER_VLINK];
}virtual_link;

typedef struct  {
        int count; //the number of virtual link in the table
        virtual_link array[MAX_VLINK_PER_CH];
}virtual_links;

// CORNTIs array for UE/eNB MAC/PHY structures
typedef struct  {
  u8 count;
```

```
  // Collaborative data sequence number;
  u16 sn[MAX_VLINK_PER_CH];
  u16 array[MAX_VLINK_PER_CH];
}cornti_array;


/* Forwarding table definition */
#define MAX_FW_ENTRY 10

typedef struct  {
  u8  vlid;
  u16 eNB1;
  u16 eNB2;
  u16 cornti1;
  u16 cornti2;
}forwardingTableEntry;

typedef struct  {
  u8 count;
  forwardingTableEntry array[MAX_FW_ENTRY];
}forwarding_Table;
```

Table 7. Virtual Link, CO-RNTI and forwarding Table definition

3.2.1.1.4. Virtual Link Setup and MAC forwarding buffers initiation

The virtual link setup is initiated during RRC connection establishment phase between the CHs and corresponding MRs. Each CH forwards his neighbourhood at 2 hops and, thanks to this, other CHs understand the presence of bridging MRs, and then build Vlinks. This procedure is initiated by the CHs in order to shorten the number of IP network-layer hops for routes going through multiple CHs by exploiting the forwarding in the MAC layer.

During the RRCConectionReconfiguration exchange message that a CH sends to a MR, the first phase of the establishment of a virtual link is initiated. The ingress CO-RNTI is established after the successful receipt of the RRCConectionReconfigurationComplete message by the source CH.

Then the same procedure is initiated for the destination CH with the corresponding MR for establishing the egress CO-RNTI. After the successful reception of RRCConectionReconfigurationComplete message by the destination CH from MR, a virtual link is established between the two CHs. Now the virtual link is composed by two CHs, one MR and the corresponding ingress and egress CORNTIs. For the other MRs that might participate in the virtual link, the forwarding table is being updated with the respective CORNTIS after the successful exchange and message negotiation among the MRs and the source and destination CH.

At the same time, as the exchange of RRC Connection and configuration message is being processed, the initiation of the MAC forwarding queue for the particular CORNTI takes place. We will clarify those details in the next subsection. However, we note that for each CORNTI being established, we maintain a respective buffer for incoming packets on the MAC layer of the respective MR where they are queued and stored.

A representative illustration of two virtual links setup is depicted on Figure 21 - Figure 24. There are two triangle networks forming two virtual links between CH1-CH2 and CH1-CH3. A forwarding MR is commonly used by both virtual links. Each figure illustrates the respective CORNTI establishment and the MAC buffer initiation between each CH and MR pair during the RRC connection and reconfiguration message exchange in the point-to-point link. At the end, there are two virtual links established, each one composed of one ingress and one egress CORNTI. Therefore, since we maintain one buffer for each established CORNTI, four buffers are initiated for packet storage and queuing on the MR's MAC layer.

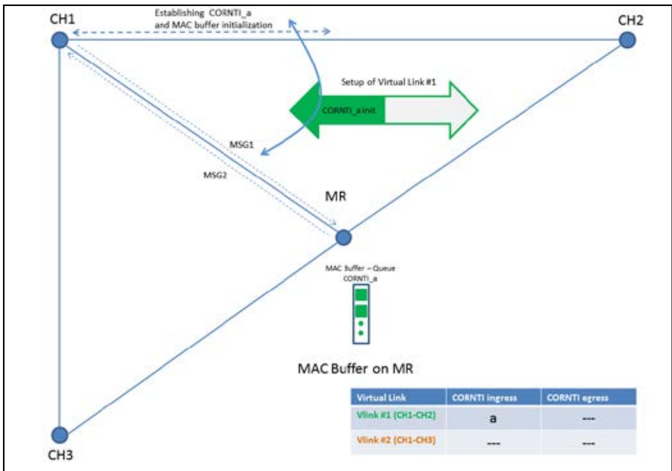


Figure 21. Establishing CORNTI a and MAC buffer queue initiation in MR for VLink #1

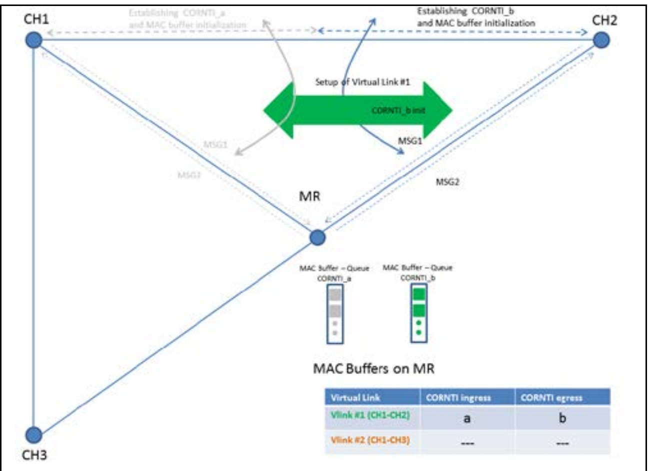


Figure 22. Establishing CORNTI b and MAC buffer queue initiation in MR for VLink #1

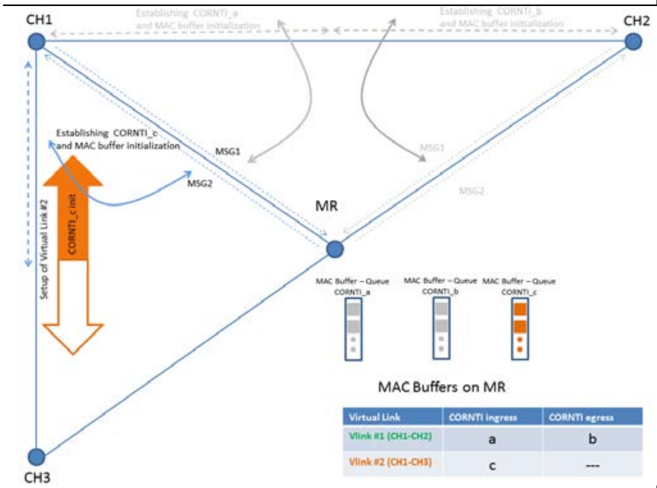


Figure 23. Establishing CORNTI c and MAC buffer queue initiation in MR for VLink #2

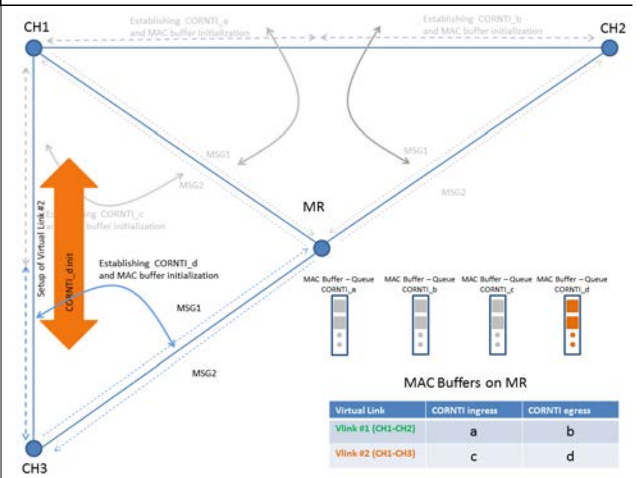


Figure 24. Establishing CORNTI d and MAC buffer queue initiation in MR for VLink #2

3.2.1.1.5. The MAC forwarding Buffer queue

The MAC forwarding queue is initiated per CO-RNTI on each participating MR in a Virtual Link. A simple forwarding buffer can be seen to perform a FIFO process, however selecting this approach can lead to severe latency in the case where the destination CH needs to schedule certain packets being identified by their sequence number (SN) or by a certain MAC PDU_SIZE. In the case where each buffer maintains a simple FIFO buffer, for every operation that would require a search of this packet whether to identify if it is already stored, the extensive sequential search over all the elements would be unavoidable. In the case where we need to schedule one single packet the time required to complete a search for finding whether is stored or not would be O(n) and in the case where we need to schedule a bunch of packets of a certain size, then we would require at least polynomial time to find a solution to the well-known knapsack problem.

In order to alleviate the aforementioned obstacles, we enhanced the MAC queue with a sophistication storage capability that can provide O(logn) access time in search, delete and insert operations. For that reason we used self-balancing data structures which called AVL trees (AVL are the initials of the names of its inventors Adelson, Velski, Landis) that provide to the MAC buffer assistance and good performance in search, and insert and delete operations.

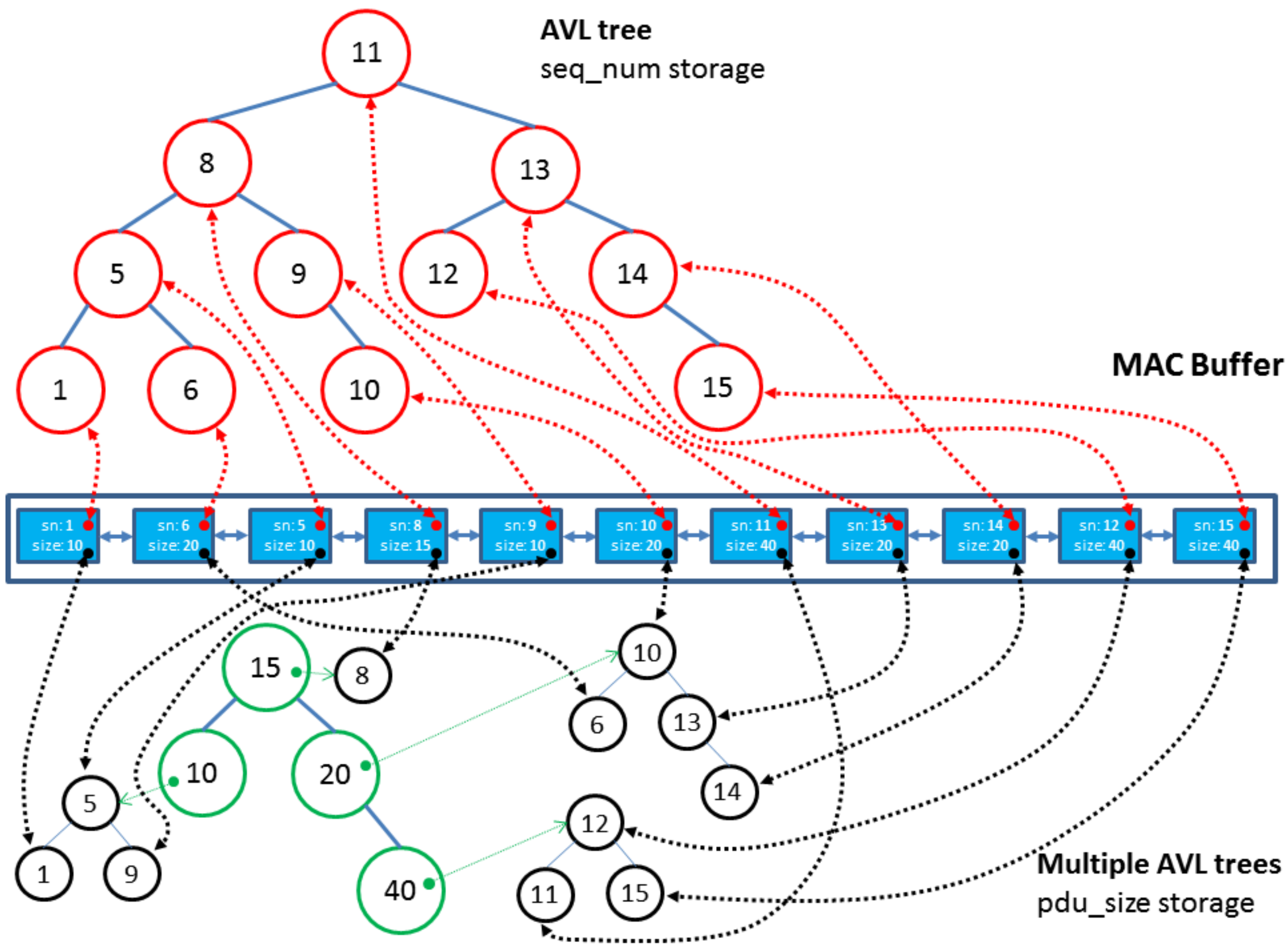


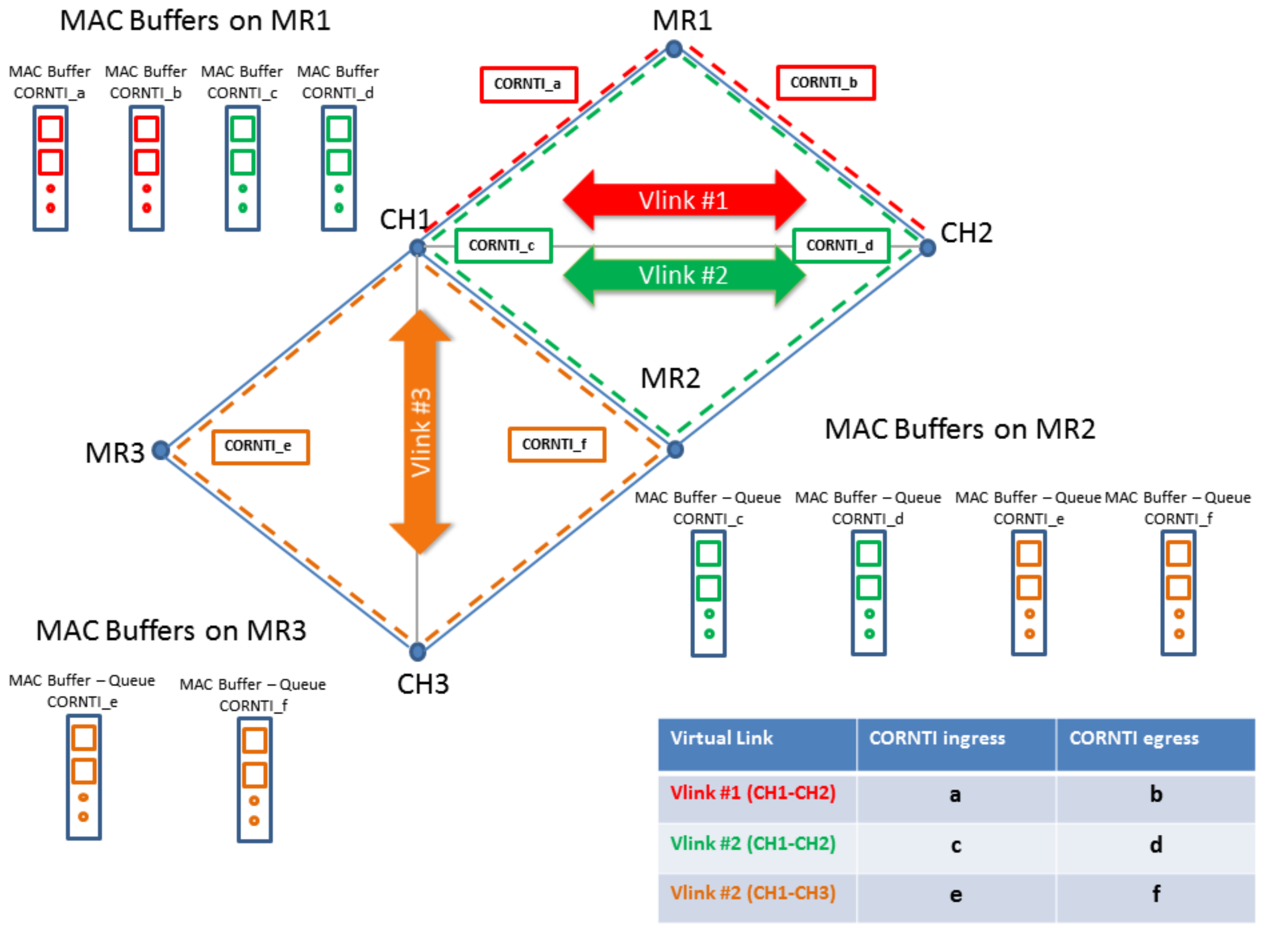
Figure 25. MAC Buffer. Each PDU is stored in the queue. AVL trees for supporting logarithmic time in storage, search and deletion are also maintained.

In Figure 25, an illustration of a MAC buffer instance is given. It depicts PDUs stored in a buffer (double-link-list). Each PDU maintains two pointers indicating nodes in AVL trees where information about PDUs sequence number and PDU size are kept. For example the tree with the red nodes shows an AVL tree that is self-created and balanced based on the PDUs sequence number. Whenever a PDU is inserted (or deleted) in the MAC buffer, an insertion (or deletion) happens in the corresponding trees. Each sequence number is a unique identifier and thus we need to maintain only one tree for the sequence number storage.

The AVL tree with the green nodes is a more complex structure. It stores elements relying on the PDUs size. However, PDU size is not a unique number among PDUs and therefore we cannot maintain a sole AVL tree for indicating the correct PDU. For that reason, each node in the green AVL tree has a pointer indicating an AVL tree that maintains PDUs sequence numbers from PDUs with the same PDU size (black AVL tree). Next we describe the three supported operations, ***search, insert, delete*** in the MAC buffer.

- **Search,** It supports functionality to search for a specific PDU based on the PDUs sequence number, or to search for a PDU with a size less than or equal to a given value. The above operation is achieved in O(logn) time due to the existence of the AVL trees.
- **Insert,** Suppose that a new PDU is on to be inserted on the MAC buffer. Before this operation is allowed, a first check must be performed whether this PDU is already stored or not. This can be done through a search in the corresponding AVL tree for the specific PDU sequence number. If this PDU is already stored, the packet is discarded, otherwise it enters the buffer and it is attached on tail. At the same time, element instances-nodes in the corresponding AVL trees for the sequence number and the PDU packet size are created and the self-balancing operations of the AVL trees are triggered. At the end, each PDU is stored in the MAC buffer maintaining also two pointers: the first one indicating the appropriate node in the PDU's sequence number AVL tree, and the other one the respective AVL tree for the specific PDU size.
- **Delete,** As in the insert operation where a check for the PDU is needed to be performed before the operation execution, the same check is essential for the delete operation. For a PDU with a certain PDU sequence number, we search in the corresponding AVL tree to track if it is stored or not. If not, delete operation aborts, otherwise the PDU is deleted from the buffer along with the respective nodes in the AVL trees. After that, self-balancing operations on the AVL trees are automatically triggered.

All of the above describe the internal functionality of the buffer in each MR maintained on the MAC layer. However, what is essential is to provide an API for the MAC layer by using specific primitives. The corresponding primitives are given in Table 3 and have been developed on OAI implementation.



Virtual Link	CORNTI ingress	CORNTI egress
Vlink #1 (CH1-CH2)	a	b
Vlink #2 (CH1-CH2)	c	d
Vlink #2 (CH1-CH3)	e	f

Figure 26. Vlinks and CORNTIs in a two diamond topology. Respective Buffers maintained in MAC level of each participating MR are also shown.

3.2.1.2. Scheduling with bounded delay and back-pressure

Resource allocation and scheduling based on optimization criteria can improve the performance efficiency in the collaborative communication. A plain use of queues in the MAC layer of the collaborative MRs may lead in reverse results, and instead of a good performance due to the collaborative communication, the LTE network might experience severe degradation in its performance, if scheduling is not efficiently applied. For that reason, tools from optimization theory can be used in order to acquire rules for indicating optimal transmissions and scheduling towards a certain optimization objective such as maximizing throughput, minimizing delay or minimizing power consumption performance in the LTE network while also ensuring queuing stability.

Queuing instability is a degrading factor that deteriorates devices performance since buffer overflowing has a negative impact in performance metrics for QoS and QoE. The back-pressure algorithm relying on tools from optimization theory introduced the notion of taking advantage of the volatile network dynamics in order to indicate the optimal scheduling decisions that benefit the network operation towards a pre-defined optimization objective while also keeping stability ensured.

In LTE, resources blocks are finite and must be shared among users both in uplink and downlink transmissions. Partitioning the resources blocks among users requires a scheduling policy that ensures that each resource block is assigned to a sole user relying on a metric that couples jointly the queuing congestion and an estimation of the link channel quality between the user and the CH. Moreover, back-pressure based scheduling and resources allocation policies can achieve performance-objective optimization while also attaining a certain tradeoff with networking delay. This is due to the fact that optimizing a specific system-parameter such as power consumption comes at a cost in delay that can be calibrated via a tunable parameter. For example, considering the Lyapunov-drift formula [9][10], which captures the difference in queuing dynamics and the dynamic nature of the channel condition, we are able to formulate an optimization problem whose proposed solution ends in backpressure based policies.
An example is illustrated in Figure 25, where a frame is constructed relying on a coupling of resources block allocation and MAC queue buffers length.

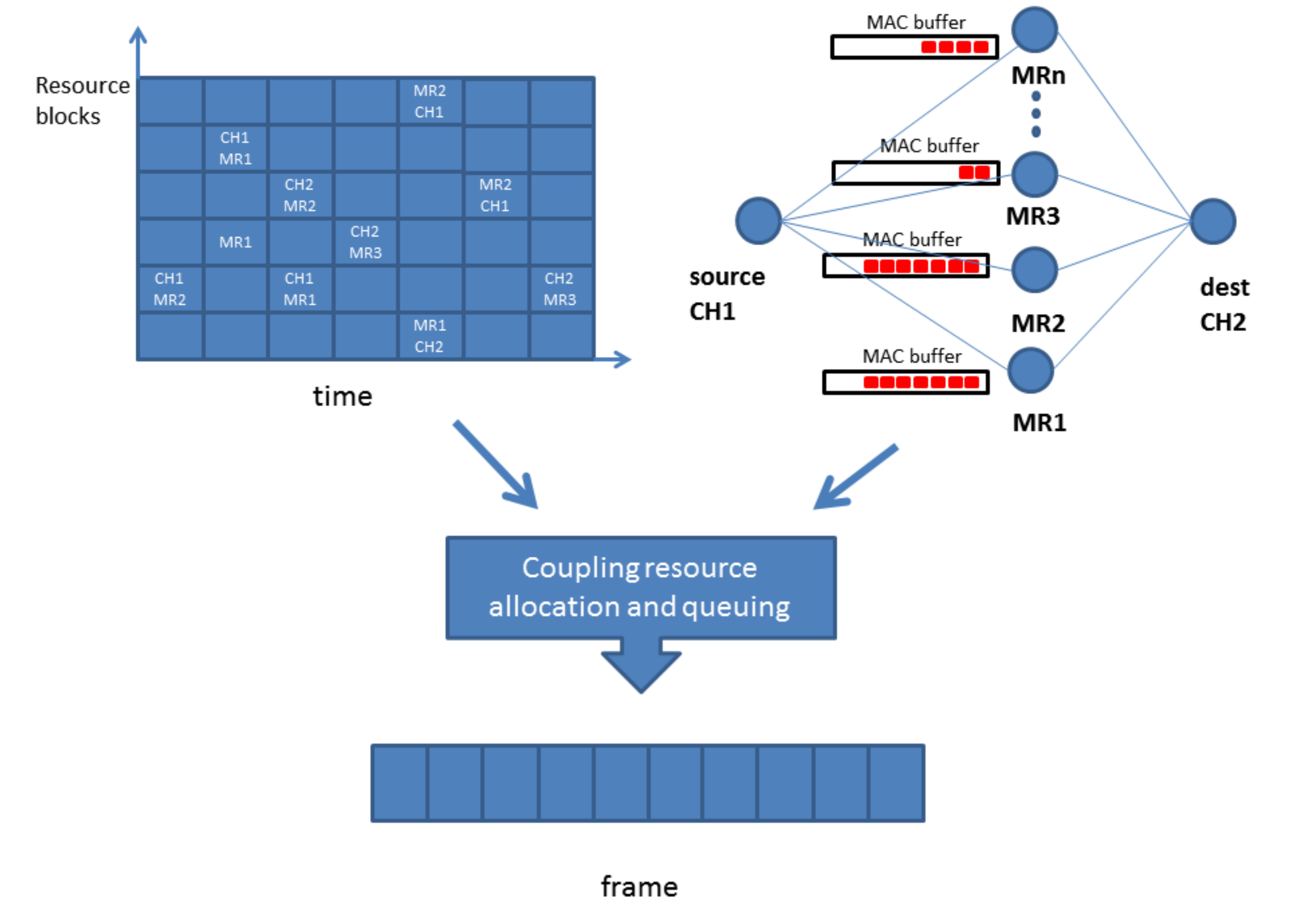


Figure 27. Resource Allocation and Back-pressure based scheduling

Maximum rate in DL was limited to that of the maximum rate in UL so that the same PDU could be sent on the second hop. The packet size was limited to avoid the RLC fragmentation at the source CH in order to obtain the best delay performance (no delay induced by RLC fragmentation and reassembly).

3.2.2. Result

3.2.2.1. RRC-Control Plane

Here we demonstrate the Control Plane procedures which take place on the RRC layer and are used to setup and configure the radio resources in order to create at first point-to-point links among the CHs and the collaborating MRs and then to establish a virtual link. As a first phase we describe analytically the RRC procedures for a simple triangle network (CH0-MR-CH1) and then we present the same procedures for the diamond network (CH0-(MR0-MR1)-CH1).

A simple triangle network where a MR (referred as UE in the log output presented on Table 8 and Table 9) establishes at first a physical point-to-point connection with two

CHs (referred as eNBs in the aforementioned Tables) and then a Virtual Link is set on through RRCConnenctionReconfiguration messaging.

In Table 8, the whole output from the execution in the OAI simulator is given. Command line arguments are selected to setup a virtual link between two CHs and one connected MR. In frame 9, an RRCConnectionRequest is sent over the UL-CCCH channel from MR to CH0 (UE 0). In frame 14, on the downlink logical channel DL-DCCH a RRCConnectionReconfiguration message from CH0 to MR is being sent. During the same frame, MR receives control information from CH0 and configures a Data Radio Bearer with id 0 and a Logical Channel ID 3. (DRB 0/LCID 3). Moreover at the same frame 14 the exchange of the RRCConnectionReconfigurationComplete message in the uplink, is received by the CH0 and finally the connection is established on frame 18.

One frame before the frame 18 where the connection of CH0 and MR was established, in frame 17 the same procedure started for the pair MR-CH1, in order for the relaying point-to-point link to be established. This procedure ends in frame 26. Upon this time both point-to-point links have been initiated (CH0-MR and MR-CH1). Then an opportunity for creating a Virtual Link is identified so that both CHs to be able to bypass the IP Layer of the MR protocol stack and a specific message is sent over the uplink from MR to CH1 for reconfiguring the respective DRB. Then, in Frame 100 both CHs initiate the collaborative RBs by instantiating two CO-RNTIs for the vlid 0. Finally the procedure ends in Frame 104 where both CHs are participating in a Virtual Link with the aid of an intermediate MR. This is verified by the reception of the message RRCConnectionReconfigurationComplete by both CHs at the uplink channel UL-DCCH.

In Table 9, we filter only the **received RRCConnectionReconfiguration** messaging (grep –i "received rrc") where we can observe the acknowledged messages at each communication step towards at first creating point-to-point links and afterwards creating a virtual link.

Table 8. RRC exchange messaging (CH0-MR-CH1 topology)

```
apaposto@ubuntu:~/openair4G_mesh/targets/SIMU/USER$ ./oaisim -a -b2 -H2 -u1 -w 1
-W1 -L 1 -n 200 | grep -i "rrcconnectionre"

[RRC][D][UE] RRCConnectionRequest Encoded 48 bits (6 bytes), ecause 0
[RRC][I][UE 0] : Frame 9, Logical Channel UL-CCCH (SRB0), Generating
RRCConnectionRequest (bytes 6, eNB 0)
[RRC][D][MSC_MSG][FRAME 00010][RRC_UE][MOD 00][][--- MAC_DATA_REQ
(RRCConnectionRequest eNB 0) --->][MAC_UE][MOD 00][]
[MAC][D][UE 0] Frame 10: Requested RRCConnectionRequest, got 6 bytes
c8.da.a2.1f.4.0.0.0.0.[RRC][D][MSC_MSG][FRAME 00011][MAC_eNB][MOD 00][][---
MAC_DATA_IND  (rrcConnectionRequest on SRB0) -->][RRC_eNB][MOD 00][]
            <rrcConnectionReconfiguration>
                        <rrcConnectionReconfiguration-r8>
```

```
                        </rrcConnectionReconfiguration-r8>
            </rrcConnectionReconfiguration>
[RRC][I]RRCConnectionReconfiguration Encoded 154 bits (20 bytes)
[RRC][I][eNB 0] Frame 14, Logical Channel DL-DCCH, Generate
RRCConnectionReconfiguration (bytes 20, UE id 0)
[RLC][D][MSC_MSG][FRAME 00014][RRC_eNB][MOD 00][][--- PDCP_DATA_REQ/20 Bytes
(rrcConnectionReconfiguration to UE 0 MUI 0) --->][PDCP][MOD 00][RB 01]
            <rrcConnectionReconfiguration>
                        <rrcConnectionReconfiguration-r8>
                        </rrcConnectionReconfiguration-r8>
            </rrcConnectionReconfiguration>
[RRC][I][UE 0] Frame 14: Receiving from SRB1 (DL-DCCH), Processing
RRCConnectionReconfiguration (eNB 0)
[RRC][D][UE 0] Frame 14: RRCConnectionReconfiguration Configuring DRB 0/LCID 3
[RRC][D]RRCConnectionReconfigurationComplete Encoded 20 bits (3 bytes)
[RRC][I][UE 0] Frame 14 : Logical Channel UL-DCCH (SRB1), Generating
RRCConnectionReconfigurationComplete (bytes 3, eNB_index 0)
[RLC][D][MSC_MSG][FRAME 00014][RRC_UE][MOD 02][][--- PDCP_DATA_REQ/3 Bytes
(RRCConnectionReconfigurationComplete to eNB 0 MUI 1) --->][PDCP][MOD 02][RB 01]
[RRC][D][UE] RRCConnectionRequest Encoded 48 bits (6 bytes), ecause 0
[RRC][I][UE 0] : Frame 17, Logical Channel UL-CCCH (SRB0), Generating
RRCConnectionRequest (bytes 6, eNB 1)
[RRC][D][MSC_MSG][FRAME 00018][RLC][MOD 00][RB 01][--- RLC_DATA_IND 3 bytes
(RRCConnectionReconfigurationComplete) --->][RRC_eNB][MOD 00][]
[RRC][I][eNB 0][VLINK] Frame  18 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, reconfiguring DRB 0/LCID 3
[RRC][D][MSC_MSG][FRAME 00018][RRC_UE][MOD 00][][--- MAC_DATA_REQ
(RRCConnectionRequest eNB 1) --->][MAC_UE][MOD 00][]
[MAC][D][UE 0] Frame 18: Requested RRCConnectionRequest, got 6 bytes
c8.f8.a2.1f.4.0.0.0.[RRC][D][MSC_MSG][FRAME 00019][MAC_eNB][MOD 01][][---
MAC_DATA_IND  (rrcConnectionRequest on SRB0) -->][RRC_eNB][MOD 01][]
            <rrcConnectionReconfiguration>
                        <rrcConnectionReconfiguration-r8>
                        </rrcConnectionReconfiguration-r8>
            </rrcConnectionReconfiguration>
[RRC][I]RRCConnectionReconfiguration Encoded 154 bits (20 bytes)
[RRC][I][eNB 1] Frame 22, Logical Channel DL-DCCH, Generate
RRCConnectionReconfiguration (bytes 20, UE id 0)
[RLC][D][MSC_MSG][FRAME 00022][RRC_eNB][MOD 01][][--- PDCP_DATA_REQ/20 Bytes
(rrcConnectionReconfiguration to UE 0 MUI 1) --->][PDCP][MOD 01][RB 01]
            <rrcConnectionReconfiguration>
                        <rrcConnectionReconfiguration-r8>
                        </rrcConnectionReconfiguration-r8>
            </rrcConnectionReconfiguration>
[RRC][I][UE 0] Frame 22: Receiving from SRB1 (DL-DCCH), Processing
RRCConnectionReconfiguration (eNB 1)
[RRC][D][UE 0] Frame 22: RRCConnectionReconfiguration Configuring DRB 0/LCID 3
[RRC][D]RRCConnectionReconfigurationComplete Encoded 20 bits (3 bytes)
[RRC][I][UE 0] Frame 22 : Logical Channel UL-DCCH (SRB1), Generating
RRCConnectionReconfigurationComplete (bytes 3, eNB_index 1)
[RLC][D][MSC_MSG][FRAME 00022][RRC_UE][MOD 02][][--- PDCP_DATA_REQ/3 Bytes
(RRCConnectionReconfigurationComplete to eNB 1 MUI 3) --->][PDCP][MOD 02][RB 12]
[RRC][D][MSC_MSG][FRAME 00026][RLC][MOD 01][RB 01][--- RLC_DATA_IND 3 bytes
(RRCConnectionReconfigurationComplete) --->][RRC_eNB][MOD 01][]
[RRC][I][eNB 1][VLINK] Frame  26 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, reconfiguring DRB 0/LCID 3
            <rrcConnectionReconfiguration>
                        <rrcConnectionReconfiguration-r8>
                        </rrcConnectionReconfiguration-r8>
            </rrcConnectionReconfiguration>
[RRC][I]RRCConnectionReconfiguration Encoded 125 bits (16 bytes)
```

```
[RRC][D][eNB 0]Frame 100, Logical Channel DL-DCCH, Generate
RRCConnectionReconfiguration for collaborative RB (bytes 16, UE id 0, cornti
397a, vlid 0)
[RLC][D][MSC_MSG][FRAME 00100][RRC_eNB][MOD 00][--- RLC_DATA_REQ/16 Bytes
(rrcConnectionReconfiguration to UE 0) --->][RLC][MOD 00][RB 01]
            <rrcConnectionReconfiguration>
                        <rrcConnectionReconfiguration-r8>
                        </rrcConnectionReconfiguration-r8>
            </rrcConnectionReconfiguration>
[RRC][I]RRCConnectionReconfiguration Encoded 125 bits (16 bytes)
[RRC][D][eNB 1]Frame 100, Logical Channel DL-DCCH, Generate
RRCConnectionReconfiguration for collaborative RB (bytes 16, UE id 0, cornti
b39c, vlid 0)
[RLC][D][MSC_MSG][FRAME 00100][RRC_eNB][MOD 01][--- RLC_DATA_REQ/16 Bytes
(rrcConnectionReconfiguration to UE 0) --->][RLC][MOD 01][RB 01]
            <rrcConnectionReconfiguration>
                        <rrcConnectionReconfiguration-r8>
                        </rrcConnectionReconfiguration-r8>
            </rrcConnectionReconfiguration>
[RRC][I][UE 0] Frame 100: Receiving from SRB1 (DL-DCCH), Processing
RRCConnectionReconfiguration (eNB 0)
[RRC][D][UE 0][VLINK] Frame 100: RRCConnectionReconfiguration, Reconfiguring
CODRB 0/LCID 179
[RRC][D]RRCConnectionReconfigurationComplete Encoded 20 bits (3 bytes)
[RRC][I][UE 0] Frame 100 : Logical Channel UL-DCCH (SRB1), Generating
RRCConnectionReconfigurationComplete (bytes 3, eNB_index 0)
[RLC][D][MSC_MSG][FRAME 00100][RRC_UE][MOD 02][][--- PDCP_DATA_REQ/3 Bytes
(RRCConnectionReconfigurationComplete to eNB 0 MUI 4) --->][PDCP][MOD 02][RB 01]
            <rrcConnectionReconfiguration>
                        <rrcConnectionReconfiguration-r8>
                        </rrcConnectionReconfiguration-r8>
            </rrcConnectionReconfiguration>
[RRC][I][UE 0] Frame 100: Receiving from SRB1 (DL-DCCH), Processing
RRCConnectionReconfiguration (eNB 1)
[RRC][D][UE 0][VLINK] Frame 100: RRCConnectionReconfiguration, Reconfiguring
CODRB 0/LCID 179
[RRC][D]RRCConnectionReconfigurationComplete Encoded 20 bits (3 bytes)
[RRC][I][UE 0] Frame 100 : Logical Channel UL-DCCH (SRB1), Generating
RRCConnectionReconfigurationComplete (bytes 3, eNB_index 1)
[RLC][D][MSC_MSG][FRAME 00100][RRC_UE][MOD 02][][--- PDCP_DATA_REQ/3 Bytes
(RRCConnectionReconfigurationComplete to eNB 1 MUI 5) --->][PDCP][MOD 02][RB 12]
[RRC][D][MSC_MSG][FRAME 00104][RLC][MOD 00][RB 01][--- RLC_DATA_IND 3 bytes
(RRCConnectionReconfigurationComplete) --->][RRC_eNB][MOD 00][]
[RRC][I][eNB 0] Frame  104 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, DRB 0/LCID 3 already
reconfigured
[RRC][I][eNB 0][VLINK] Frame  104 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, reconfiguring CODRB 0/LCID 179
[RRC][D][MSC_MSG][FRAME 00104][RLC][MOD 01][RB 01][--- RLC_DATA_IND 3 bytes
(RRCConnectionReconfigurationComplete) --->][RRC_eNB][MOD 01][]
[RRC][I][eNB 1] Frame  104 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, DRB 0/LCID 3 already
reconfigured
[RRC][I][eNB 1][VLINK] Frame  104 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, reconfiguring CODRB 0/LCID 179
```

Table 9. Log output for Received RRCConnectionReconfigurationComplete Messaging from collaborating MR (CH0-MR-CH1) topology

```
[RRC][I][eNB 0][VLINK] Frame  18 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, reconfiguring DRB 0/LCID 3
[RRC][I][eNB 1][VLINK] Frame  26 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, reconfiguring DRB 0/LCID 3
[RRC][I][eNB 0] Frame  104 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, DRB 0/LCID 3 already
reconfigured
[RRC][I][eNB 0][VLINK] Frame  104 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, reconfiguring CODRB 0/LCID 179
[RRC][I][eNB 1] Frame  104 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, DRB 0/LCID 3 already
reconfigured
[RRC][I][eNB 1][VLINK] Frame  104 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, reconfiguring CODRB 0/LCID 179
```

In Table 10, the received RRCConnectioReconfiguration messages from the log output of the oaisim execution are given. Notice that the execution arguments are configured to setup two connected MRs between two CHs. Note that, on frame 104 the virtual link between CH0-CH1 is established and it is finally composed of two intermediate MRs.

Table 10. Log output for Received RRCConnectionReconfigurationComplete Messaging from collaborating MRs (Diamond Topology)

```
apaposto@ubuntu:~/openair4G_mesh/targets/SIMU/USER ./oaisim -a -b2 -H2 -u2 -w 1
-W 2 -L 1 -n 200 | grep "received rrcconnectionre"

 [RRC][I][eNB 0][VLINK] Frame  18 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, reconfiguring DRB 0/LCID 3
[RRC][I][eNB 1][VLINK] Frame  26 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, reconfiguring DRB 0/LCID 3
[RRC][I][eNB 0][VLINK] Frame  35 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 1, reconfiguring DRB 0/LCID 14
[RRC][I][eNB 1][VLINK] Frame  57 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 1, reconfiguring DRB 0/LCID 14
[RRC][I][eNB 0] Frame  104 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, DRB 0/LCID 3 already
reconfigured
[RRC][I][eNB 0][VLINK] Frame  104 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, reconfiguring CODRB 0/LCID 179
[RRC][I][eNB 1] Frame  104 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, DRB 0/LCID 3 already
reconfigured
[RRC][I][eNB 1][VLINK] Frame  104 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 0, reconfiguring CODRB 0/LCID 179
[RRC][I][eNB 0] Frame  104 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 1, DRB 0/LCID 14 already
reconfigured
[RRC][I][eNB 0][VLINK] Frame  104 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 1, CODRB 0/LCID 179 already
reconfigured
```

```
[RRC][I][eNB 1] Frame  104 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 1, DRB 0/LCID 14 already
reconfigured
[RRC][I][eNB 1][VLINK] Frame  104 : Logical Channel UL-DCCH, Received
RRCConnectionReconfigurationComplete from UE 1, CODRB 0/LCID 179 already
reconfigured
```

3.2.2.2. Data-Plane

OAI log messages that show the scheduling and MAC forwarding via a virtual link for the triangle network is shown in Table 11. Each PDU is firstly indicated to the MAC buffer queue that is maintained on the MAC layer of the collaborative MR. In sequence, when a request is scheduled for the specific PDU that is identified by its sequence number, the operation proceeds by removing the PDU from the buffer if this PDU is stored and then the PDU is getting transmitted over the relaying phase.

Table 11. PDU MAC forwarding

```
apaposto@ubuntu:~/openair4G_mesh/targets/SIMU/USER$ ./oaisim -a -b2 -H2 -u1 -w 1
-W1 -L 1 -n 200 -c8 | egrep "Requested|o_cornti|mac_buffer_data_ind"
[MAC][D][UE 0] Frame 10: Requested RRCConnectionRequest, got 6 bytes
[MAC][D][UE 0] Frame 18: Requested RRCConnectionRequest, got 6 bytes
[MAC][I][UE 0][VLINK] Frame 111 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
[MAC][D]mac_buffer_data_ind  PACKET seq_num 1, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 1 for eNB index 1 and cornti 5354
(element 0x16ac04c8)
[MAC][I][UE 0][VLINK] Frame 116 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
[MAC][D]mac_buffer_data_ind  PACKET seq_num 2, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 2 for eNB index 1 and cornti 5354
(element 0x16ac0318)
[MAC][I][UE 0][VLINK] Frame 121 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
[MAC][D]mac_buffer_data_ind  PACKET seq_num 3, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 3 for eNB index 1 and cornti 5354
(element 0x16ac0168)
[MAC][I][UE 0][VLINK] Frame 126 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
[MAC][D]mac_buffer_data_ind  PACKET seq_num 4, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 4 for eNB index 1 and cornti 5354
(element 0x16abfe08)
[MAC][I][UE 0][VLINK] Frame 131 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
[MAC][D]mac_buffer_data_ind  PACKET seq_num 5, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 5 for eNB index 1 and cornti 5354
(element 0x16abfc58)
[MAC][I][UE 0][VLINK] Frame 136 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
[MAC][D]mac_buffer_data_ind  PACKET seq_num 6, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 6 for eNB index 1 and cornti 5354
(element 0x16abfaa8)
[MAC][I][UE 0][VLINK] Frame 141 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
```

```
[MAC][D]mac_buffer_data_ind  PACKET seq_num 7, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 7 for eNB index 1 and cornti 5354
(element 0x16abf8f8)
[MAC][I][UE 0][VLINK] Frame 146 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
[MAC][D]mac_buffer_data_ind  PACKET seq_num 8, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 8 for eNB index 1 and cornti 5354
(element 0x16abf748)
[MAC][I][UE 0][VLINK] Frame 151 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
[MAC][D]mac_buffer_data_ind  PACKET seq_num 9, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 9 for eNB index 1 and cornti 5354
(element 0x16abf598)
[MAC][I][UE 0][VLINK] Frame 156 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
[MAC][D]mac_buffer_data_ind  PACKET seq_num 10, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 10 for eNB index 1 and cornti
5354 (element 0x16abf3e8)
[MAC][I][UE 0][VLINK] Frame 161 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
[MAC][D]mac_buffer_data_ind  PACKET seq_num 11, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 11 for eNB index 1 and cornti
5354 (element 0x16abebd8)
[MAC][I][UE 0][VLINK] Frame 166 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
[MAC][D]mac_buffer_data_ind  PACKET seq_num 12, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 12 for eNB index 1 and cornti
5354 (element 0x16abea88)
[MAC][I][UE 0][VLINK] Frame 171 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
[MAC][D]mac_buffer_data_ind  PACKET seq_num 13, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 13 for eNB index 1 and cornti
5354 (element 0x16abe938)
[MAC][I][UE 0][VLINK] Frame 176 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
[MAC][D]mac_buffer_data_ind  PACKET seq_num 14, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 14 for eNB index 1 and cornti
5354 (element 0x16abe7e8)
[MAC][I][UE 0][VLINK] Frame 181 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
[MAC][D]mac_buffer_data_ind  PACKET seq_num 15, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 15 for eNB index 1 and cornti
5354 (element 0x16abe3f8)
[MAC][I][UE 0][VLINK] Frame 186 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
[MAC][D]mac_buffer_data_ind  PACKET seq_num 16, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 16 for eNB index 1 and cornti
5354 (element 0x16abe2a8)
[MAC][I][UE 0][VLINK] Frame 191 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
[MAC][D]mac_buffer_data_ind  PACKET seq_num 17, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 17 for eNB index 1 and cornti
5354 (element 0x16abe158)
[MAC][I][UE 0][VLINK] Frame 196 : DLSCH->vlink0, i_cornti 3427 -> o_cornti 5354,
src_eNB 0-> dst->eNB 1 (62 bytes)
[MAC][D]mac_buffer_data_ind  PACKET seq_num 18, pdu_size 62, HARQ_proccess_ID 0
[MAC][D][UE 0] Requested...!!! MAC PDU with sn 18 for eNB index 1 and cornti
5354 (element 0x16abe008)
```

3.2.2.3. Latency Results

We have conducted two experimentations considering a mesh topology with in an area of 500x500. The traffic is generated by the Openairinterface traffic generator (OTG) tool. The two experimentations are:

- (Topology 1) 2 CHs assisted by one collaborating MR
- (Topology 2) 2 connected CHs assisted by one regular MR

The following setup is common to both experimentations. We use free space path loss exponent 2.67 with AWGN channel (necessary for demo 1 compatibility).

```
<ENVIRONMENT_SYSTEM_CONFIG>
    <FADING>
      <LARGE_SCALE>urban</LARGE_SCALE>
      <FREE_SPACE_MODEL_PARAMETERS>
           <PATHLOSS_EXPONENT>2.67</PATHLOSS_EXPONENT>
           <PATHLOSS_0_dB>-50</PATHLOSS_0_dB>
      </FREE_SPACE_MODEL_PARAMETERS>
      <SMALL_SCALE>AWGN</SMALL_SCALE>
    </FADING>
    <WALL_PENETRATION_LOSS_dB>5</WALL_PENETRATION_LOSS_dB>
    <SYSTEM_BANDWIDTH_MB>7.68</SYSTEM_BANDWIDTH_MB>
    <UE_FREQUENCY_GHz>1.9</UE_FREQUENCY_GHz>
  </ENVIRONMENT_SYSTEM_CONFIG>
```

The emulation time for both experimentation was set to 5000ms.

```
<EMULATION_CONFIG>
   <EMULATION_TIME_ms>5000</EMULATION_TIME_ms>
   <PERFORMANCE_METRICS>
     <THROUGHPUT>enable</THROUGHPUT>
     <LATENCY>enable</LATENCY>
     <OWD_RADIO_ACCESS>enable</OWD_RADIO_ACCESS>
</PERFORMANCE_METRICS>
   <LOG>
     <LEVEL>debug</LEVEL>
     <VERBOSITY>medium</VERBOSITY>
   </LOG>
   <SEED_VALUE>1234</SEED_VALUE>
 </EMULATION_CONFIG>
```

The topology 1 and 2 configuration. When experimenting, we enable and disable the collaboration, which activate the MR collaboration in packet forwarding operation.

```
<TOPOLOGY_CONFIG>
  <AREA>
    <X_m>500</X_m>
    <Y_m>500</Y_m>
  </AREA>
  <MOBILITY>
    <UE_MOBILITY>
          <RANDOM_UE_DISTRIBUTION>
            <NUMBER_OF_NODES>1</NUMBER_OF_NODES>
          </RANDOM_UE_DISTRIBUTION>
          <UE_MOBILITY_TYPE>STATIC</UE_MOBILITY_TYPE>
    </UE_MOBILITY>
    <eNB_MOBILITY>
          <eNB_INITIAL_DISTRIBUTION>random</eNB_INITIAL_DISTRIBUTION>
          <RANDOM_eNB_DISTRIBUTION>
            <NUMBER_OF_CELLS>2</NUMBER_OF_CELLS>
          </RANDOM_eNB_DISTRIBUTION>
          <eNB_MOBILITY_TYPE>STATIC</eNB_MOBILITY_TYPE>
    </eNB_MOBILITY>
  </MOBILITY>
</TOPOLOGY_CONFIG>
```

Experimentation in both scenarios was conducted by choosing constant bit rate traffic, as described below.

```
<APPLICATION_CONFIG>
  <!-- CH1 to CH2 traffic emulating video -->
  <CUSTOMIZED_TRAFFIC>
    <SOURCE_ID>0</SOURCE_ID>
    <TRANSPORT_PROTOCOL>udp</TRANSPORT_PROTOCOL>
    <IP_VERSION>ipv4</IP_VERSION>
    <DESTINATION_ID>1</DESTINATION_ID>
    <IDT_DIST>fixed</IDT_DIST>
    <IDT_MIN_ms>100</IDT_MIN_ms>
    <IDT_MAX_ms>160</IDT_MAX_ms>
    <SIZE_DIST>uniform</SIZE_DIST>
    <SIZE_MIN_byte>20</SIZE_MIN_byte
    <SIZE_MAX_byte>30</SIZE_MAX_byte>
  </CUSTOMIZED_TRAFFIC>

  <!-- CH2 to CH1: small CBR traffic -->
   <CUSTOMIZED_TRAFFIC>
    <SOURCE_ID>1</SOURCE_ID>
    <TRANSPORT_PROTOCOL>udp</TRANSPORT_PROTOCOL>
    <IP_VERSION>ipv4</IP_VERSION>
    <DESTINATION_ID>0</DESTINATION_ID>
    <IDT_DIST>fixed</IDT_DIST>
    <IDT_MIN_ms>100</IDT_MIN_ms>
    <IDT_MAX_ms>160</IDT_MAX_ms>
    <SIZE_DIST>uniform</SIZE_DIST>
    <SIZE_MIN_byte>20</SIZE_MIN_byte>
    <SIZE_MAX_byte>30</SIZE_MAX_byte>
  </CUSTOMIZED_TRAFFIC>

</APPLICATION_CONFIG>
```

For the first topology, we enabled the the virtual link and MAC forwarding with the bidirectional traffic. The goodput seen at application level in the receiving CHs and also the aggregated one is shown in Figure 28. The latency results are illustrated in Figure 29.

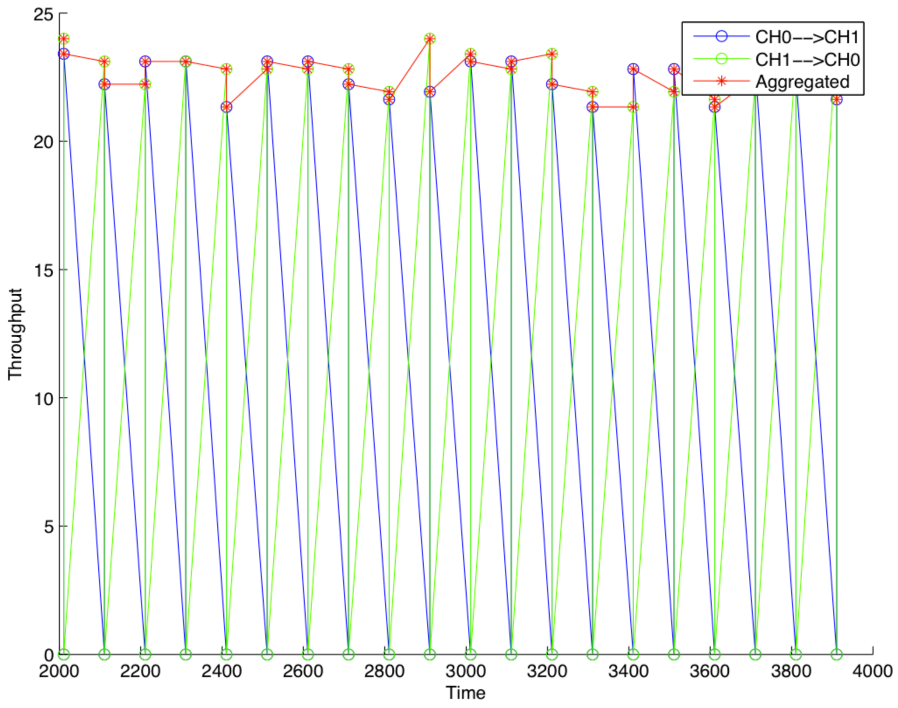


Figure 28 – (Topology 1, 2 CHs and 1 collaborating MR) Throughput received in both directions: (CH0→MR→CH1 and CH1→MR→CH0). The red legend illustrates the aggregated throughput

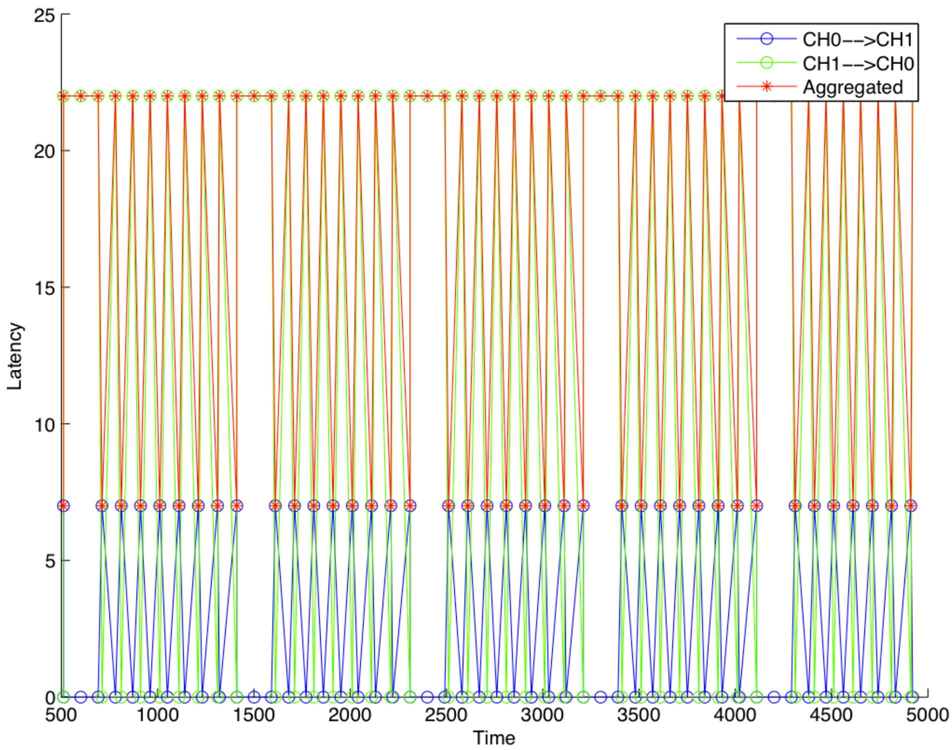


Figure 29 – (Topology 1, 2 CHs and 1 collaborating MR) Latency received in both directions: (CH0→MR→CH1 and CH1→MR→CH0). The red legend illustrates the aggregated latency

The overall statistics of the virtual link between CH0 to CH1 can be summarized as follows.

```
----------------------------------------------------------
Total Time= 3990
[APPLICATION] VLink [CH:0, CH:1]
[APPLICATION] Total packets(TX)= 20
[APPLICATION] Total bytes(TX)= 1515
[APPLICATION] RTT MIN (one way)ms= 27.00
[APPLICATION] RTT MAX (one way)ms= 27.00
[APPLICATION] TX throughput = 0.38(KB/s)
[APPLICATION] RX goodput= 0.38 (KB/s)
----------------------------------------------------------
Total Time= 3990
[APPLICATION] VLink [CH:1,CH0:0]
[APPLICATION] Total packets(TX)= 20
[APPLICATION] Total bytes(TX)= 1539
[APPLICATION] RTT MIN (one way)ms= 27.00
[APPLICATION] RTT MAX (one way)ms= 27.00
[APPLICATION] TX throughput = 0.39(KB/s)
[APPLICATION] RX goodput= 0.39 (KB/s)
**************** TOTAL VLink RESULTS ******************
Total Time= 3990
[DATA] Total packets(TX)= 40
[DATA] Total packets(RX)= 40
[DATA] Total bytes(TX)= 3054
[DATA] Total bytes(RX)= 3054
[DATA] RTT MIN (one way)ms= 27.00
[DATA] RTT MAX (one way)ms= 27.00
[DATA] TX throughput = 0.77(KB/s)
[DATA] RX throughput = 0.77(KB/s)
```

For the second topology, the forwarding is done at the higher layer, namely PDCP in this experiment with a unidirectional traffic. The goodput seen at application level in the receiving CHs and also the aggregated one is shown in Figure 28. The latency results are illustrated in Figure 29.

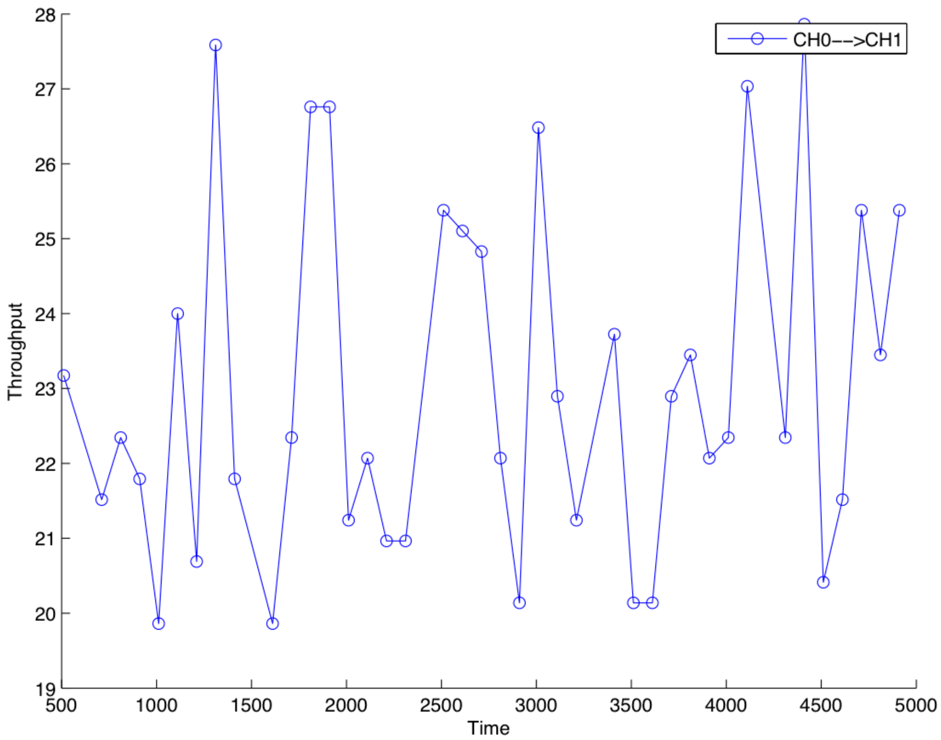


Figure 30 – (Topology 2, 2 connected CHs direct transmission) Goodput from CH0→CH1.

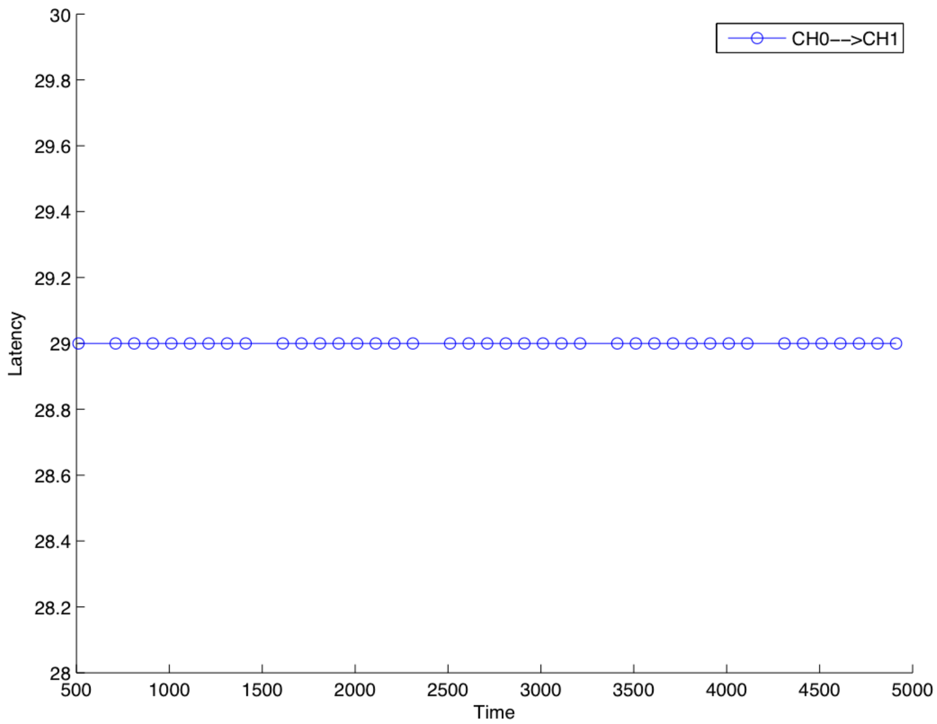


Figure 31 – – (Topology 2, 2 connected CHs direct transmission) Latency from CH0→CH1

The overall statistics of the virtual link between CH0 to CH1 can be summarized as follows.

```
----------------------------------------------------------
Total Time= 3990
[APPLICATION] DirectLink [CH:0, CH:1]
[APPLICATION] Total packets(TX)= 20
[APPLICATION] Total bytes(TX)= 1547
[APPLICATION] RTT MIN (one way)ms= 29.00
[APPLICATION] RTT MAX (one way)ms= 29.00
[APPLICATION] TX throughput = 0.23(KB/s)
[APPLICATION] RX goodput= 0.23 (KB/s)
----------------------------------------------------------
**************** TOTAL DirectLink RESULTS ******************
Total Time= 3990
[DATA] Total packets(TX)= 20
[DATA] Total packets(RX)= 20
[DATA] Total bytes(TX)= 1547
[DATA] Total bytes(RX)= 1547
[DATA] RTT MIN (one way)ms= 29.00
[DATA] RTT MAX (one way)ms= 29.00
[DATA] TX throughput = 0.23(KB/s)
[DATA] RX throughput = 0.23(KB/s)
```

This experimentation assessed the lower bound for the latency for the two topologies using the same configuration as the traffic is chosen to be very low data rate such that the effect of RLC fragmentation and reassembly is mitigated (RLC is acting as an interface to the PDCP). Furthermore in the experimentation, we did not have PDCP RoHC and encryption/decryption function, which in turn mitigate the PDCP processing delay. We observe a decrease in the total latency by 2ms when the collaborative Topology 1 (total latency = 27ms) is used instead of the direct transmission of Topology 2 (total latency 29ms). The results proofs the concept of joint scheduling and MAC forwarding on a virtual link and demonstrate that there is latency gain even in the most unfavourable condition.

3.3. Realtime RF system validation platform

Please note that this part will be filled upon the final demo.

3.3.1. WP4 PHY Layer Techniques

3.3.1.1. Synchronization

3.3.1.2. Cooperative communication strategies (distributed Alamouti)

3.3.1.3. Physical layer procedures

3.3.2. Results

4. CONCLUSION

In this deliverable we summarize the implementation and performance evaluation results of the selected techniques related to the LOLA testbed 3 for rapidly deployable wireless mesh network for safety applications. We provide detail description of the selected techniques and elaborate the implementation details, in particular virtual collaborative logical link, mac forwarding and inter-play with the higher layers. The idea is to create virtual links connecting directly CHs of adjacent clusters such that routing decisions are taken only at the CH level and not at the MR level. Moreover, in order to speed the communications through the MRs, MAC layer joint forwarding and scheduling is implemented. When more than one MR is connecting the same two CHs, cooperation among the MRs can be activated in order to increase the robustness of the communication.

We perform extensive emulations to validate the mesh topology formation, virtual link operation, MAC forwarding and scheduling scheme on the OAI platform. The signaling results in RRC, MAC, and PHY layer show the virtual link and forwarding are properly implemented on OAI. Concretely, the emulation results show that works as following: (1) a CH initiates the establishment of a virtual link triggered by the higher layers (i.e. routing protocol) through RRC signaling; (2) a MR receives this RRC message and in its turn establish the other part of the virtual link; (3) the CH allocates resource for collaborative transmission and multiplex/demultiplex outgoing/incoming traffic to/from the collaborative channel ; (4) a MR will store the MAC PDUs in its local buffer and will transmit through the collaborative channel when it detect a transmission opportunity; (5) if a CO-RNTI is received, the CH will decode and forward the MAC PDU to the higher layers.

The simulation results have shown that there is latency gain without any loss in the average throughput in transmitting a packet through a virtual link in the most unfavourable conditions. The latency gain is mainly related to that of processing time of the higher layers, namely RLC, PDCP, IP and above. The gain is mainly proportional to the packet size as the larger the packet is, the higher is the its latency and processing time at the higher layers.

5. ACRONYMS AND DEFINITIONS

5.1. Acronyms

Acronym	**Defined as**
3GPP	3rd Generation Partnership Project
ACK	Acknowledgement
AMC	Adaptive Modulation & Coding
AVL	Adelson-Velskii-Landis self-balancing binary tree
AT4LP	AT4 wireless LOLA Platform
BCH	Broadcast Channel
BLER	Block Error Rate
BSR	Buffer Status Report
CE	Control Element
CCE	Control Channel Element
CH	Cluster Head
CO-RNTI	Collaborative-Radio Network Temporary Identifier
CO-BSR	Collaborative-BSR
CP	Cyclic Prefix
CQI	Channel Quality Indicator
DAI	Downlink Assignment Index
DCI	Downlink Control Information
DF	Decode & Forward
DL	Downlink
DL-SCH	Downlink-Shared Channel
DRB	Data Radio Bearer
DRS	Demodulation Reference Signal
DwPTS	Downlink Pilot Time Slot
E-UTRAN	Evolved-Universal Terrestrial Radio Access Network
eNB	E-UTRAN Node B
EPA	Extended Pedestrian A
ETU	Extended Typical Urban

EVA	Extended Vehicular A
FDD	Frequency Division Duplex
GP	Guard Period
HARQ	Hybrid Automatic Repeat Request
IOT	Interoperability
IP	Internet Protocol
IPDV	IP Delay Variation
LSB	Least Significant Bit
LTE	Long Term Evolution
MAC	Medium Access Control
M2M	Machine-To-Machine
MBSFN	Multimedia Broadcast over a Single Frequency Network
MCS	Modulation and Coding Scheme
MIMO	Multiple-Input Multiple-Output
MR	Mesh Relay
MSB	Most Significant Bit
NACK	Negative Acknowledgement
NAS	Non-Access Stratum
OFDM	Orthogonal Frequency Division Multiplexing
OWD	One-Way Delay
P-RNTI	Paging-Radio Network Temporary Identifier
PCFICH	Physical Control Format Indicator Channel
PCH	Paging Channel
PDCCH	Physical Downlink Control Channel
PDCP	Packet Data Convergence Protocol
PDF	Probability Density Function
PDSCH	Physical Downlink Shared Channel
PDU	Protocol Data Unit
PER	Packet Error Rate
PHICH	Physical HARQ Indicator Channel

PHY	PHYsical (layer)
PMCH	Physical Multicast Channel
PMI	Precoding Matrix Indicator
PRACH	Physical Random Access Channel
PRB	Physical Resource Block
PUCCH	Physical Uplink Control Channel
PUSCH	Physical Uplink Shared Channel
PSS	Primary Synchronization Signal
QAM	Quadrature Amplitude Modulation
QPSK	Quadrature Phase Shift Keying
RNTI	Radio Network Temporary Identifier
RA-RNTI	Random Access-Radio Network Temporary Identifier
RB	Resource Block
RI	Rank Indicator
RLC	Radio Link Control
RMC	Reference Measurement Configuration
RRA	Radio Resource Allocation
RRC	Radio Resource Control
RRM	Radio Resource Management
RTT	Round Trip Time
SC-FDMA	Single Carrier-Frequency Division Multiple Access
SDK	Software Development Kit
SI-RNTI	System Information-Radio Network Temporary Identifier
SISO	Single-Input Single-Output
SPS	Semi-Persistent Scheduling
SR	Scheduling Request
SRS	Sounding Reference Signal
SSS	Secondary Synchronization Signal
TB	Transport Block
TDD	Time Division Duplex

TV	Test Vector
UCI	Uplink Control Information
UE	User Equipment
UL	Uplink
UpPTS	Uplink Pilot Time Slot
VRB	Virtual Resource Block
VSG	Vector Signal Generator

6. REFERENCES

[1] 3GPP TS 23.401 V9.3.0 (2009-12), "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access (Release 9)".

[2] S. Sesia, I. Toufik and M. Baker, "LTE - The UMTS Long Term Evolution - From Theory to Practice", Second Edition, John Wiley & Sons, 2011.

[3] 3GPP TS 36.213 V10.1.0 (2011-03), "3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Physical Layer Procedures (Release 10)".

[4] LOLA Project (Achieving Low-Latency in Wireless Communications), "D4.3 Adaptive modulation and coding scheme and HARQ mechanism", December, 2013.

[5] LOLA Project (Achieving Low-Latency in Wireless Communications), "D4.2 Enhancements to framing and low –layer signalling". March, 2012.

[6] LOLA Project (Achieving Low-Latency in Wireless Communications), "D5.6 First report on integration of WP4 algorithms on testbed 3", v1.1, July, 2012.

[7] LOLA Project (Achieving Low-Latency in Wireless Communications), "D3.5 Traffic Models for M2M and Online Gaming Network Traffic", v1.5, Jan, 2012.

[8] M. Laner, P. Svoboda, and M. Rupp, "modeling randomness in Network Traffic", SIGMETRICS, 2012.

[9] M. J. Neely. "Stochastic Network Optimization with Application to Communication and Queueing Systems", Morgan & Claypool, 2010.

[10] Leonidas, Georgiadis, Michael, Neely, Leandros, Tassiulas, "Resource allocation and cross-layer control in wire-less networks, Resource allocation and cross-layer control in wire-less networks, 2006.