Self-Backhauled Autonomous LTE Mesh Networks

Romain Favraud, Chia-Yu Chang, Navid Nikaein
Communication Systems Department, EURECOM, France
Email: firstname.name@eurecom.fr

***Abstract*—Reliable service provisioning is crucial to the public safety (PS) communications especially when network outage happens. Isolated E-UTRAN operation, introduced in LTE Release 13, is able to host separate core network functions at the base stations (BSs) to provide limited set of services to the users. However, a significant issue remains to be solved is to coordinate among BSs to create an autonomous network and enhance service availability and reliability. In this paper, an in-band LTE self-backhauling operation leveraging the relay interface is proposed to create an autonomous mesh network of BSs. This calls for an efficient resource allocation for multiple unplanned backhaul links between BSs. To this end, we present a cross-layer scheduling problem for in-band self-backhauled LTE network, and provide a interference-aware hierarchical resource allocation algorithm that is able to meet specific quality of service (QoS) requirements for real-time traffic while adapting to the workload of other types of traffic, through efficient leverage of FDD capabilities and network frequency re-use. Finally, a thorough evaluation on our proposed approach is realized via extensive simulations on different network topologies and diverse traffic flows, and the results demonstrate our work effectiveness utilization of available resources to satisfy QoS requirements.**

I. Introduction

As posed in the dominating position in the current 4G era, Long Term Evolution (LTE) is still continuing to evolve and extend its features to be compatible with divergent 5G use-cases. Among these cases, the Public Safety (PS) communications is crucial for the future radio access technologies (RATs) and highly desirable by ITU [1]. In that sense, several specification groups and working items have now emerged in this regard to make current LTE to be compliant with PS communications as summarized in [2].

Unlike common scenarios, LTE Base Stations (BSs), called eNodeBs (eNBs), may lose access to the Evolved Packet Core (EPC) in PS use-case due to the network outage or lack of equipment. When it happens, there is no possibilities for the network to provide any service to all served user equipments (UEs). For this purpose, 3GPP addresses the *Isolated E-UTRAN* concept for the PS communications that allows eNBs to continue providing minimal services for local PS UEs (TS 22.346, TR 23.797). In [3], an evolution of the *Isolated E-UTRAN* is proposed as a new BS architecture for nomadic and autonomous LTE networks called as enhanced eNB (e2NB). Each e2NB now embeds essential core network functions (i.e. a micro EPC that hosts MME and HSS in Fig. 1) to provide local services and has the capability to connect with other e2NBs via incorporating virtual UEs (vUEs) and leveraging the LTE relay interface (i.e. *Un* interface) to create a mesh

Research leading to these results received funding from Naval Group and 5G-PPP COHERENT project under grant agreement 671639.

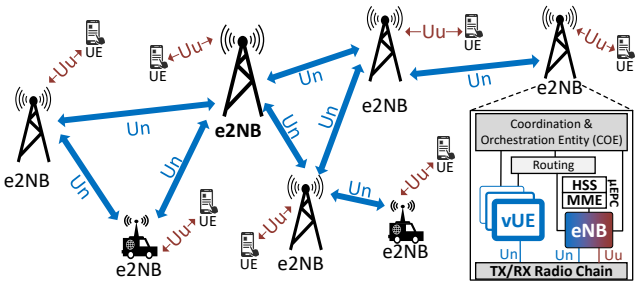


Fig. 1: LTE mesh network based on e2NB with LTE backhaul

network as depicted in Fig. 1. Such e2NB architecture allows to support PS use-cases, vehicular communications or other scenarios where dynamic meshing among fix and/or moving BSs [4] is required.

Despite its appeal, several challenges are encountered when applying such e2NB architecture. First, the 3GPP *Un* interface is leveraged to relay traffic between e2NBs, which is different from the legacy *Uu* interface between UE and eNB. As shown in [5], the physical channels of *Un* interface have comparable performance to the ones of *Uu* interface and allow efficient transmissions between e2NBs. Further, another challenge is to dynamically mesh e2NBs when reusing a single carrier frequency between *Un* and *Uu* interface. Such frequency reuse manner will require the cooperation and coordination between e2NBs to avoid excessive inter-cell interference. Last but not least, the final challenge is to maintain the Quality of service (QoS) of each end-to-end connection to provide service reliability and resilience for self-backhauled network. This QoS maintenance can be based on diverse requirements on different user-plane traffic (e.g. latency of VoIP traffic).

In this paper, we propose a joint access link (*Uu*) and backhaul link (*Un*) resource allocation algorithm with QoS guarantee in a single frequency self-backhauled LTE mesh network. Our contributions are summarized as follows:

- We highlight some important characteristics of *Un* interface, introduce the design elements, and consider traffic patterns of the self-backhauled mesh network (Section II);
- We propose an interference-aware cross-layer hierarchical approach to allocate resources while providing QoS guarantee, leveraging FDD capabilities if available to create an autonomous network (Section III);
- We evaluate the performance of the proposed approach over several network topologies via extensive simulations (Section IV).

Finally, we conclude this work and present the next steps of our research.

II. Design Elements of Self-Backhauled Autonomous LTE Network

Before unveiling the potential issues of the in-band self-backhauled LTE mesh network, we first briefly provide some essential design elements.

A. LTE Relay Channel (Un)

The feature of LTE relay is first standardized since 3GPP Release 10 which allows a relay node (RN) to serve UEs on its access link (*Uu* interface) and reach the EPC through its backhaul link (*Un* interface) with its anchor eNB, called Donor eNB (DeNB) [6]. However, the interference between access link and backhaul link requests a sufficient isolation [7] which can be enabled via multiplexing in frequency (a.k.a. out-band approach) or time domain (a.k.a. in-band approach). For the out-band case, an additional carrier frequency is required and the frequency domain multiplexing (FDM) is applied. Whereas the in-band characteristic between *Uu* and *Un* interface will reuse a common frequency band in a time division multiplexing (TDM) manner. Since few impacts can be observed when using FDM manner, we focus on the in-band case in this work.

To enable such behavior, the *Un* interface utilizes a mechanism that is originally introduced in LTE eMBMS (enhanced Multimedia Broadcast Multicast Service, a.k.a. MBSFN) to differentiate multicast/broadcast subframes (SFs) from the unicast ones in a TDM way within a single LTE frame[1]. Moreover, according to the 3GPP standard (TS 36.331), in an LTE FDD frame, a maximum of 6 MBSFN SFs are allowed; that is to say, at least 4 non-MBSFN SFs are in each frame. In that sense, a relay can exploit MBSFN SFs for backhaul link from/to its DeNB on the downlink(DL)/uplink(UL) direction and non-MBSFN SFs to communicate with all associated UEs on the access link [8]. To sum up, the TDM-based MBSFN frame structure is complement to the in-band backhauling using LTE relay interface; however, the backhaul link can only reach up to 60% of maximum achievable data rate on the access link as it can use a maximum of 6 SFs per frame.

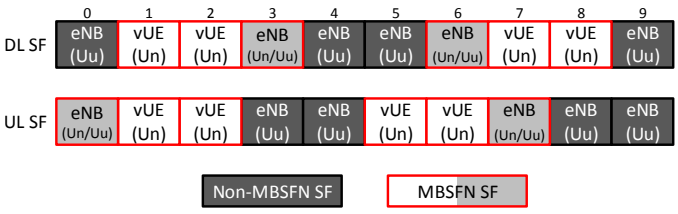


Fig. 2: Example of SFs allocation at an e2NB

B. Single Frequency Wireless Mesh Network

As indicated hereinbefore, the *Un* interface is used to extend the eNB capability of the e2NB with in-band self-backhauling to realize self-organized LTE mesh network. There are some other works regarding to use the *Un* interface for self-backhauling [9], [10]; however, they neither considered meshing BSs that host their own EPC nor had a preferential backhaul path towards a gateway. To the best of our knowledge, this is the first work that aims to utilize the *Un* interface to mesh a self-backhauling single frequency LTE network.

[1]Recall that an LTE frame is composed of 10 subframes each with 1ms time duration.

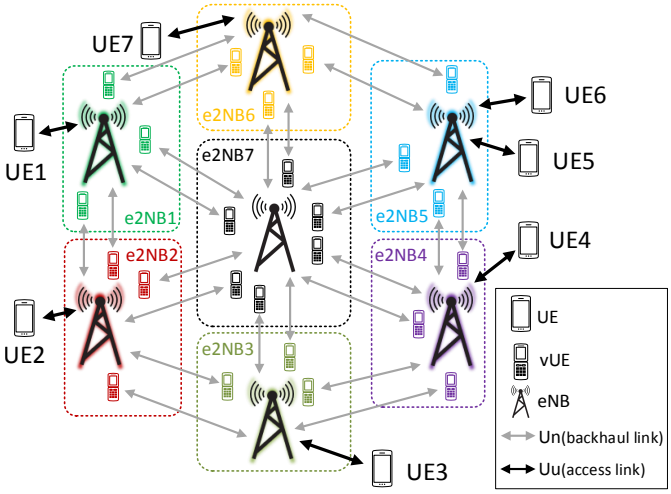


Fig. 3: Example of network with full e2NB mesh.

In Fig. 2, an example of frame structure at an e2NB is presented with 4 non-MBSFN SFs dedicatedly allocated for access link (i.e, DL SF 0,4,5,9) and up to 6 MBSFN SFs that can be allocated for backhaul links. We can notice that all 6 MBSFN SFs are shared between vUEs (i.e. DL SF 1,2,7,8) and eNB (i.e. DL SF 3,6) in a single e2NB and allows the e2NB to communicate with several adjacent e2NBs.

An example of the resulted network topology is shown in Fig. 3. We can see that there are 7 e2NBs within this network to serve all 7 UEs; however, further 24 vUEs (6 vUEs for $e2NB_7$ and 3 vUEs for each of the rest e2NBs) are included in all e2NBs in order to fully mesh the network. These 24 vUEs can be used flexibly to relay the traffic in backhaul links using the *Un* interface. Take a traffic flow that sources from UE1 to destination UE3 as an example, there exists 8 possibilities (i.e., 2 possible routes time 4 possible combinations of direction) when considering a 3-hop relaying case as listed in TABLE I. Hence, all these links between eNBs and UEs/vUEs shall be considered jointly to mesh a self-backhauling single frequency network without introducing strong interference to each others. Last but not least, the routes and directions of each incoming packet shall be selected carefully to avoid excessive latency.

TABLE I: 3-hop relaying of traffic from UE1 to UE3

Src→Hop1 (UE1→e2NB1)	**Hop1→Hop2** (e2NB1→e2NB2/e2NB7)	**Hop2→Hop3** (e2NB2/e2NB7→e2NB3)	**Hop3→Dest** (e2NB3→UE3)
UL (UE→eNB)	DL (eNB→vUE)	DL (eNB→vUE)	DL (eNB→UE)
	DL (eNB→vUE)	UL (vUE→eNB)	
	UL (vUE→eNB)	DL (eNB→vUE)	
	UL (vUE→eNB)	UL (vUE→eNB)	

However, there are still more issues that need to be dealt with to realize an efficient wireless mesh network over a single frequency, including (a) topology control, (b) routing, (c) link scheduling, (d) interference measurement, and (e) power control, as summarized in [11]. All these issues are

highly inter-dependent across different network layers and can not be solved separately. To this end, a cross-layer approach is necessary to deal with several issues jointly in order to guarantee the per-flow QoS.

C. Coordination and Orchestration Entity (COE)

To enable such cross-layer approach, we proposed in [2], [3] to rely on a logically centralized control entity that manages and orchestrates the mesh network through *policy enforcement*. The COE is a logically centralized entity that is connected to a number of COE agents [12], one per e2NB in a typical case (refer to Fig. 1). The COE agent can either act as a local controller delegated by the centralized controller, or in coordination with other agents and the centralized COE controller. The communication protocol between the centralized controller and agents is done through bi-directional message exchange over the backhaul links. In one direction, the COE agent sends measured performance indicators and e2NB status to the centralized controller and other agents, while in the other direction the centralized controller enforces policies that define the operation to be executed by the agents and the underlying eNB and vUEs. Such design provides substantial flexibility to realize the hierarchical approach, and is able to reduce the control overhead by delegating more functions to the COE agent at the cost of less coordination.

D. Traffic pattern

In the considered scenario, each e2NB can not only operate independently by hosting its own services but also provide services to the other BSs through the self-backhauled LTE mesh network. Moreover, two different types of traffic are considered: real-time traffic with latency requirement (e.g. VoIP) and elastic traffic that is served in the best-effort manner. A timely-delivery of real-time traffic is essential to guarantee QoS for PS purpose [13]. Further, the traffic flow in our considered scenario can be heterogeneously distributed which is composed of intra-cell[2] or inter-cell UE-to-UE (U2U), eNB-to-eNB (N2N) and UE-to-eNB (U2N/N2U) communications. For instance, in Fig. 3, $e2NB_5$ can locally route the intra-cell U2U traffic between UE5 and UE6, and backhaul route both the inter-cell U2U traffic between UE4 and UE7 and N2N traffic from $e2NB_4$ to $e2NB_6$.

III. Proposed scheduling algorithms

In this section, we firstly outline the proposed hierarchical approach for the resource scheduling problem, and then detail on algorithms for both centralized and distributed schedulers.

A. Overall approach

Based on the considered scenario, all related cross-layer parameters shall be scheduled in a centralized manner which include (a) next e2NB hop for backhaul relaying (global routing), (b) MBSFN SFs for backhaul transportation (cf. Fig. 2), (c) relaying transportation direction (DL/UL), and (d) Low-layer transportation resource (e.g., physical resource blocks (PRBs), modulation and coding scheme (MCS)) for both access and backhaul links. However, due to the limitation of real deployment and implementation, the propagation of control messages over the backhaul links cannot be instantaneous. Thus, parameters that shall be scheduled in a real-time manner (e.g. (c), (d) for link scheduling) need to be handled distributively whereas some others can be allocated centrally in a larger time-scale benefiting from a whole network view (e.g. (a), (b) for node scheduling).

To enable such hierarchical approach, network information is necessary to be abstracted, for instance, the link status between two adjacent e2NBs is crucial for the central node scheduler and it can be derived from the per-link queue size at each e2NB. The local COE agent is responsible to make such abstraction based on the underlying network information. Via the abstraction, the centralized scheduler can have a simple but sufficient information for a scalable network-wide coordination on resource allocation.

B. Topology Control

As mentioned beforehand, the network view is centralized and the topology control aims to relay each traffic flow between e2NBs. Here, we use the standard graph notation $\mathcal{G} = (\mathcal{V}_{e2NB}, \mathcal{E}_{link})$ to represent the network. The vertex set $\mathcal{V}_{e2NB}$ comprises all e2NBs in the network, for instance, $\mathcal{V}_{e2NB} = \{e2NB_1, e2NB_2, \cdots, e2NB_7\}$ in Fig. 3. The bidirectional edge (u, v) represents a link between two e2NBs where e2NB u acts as an eNB and e2NB v as a vUE, and the edge set $\mathcal{E}_{link}$ comprises all these links in the mesh network. Lastly, a neighboring vertice set of e2NB u is defined as N_u that comprises all its adjacent e2NBs, e.g., $N_{e2NB_2} = \{e2NB_1, e2NB_3, e2NB_7\}$ in Fig. 3.

Based on the graph definition, we use the Dijkstra's shortest path algorithm in terms of the number of hops to decide the route in the backhaul links of each traffic. Moreover, since the orthogonal frequency division multiple access (OFDMA) scheme is applied in LTE, different traffic flows can be routed at the same time along their shortest paths via multiplexing access in the frequency domain resources (i.e. PRBs). Such algorithm can significantly reduce the per-flow latency generated by extra hops; however, it can also be adapted to different edge weights to cope with different traffic patterns and network topologies [14]. After the routing and using the real-time ow information available at the COE, we can compute the link load on edge (u, v) as $load_{u,v}$ in terms of the number of real-time traffic bits to be transported in a SF. Such metric will be further used to schedule resource of real-time traffic.

C. Resource scheduling

The main scheduling problem is to share the time resources (MBSFN SFs) and frequency resources (PRBs) between e2NBs. As mentioned before, such scheduling is achieved in a hierarchical manner as summarized in Algorithm 1. Here, we introduce the *superframe* concept which defines the periodicity[3] (P_{SuF} in Algorithm 1) to update the centralized node

[2]Intra-cell traffic can be routed locally.

[3]Normally in hundreds of milliseconds.

Algorithm 1: Hierarchical Scheduling Algorithm

Input : P_{SuF} is the SuperFrame update periodicity
$\mathcal{V}_{e2NB}$ is the set of e2NBs
$rtFlow$ is the newly added real-time flow

begin
  $SF = 0$ /* Initialize subframe index */
  while *network running* **do**
    $SF = SF + 1$ /* Current subframe index */
    $Event = 0$
    if *rtFlow* **then**
      $Event = 1$
    if $SF \equiv 0 \pmod{P_{SuF}}$ *or* $Event == 1$ **then**
      $SF^{pre}_{MBSFN} = SF_{MBSFN}$ /*Store previous results*/
      foreach $u \in \mathcal{V}_{e2NB}$ **do**
        $SF^{pre}_{TX}[u] = SF_{TX}[u]$ /*Store previous results*/
      $compute(L_{SuF})$ (cf. Eq. (1))
      $rtDisSat = 1$
      while $rtDisSat == 1$ **do**
        $compute(SF_{rt})$ (cf. Alg. 2)
        $compute(SF^{D}_{e})$ (cf. Eq. (3))
        $compute(SF^{U}_{e})$ (cf. Eq. (3))
        $[SF_{TX}, rtDisSat] =$
        $central\_NS(SF_{rt}, SF^{D}_{e}, SF^{U}_{e})$ (cf. Alg. 3)
        if $rtDisSat == 1$ **then**
          $reject()$ /*too many real-time flows*/
          $compute(L_{SuF})$ (cf. Eq. (1))
    foreach $u \in \mathcal{V}_{e2NB}$ **do**
      $distributed\_LS\,(u, SF, SF_{TX})$ (cf. Alg. 5)

scheduler results (SF_{TX} from $central\_NS$ in Algorithm 1). Further, such update can also be triggered via some events like a newly-added real-time traffic flow ($rtFlow$ in Algorithm 1). Afterwards, each distributed link scheduler will based on the results of the centralized scheduler to allocate the low-layer transportation resources at every SF ($distibuted\_LS$ in Algorithm 1). In the following, we elaborate on these two schedulers in detail.

1) Centralized node scheduler (NS): As its centralized manner, NS will firstly get the duration of superframe (L_{SuF} in Algorithm 1) that is used by all e2NBs for self-backhauling. Then, all time-domain MBSFN SFs within the superframe duration can be scheduled for self-backhauling. Its goal is to allocate enough SFs within the superframe duration to each e2NB in order to fulfill the real-time traffic transportation bandwidth. If that is not possible, a dissatisfaction indicator is used ($rtDisSat$ in Algorithm 1) to enable the flow control operation via rejecting or removing some traffic flows based on their pre-defined priorities. The COE is responsible to manage all real-time flows via integrating a flow control entity.

As the first step, the duration of the superframe (normally in tens of milliseconds) is determined as in Eq. (1) where $MaxLat$ is the maximum acceptable latency for the real-time flows (e.g. 150 ms for VoIP detailed in section IV), M_{hops} is the expected maximum number of hops for all active real-time traffics and $offset$ is a stretch factor of M_{hops} that depends on the mobility pattern (i.e. vehicular speed) and network size. Note such superframe duration is the same for all e2NBs and we define a set SF_{MBSFN} that comprises all MBSFN SFs

Algorithm 2: Computation of SF_{rt}

Input : $\mathcal{V}_{e2NB}$ is the set of e2NBs
L_{SuF} is the superframe duration
$load_{u,v}$ is the link load between u and v
N_u is set of the neighboring e2NB of u

begin
  foreach $u \in \mathcal{V}_{e2NB}$ **do**
    $rPRB[u] = 0$ /* Initialize required PRBs for u */
    $nPRB[u] = 0$ /* Initialize allocated PRBs for u */
    foreach $v \in N_u$ **do**
      $LL_{u,v} = load_{u,v}$ /* Initialize each link load */
  foreach $u \in \mathcal{V}_{e2NB}$ **do**
    $rPRB[u] = \sum_{v \in N_u} PRB^{D}_{u,v}(LL_{u,v} \cdot L_{SuF})$
    $SF_{rt}[u] = \left\lceil \frac{rPRB[u]}{N_{PRB}} \right\rceil$
    $nPRB[u] = SF_{rt}[u] \cdot N_{PRB}$
    foreach $v \in N_u$ **do**
      $tPRB =$
      $\min(PRB^{U}_{v,u}(LL_{v,u} \cdot L_{SuF}), nPRB[u])$
      $nPRB[u] = nPRB[u] - tPRB$
      $LL_{v,u} =$
      $\max(0, LL_{v,u} \cdot (1 - \frac{tPRB}{PRB_{v,u}(LL_{v,u} \cdot L_{SuF})}))$

within this superframe duration.

$$L_{SuF} = \lceil MaxLat/((M_{hops} + offset)) \rceil \tag{1}$$

After getting the duration of the superframe, we then compute the number of SFs within this superframe duration that is required by each e2NB to transport real-time flows as $SF_{rt}[u]$, $\forall u \in \mathcal{V}_{e2NB}$. It is summarized in Algorithm 2 with inputs $N[u]$ and $load_{u,v}$ as previously introduced. Note the main equation to get the $SF_{rt}[u]$ is computed based on dividing the number of required PRBs (i.e. $rPRB[u]$) to the total number of PRBs per SF (i.e. N_{PRB}). In the computation of the required PRB, we firstly multiply $LL_{u,v}$ by L_{SuF} to get the number of bits within the superframe duration and introduce a function $PRB^{D/U}_{u,v}(x)$ that returns the number of required PRBs to transport x bits over link (u, v) on DL/UL direction. Further, the FDD characteristic (i.e. full-duplex of DL/UL) is leveraged to allocate the link load of reverse direction (i.e. (v, u)) within the allocated PRBs (i.e. $nPRB[u]$).

Before introducing the counterparts of elastic traffics, we firstly introduce the "saturated" concept. A SF over DL ($DL(u, v)$) or UL direction ($UL(u, v)$) of edge (u, v) is viewed as a "saturated SF" when it can only transport less bits than the queued bits of elastic traffic flows. A direction is then considered to be "saturated" if the ratio of the number of "saturated SF" and all allocated SFs is higher than a pre-defined value, e.g., 90%. Finally, saturated neighboring vertex sets of e2NB u in DL and UL directions are defined in Eq. (2).

$$S^{D}_{u} \triangleq \{v : v \in N_u, DL(u, v) \text{ is saturated}\} \tag{2a}$$

$$S^{U}_{u} \triangleq \{v : v \in N_u, UL(v, u) \text{ is saturated}\} \tag{2b}$$

Based on the defined S^{D}_{u} and S^{U}_{u}, two metrics are computed for elastic traffics, i.e., $SF^{U}_{e}[u]$ and $SF^{D}_{e}[u]$, to represent the number of SFs that are required by e2NB u to compensate the saturated links as detailed in Eq. (3). As the first step,

we estimate the average frequency reuse (AFR) factor from the previous scheduling results (i.e. SF^{pre}_{MBSFN}, $SF^{pre}_{TX}[u]$ in Algorithm 1) where $|\cdot|$ is the set cardinality. Such AFR indicates the level of resource reusing in the whole network and will be larger than 1. Then, we can get the number of "free SFs" in a superframe duration as SF_{free} via excluding SFs for real-time traffics from all MBSFN SFs. Afterwards, we define $B^D_e[u]$ and $B^U_e[u]$ as the sum of average transport block size per PRB of all saturated DL and UL links from u, respectively. These two summations use $TBS(a,b)$ function which returns the transport block size (TBS) when applying MCS index a with b PRBs. $MCS^{D/U}_{u,v}$ represent the applied MCS index on edge (u,v) over DL/UL direction. Finally, the number of SFs that are required by u for elastic traffic of UL/DL directions are derived as $SF^U_e[u]$ and $SF^D_e[u]$.

$$AFR = \frac{\sum_{u\in\mathcal{V}_{e2NB}} SF^{pre}_{TX}[u]}{|SF^{pre}_{MBSFN}|} \tag{3a}$$

$$SF_{free} = \left(|SF_{MBSFN}| - \sum_{u\in\mathcal{V}_{e2NB}} SF_{rt}[u]\right)\cdot AFR \tag{3b}$$

$$B^D_e[u] = \sum_{v\in S^D_u} TBS\left(MCS^D_{u,v}, 1\right) \tag{3c}$$

$$SF^D_e[u] = \left\lceil \frac{B^D_e[u]\cdot SF_{free}}{\sum_{v\in\mathcal{V}_{e2NB}} B^D_e[v]} \right\rceil \tag{3d}$$

$$B^U_e[u] = \sum_{v\in S^U_u} TBS\left(MCS^U_{v,u}, 1\right) \tag{3e}$$

$$SF^U_e[u] = \left\lceil \frac{B^U_e[u]\cdot SF_{free}}{\sum_{v\in\mathcal{V}_{e2NB}} B^U_e[v]} \right\rceil \tag{3f}$$

Based on the above derivations, the central NS is shown in algorithm 3 in order to allocate SFs for both real-time and elastic traffics. $SF_{TX}[u][v]$ and $SF_{RX}[v][u]$ are the set of SFs used for transmission and reception on edge (u,v), respectively[4]. The main design principle of this algorithm is to allocate SFs based on the prioritization of real-time traffic over elastic traffic such that there is no overlap between e2NBs that would be interfering too much. As the output, SF_{TX} contains all transmitting SFs during a superframe and $rtDisSat$ indicates if the scheduler can satisfy all required SFs for real-time traffics or not.

Last but not least, the interference blocking set $\mathcal{I}_{u,v}$ comprises the e2NBs which shall be blocked due to the transmission on edge (u,v) as shown in Algorithm 4. Note the decision is made based on the received signal power (i.e. $P_{u,v}$) and a pre-specified blocking criteria as $criteria(a,b)$ which can be the difference of two input signal power (i.e. like A3 event in handover) or the difference of mapped MCS index from input signal power.

2) Distributed scheduling: Based on the results of centralized NS scheduling (i.e. SF_{TX}), the distributed LS scheduler is aimed to allocate the frequency domain resource (i.e. PRB) and transported bits (i.e. TBS) at each e2NB u in per-SF basis as shown in Algorithm 5. Here, a local network view is maintained by each e2NB via forming the vUE set (i.e., $\mathcal{V}_{vUE}$) and UE set (i.e., $\mathcal{V}_{UE}$) for its link scheduling purpose. Our designed algorithm is to prioritize backhaul links (i.e. vUE at Un interface) over access link (i.e. UE at Uu interface) as the former one can reach 60% of pack rate as the latter one and also prioritize real-time traffics over elastic ones. Further, in

[4]The wildcard character in algorithm 3 represent all possible e2NBs.

Algorithm 3: Centralized Node Scheduler (centralized_NS)

Input : SF_{MBSFN} is the set of MBSFN SFs during L_{SuF}
$\mathcal{V}_{e2NB}$ is the set of e2NBs
$\mathcal{E}_{link}$ is the set of all links in the mesh network.
SF_{rt}, SF^D_e, SF^U_e.
Output: SF_{TX}, $rtDisSat$

```
foreach u ∈ V_e2NB do
    foreach v ∈ V_e2NB and (u,v) ∈ E_link do
        SF_TX[u][v] = MBSFN /* Initialize transmit SFs */
        SF_RX[v][u] = MBSFN /* Initialize receive SFs */
rtDisSat = 1
foreach SF ∈ SF_MBSFN do
    sort_descend(V_e2NB, SF_rt, SF^D_e, SF^U_e)
    A_e2NB = ∅ /* Initialize active e2NB set */
    foreach u ∈ V_e2NB do
        if SF_rt[u] ≥ 1 or SF^D_e[u] ≥ 1 or SF^U_e[u] ≥ 1 then
            if SF ∈ SF_TX[u][*] then
                Transmit = 0 /* Initialize transmit indicator */
                AllvUE = 1 /* Initially assume all vUEs are on*/
                foreach v ∈ V_e2NB and (u,v) ∈ E_link do
                    if SF ∈ SF_RX[v][u] then
                        I_{u,v} = genIntf(u,v) (c.f. Alg. 4)
                        if ∃ w ∈ A_e2NB and w ∈ I_{u,v} then
                            remove(SF, SF_RX[v][u])
                            remove(SF, SF_TX[u][v])
                            AllvUE = 0
                        else
                            Transmit = 1
                            remove(SF, SF_RX[u][*])
                            remove(SF, SF_TX[v][*])
                            foreach w ≠ u ∈ V_e2NB and
                              (v,w) ∈ E_link do
                                remove(SF, SF_RX[v][w])
                            foreach w ∈ I_{u,v} do
                                remove(SF, SF_TX[w][*])
                    else
                        remove(SF, SF_TX[u][v])
                        AllvUE = 0
                if Transmit == 1 then
                    A_e2NB = u ⋃ A_e2NB
                    if AllvUE == 1 then
                        if SF_rt[u] > 1 then
                            SF_rt[u] = SF_rt[u] − 1
                        else if SF^D_e[u] > 1 then
                            SF^D_e[u] = SF^D_e[u] − 1
                        else if SF^U_e[u] > 1 then
                            SF^U_e[u] = SF^U_e[u] − 1
    if SF_rt[u] == 0 ∀u then
        rtDisSat = 0
```

Algorithm 4: Generate interferer e2NB set ($genIntf\,(u,v)$)

Input : (u,v) is the edge of the graph from u to v
$P_{x,w}$ is the received signal power at e2NB w from x
$criteria(a,b)$ is the blocking criteria with input a, b.
Output: $\mathcal{I}_{u,v}$
begin
  $\mathcal{I}_{u,v} = \varnothing$
  foreach $w \in \mathcal{V}_{e2NB}$ *and* $w \neq u$ **do**
    if $criteria(P_{u,v}, P_{w,v})$ **then**
      $\mathcal{I}_{u,v} = w \bigcup \mathcal{I}_{u,v}$

Algorithm 5: Distributed Link scheduler (distributed_LS)

Input : u is current e2NB identifier
SF is current subframe identifier
$\mathcal{V}_{UE}$ is a set of UEs by u with non-empty queues
$\mathcal{V}_{vUE}$ is a set of vUEs at u with non-empty queues
$Q[x][p]$ is queue size of (v)UE x with flow priority p
N_{PRB} is the number of available PRBs within SF
SF_{TX} is transmit SFs from Algorithm 3
MCS is the applied MCS index of vUE/UE
Output: PRB, TBS
$sort\_descend(\mathcal{V}_{UE}, Q[*][0])$ /* sort UEs based on queue size */
$sort\_descend(\mathcal{V}_{vUE}, Q[*][0])$ /* sort vUEs based on queue size */
foreach $x \in \mathcal{V}_{UE} \bigcup \mathcal{V}_{vUE}$ **do**
  $PRB\,[x] = 0$ /* Initialize allocated PRB */
  $TBS\,[x] = 0$ /* Initialize allocated TBS */
$priority = 0$ /* 0: real-time, 1: elastic */
while $N_{PRB} > 0$ *and* $priority < 2$ **do**
  $satisfy = 1$ /* Indicate flows of current priority are satisfied */
  foreach $x \in \mathcal{V}_{vUE}$ **do**
    if $SF \in SF_{TX}\,[u]\,[x]$ **then**
      $ReqBit = \sum_{p=0}^{priority} Q\,[x]\,[p]$
      if $N_{PRB} > 0$ *and* $TBS[x] < ReqBit$ **then**
        $PRB\,[x] = PRB\,[x] + 1$
        $N_{PRB} = N_{PRB} - 1$
        $TBS[x] = TBS\,(MCS[x], PRB[x])$
        $satisfy = 0$
  if $satisfy == 1$ **then**
    foreach $x \in \mathcal{V}_{UE}$ **do**
      $ReqBit = \sum_{p=0}^{priority} Q\,[x]\,[p]$
      if $SF \in SF_{TX}\,[u]\,[*]$ **then**
        if $N_{PRB} > 0$ *and* $TBS[x] < ReqBit$ **then**
          $PRB\,[x] = PRB\,[x] + 1$
          $N_{PRB} = N_{PRB} - 1$
          $TBS[x] = TBS\,(MCS[x], PRB[x])$
          $satisfy = 0$
  if $satisfy == 1$ **then**
    $priority = priority + 1$

the PRB provisioning, the number of requested bit ($ReqBit$) within the corresponding queues is considered to avoid over-provisioning. Note that Algorithm 5 shows only the DL link scheduling but the same algorithm is also used for UL link scheduling for UEs/vUEs using UL queue sizes and UL SFs corresponding to SF_{TX}. Lastly, such link scheduling can be adopted to apply the inter-(v)UE priority based on pre-defined purpose policy and charing function (PCRF) rather than the round-robin manner used in Algorithm 5.

D. Relaying direction selection

As shown in TABLE I, both DL and UL directions can be selected when relaying in backhaul links. It means that each packet can go over the DL or UL direction to reach the next hop. However, the selection of direction highly depends on the traffic QoS requirement, for instance, an ultra reliable traffic will select the one with better signal quality whereas the mobile broadband traffic prefers the one with higher throughput. Since our considered real-time traffic is sensitive to the latency, we use the expected waiting time of both UL and DL queues as the metric to decide which queue (DL or UL) is more preferred.

In this section, a hierarchical scheduling algorithm (cf. Algorithm 1) is proposed that is composed of a centralized node scheduler (cf. Algorithm 3) and a distributed link scheduler (cf. Algorithm 5). Note the centralized and distributed schedulers are executed in different time-scale, i.e., superframe and subframe respectively, for several purposes: (i) reduce excess control-plane overhead of full centralization, (ii) reuse the legacy subframe-based link adaption scheduler, and (iii) flexible network management and orchestration. Finally, TABLE II summarizes these two schedulers, and our proposed approach shows its practical use and implementation while being compliant with legacy design.

TABLE II: Comparison of centralized/distributed schedulers

Characteristic	Centralized NS	Distributed LS
Network view	Central view with $\mathcal{G}$	Local view with $\mathcal{V}_{vUE}$, $\mathcal{V}_{UE}$
Periodicity	superframe	subframe
Considered link	Backhaul link	Backhaul and Access link
Scheduled resource	Time-domain MBSFN SF	Frequency-domain PRB
Legacy compliance	No legacy design	Compliant with *Uu* scheduler
Interference impact	Interference coordination	Link adaptation
Node prioritization	Prioritize e2NB with high real-time demand	Prioritize vUE over UE
Traffic prioritization	Prioritize real-time traffic over elastic traffic	

IV. Evaluation of the Proposed Approach

In this section, we evaluate the performance of our proposed hierarchical scheduling approach based on different network topologies and various traffic flows.

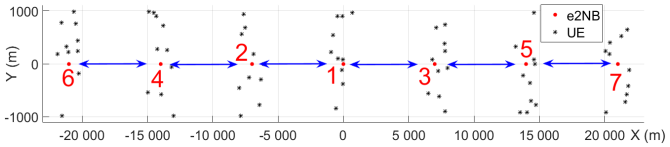
A. Simulation environment

A complete LTE simulator is developed in MATLAB allowing to create a 2D-map of an arbitrary network of e2NBs with their associated UEs and to generate arbitrary flows between nodes (e.g. U2U, N2U, N2N traffic). To model the processing time for each incoming packet at e2NB, we assume that it takes 5ms to finish all processing before pushing it to queue (DL or UL) for relaying to next hop.

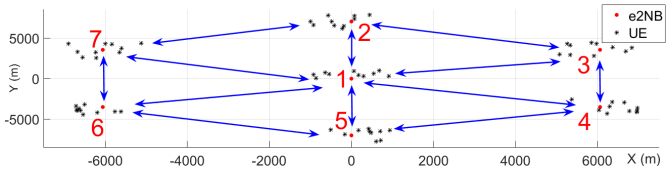
1) Simulation parameters: Our simulation parameters applied to UEs, eNBs are mostly taken from 3GPP documents (TR36.814, TR36.942, TR25.942) with each e2NB operates in FDD, SISO mode. To characterize the in-band characteristic, we use the same carrier frequency (2.1GHz, band 4) through the network with 10MHz radio bandwidth. Further, to evaluate the interference impact, we do not assume any applied interference cancellation scheme and use the omni-directional

antenna at both eNB and UE. $offset$ for L_{SuF} computation in Eq. (1) is set to 1 and P_{SuF} is set to 400ms. Finally, the simulations are performed for a duration of 10000 SFs.

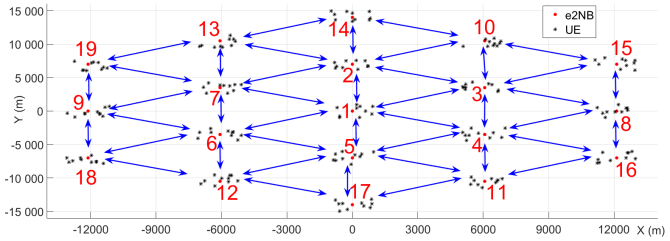
2) Network topology: Here, we consider three different network topologies as shown in Fig. 4(a), 4(b) and 4(c). In each topology, all e2NBs have 10 attached UEs and are connected to adjacent e2NBs as indicated by the bi-directional arrows as shown in Fig. 3.



(a) Line topology with 7 e2NBs and 70 UEs.



(b) Hexagonal topology with 7 e2NBs and 70 UEs.



(c) Hexagonal topology with 19 e2NBs and 190 UEs.

Fig. 4: Considered network topologies

3) Traffic patterns: For real-time traffic, we randomly pair all UEs to establish bi-directional VoIP calls with 20 bytes packet size (40 bytes on L1) and 20ms arrival rate. For QoS requirement of the real-time traffic, we use the maximum one-way-delay of 150ms for 95-percentile of the packet to ensure a quality call with a Mean Opinion Score (MOS) of 3.5 using a G.729 codec in [15]. Whereas the elastic traffic is set between BSs to represent the inter-site data transfers that often happen in military and public safety scenarios. Each elastic traffic is served in best effort to maximize its bandwidth and behaves as a connection-oriented acknowledgment service in that a new packet will be generated only if the maximum number of non-acknowledged packets is not reached.

B. Considered Algorithms

In our previous work [16], we compared a light version of hierarchical approach to a legacy link scheduling algorithm for mesh networks and the results showed our approach to be superior to the legacy one. Hence, in this section, we aim to compare three realistically implementable variant algorithms of our proposed approach:

1) A baseline algorithm which is unaware of the required SF of all traffic flows, i.e., $SF_{rt} = SF_e^D = SF_e^U = 1$, denoted as *Basic Alg.*.
2) A simplified algorithm which does not leverage FDD characteristic when computing required SFs for real-time traffic in Algorithm 2 denoted as *DL Alg.*
3) The full algorithm proposed in this work denoted as *UL Alg.*

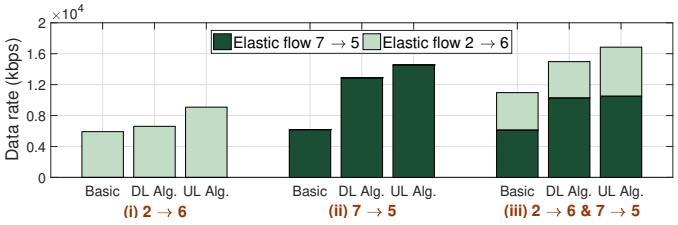
Further, to avoid the impact of a specific flow control policy on the total number of traffic flows, we increase the superframe duration (i.e. L_{SuF}) by 10 SFs when dissatisfaction happens (i.e. $rtDisSat == 1$ in Alg. 1) rather than rejecting/removing any real-time flow.

C. Simulation Results

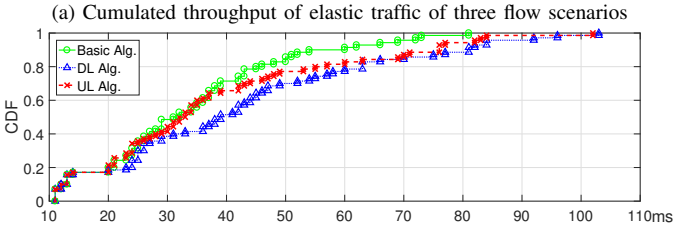
Based on the aforementioned three network topologies in Fig. 4, we compare different performance metrics of the two types of traffic.

- **Real-time traffic**: Satisfaction ratio in terms of the 95-percentile point of per-flow latency
- **Elastic traffic**: Cumulated throughput of all elastic flows

1) Line topology with 7 e2NBs: Besides the randomly-paired VoIP traffic, three different flow scenarios are evaluated for elastic traffic: (i) from $e2NB_2$ to $e2NB_6$ ($2 \rightarrow 6$), (ii) from $e2NB_7$ to $e2NB_5$ ($7 \rightarrow 5$) and (iii) both aforementioned two elastic flows ($2 \rightarrow 6$ & $7 \rightarrow 5$). In Fig. 5.(a), we can observe that both *DL Alg.* and *UL Alg.* provide higher throughput of elastic flows than *Basic Alg.* among all three flow scenarios. Moreover, *UL Alg.* performs slightly better than *DL Alg.* since it exploits full FDD capability. In Fig. 5.(b), we show the performance metric of real-time flows under the two elastic flow scenarios (i.e. $2 \rightarrow 6$ & $7 \rightarrow 5$) in terms of the CDF plot of 95-percentile packet delay among all real-time flows. We can observe that all real-time flows can be satisfied since the maximum 95-percentile delay is about 110ms which is less than 150ms, i.e. satisfaction ratio is 100%.



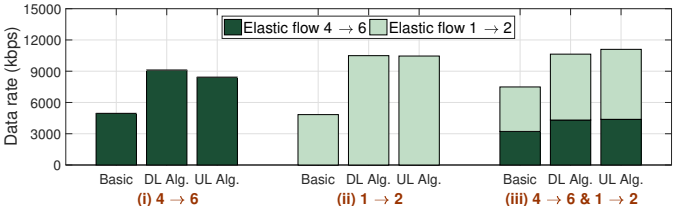
(a) Cumulated throughput of elastic traffic of three flow scenarios



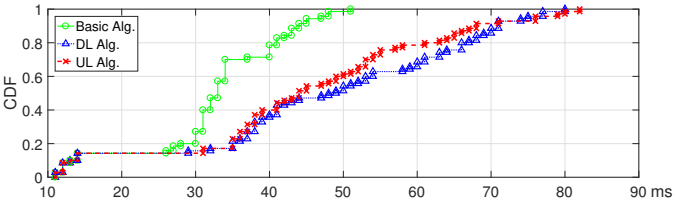
(b) CDF plot of the 95-th percentile of packet latency among real-time flows

Fig. 5: Line topology with 7 e2NBs and 70 UEs

2) Hexagonal topology with 7 e2NBs: Like the line topology, we explore three flow scenarios for elastic traffics: (i) from $e2NB_4$ to $e2NB_6$ ($4 \rightarrow 6$), (ii) from $e2NB_1$ to $e2NB_2$ ($1 \rightarrow 2$) and (iii) both aforementioned two elastic flows ($4 \rightarrow 6$ & $1 \rightarrow 2$). It can be observed in Fig. 6.(a) that again, both *DL Alg.* and *UL Alg.* show higher elastic flow throughput than *Basic Alg.*. However, the results of *DL Alg.* and *UL Alg.* are close among these three flow scenarios. Then, Fig. 6.(b) shows the performance metric of real-time traffic flow under the case



(a) Throughput of elastic traffic of three flow scenarios

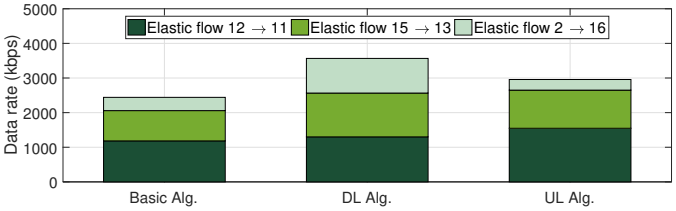


(b) CDF plot of the 95-th percentile of packet latency among real-time flows

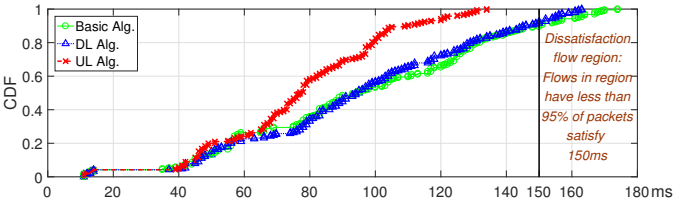
Fig. 6: Hexagonal topology with 7 e2NBs and 70 UEs.

with two elastic flow (i.e. $4 \rightarrow 6$ & $1 \rightarrow 2$). It can be observed that all real-time flows can satisfy 150ms requirement (i.e. satisfaction ratio is 100%). Nevertheless, we can see that the *Basic Alg.* performs better than the others since the maximum 95-percentile delay is around 50ms while it is 80ms for *DL Alg.* and *UL Alg.*. This is because of the trade-off of boosting the throughput of elastic traffic shown in Fig. 6.(a).

3) Hexagonal topology with 19 e2NBs: Lastly, we explore a scenario on the hexagonal topology with 19 e2NBs and we consider the case with 3 concurrent elastic flows and 95 random-paired VoIP flows: one flow from $e2NB_{12}$ to $e2NB_{11}$ ($12 \rightarrow 11$), one flow from $e2NB_{15}$ to $e2NB_{13}$ ($15 \rightarrow 13$) and one from $e2NB_2$ to $e2NB_{16}$ ($2 \rightarrow 16$). It can be observed in Fig. 7.(a) that the *DL Alg.* outperforms the other two algorithms in the cumulated throughput of all elastic traffics when compared with the ones of *Basic Alg.* and *UL Alg.*. However, in Fig. 7.(b), *UL Alg.* is the only approach that can respect the latency requirement for all VoIP flows over the network; that is to say, the satisfaction ratio is 100% whereas the satisfaction ratio is only 90% for both *Basic Alg.* and *DL Alg.*. This is because these two approaches are not able to allocate the required real-time SFs (i.e. SF_{rt}) based on the computed superframe duration L_{SuF} in Eq. (1). Hence, it leads to increase the duration of superframe that introduces



(a) Cumulated throughput of elastic traffics



(b) CDF plot of the 95-th percentile of packet latency among real-time flows

Fig. 7: Hexagonal topology with 19 e2NBs and 190 UEs

extra latency for VoIP flows, while at the same time allowing for better cumulated throughput for elastic flows for *DL Alg.*

To conclude, the full version of our proposed approach (*UL Alg.*) achieves the best trade-off between satisfying the latency requirements for real-time flows and providing the largest throughput for elastic flows of all considered topologies. Such superiority bases on the novelty to leverage FDD property and efficient frequency reuse with interference avoidance.

V. Conclusion

In this work, we firstly leverage the *Un* relay interface and the e2NB architecture to enable an in-band self-backhauling LTE mesh network. To efficiently allocate resources across e2NBs over both access/backhaul links, we propose a low complexity and realistically implementable hierarchical approach to deal with the original cross-layer scheduling problem in different time-scales. Such hierarchical approach can bring autonomy into the network via COEs within each e2NB and is compatible with the legacy scheduler at access link. Finally, the simulation results over different network topologies and various traffic flows show that taking into account FDD property is beneficial and that the proposed approach not only meets specific QoS requirements for real-time traffic but also provides high throughput for elastic traffic. In future, we plan to provide a complete solution of BS architecture to enable autonomous self-backhaul LTE mesh network in every detail.

References

[1] ITU-R, "IMT vision - framework and overall objectives of the future development of IMT for 2020 and beyond," Tech. Rep.

[2] R. Favraud *et al.*, "Toward moving public safety networks," *IEEE Communications Magazine*, vol. 54, no. 3, pp. 14–20, March 2016.

[3] R. Favraud and N. Nikaein, "Wireless mesh backhauling for LTE/LTE-A networks," in *IEEE MILCOM 2015*, Oct 2015, pp. 695–700.

[4] Y. Sui *et al.*, "Moving cells: a promising solution to boost performance for vehicular users," *IEEE Communications Magazine*, vol. 51, no. 6, pp. 62–68, 2013.

[5] R. Favraud and N. Nikaein, "Analysis of LTE relay interface for self-backhauling in LTE mesh networks," in *IEEE VTC-fall*, September 2017.

[6] O. Teyeb *et al.*, "Dynamic relaying in 3GPP LTE-Advanced networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2009, no. 1, 2009.

[7] C. Hoymann *et al.*, "Relaying operation in 3GPP LTE: challenges and solutions," *IEEE Communications Magazine*, vol. 50, no. 2, 2012.

[8] Y. Yuan, *LTE-Advanced Relay Technology and Standardization*. Springer-Verlag Berlin Heidelberg, 2013.

[9] A. S. Mohamed *et al.*, "Self-organized dynamic FFR resource allocation scheme for LTE-Advanced relay based networks," *Wireless Personal Communications*, vol. 91, no. 2, pp. 933–955, 2016.

[10] J. Han and H. Wang, "Uplink performance evaluation of wireless self-backhauling relay in LTE-Advanced," in *WiCOM 2010*, Sept 2010.

[11] P. H. Pathak and R. Dutta, "A survey of network design problems and joint design approaches in wireless mesh networks," *IEEE Communications Surveys Tutorials*, vol. 13, no. 3, pp. 396–428, Third 2011.

[12] X. Foukas *et al.*, "FlexRAN: A flexible and programmable platform for software-defined radio access networks," in *ACM CoNEXT*, 2016.

[13] Nokia, "LTE networks for public safety services," White Paper, 2014.

[14] I. F. Akyildiz *et al.*, "Wireless mesh networks: A survey," *Comput. Netw. ISDN Syst.*, vol. 47, no. 4, pp. 445–487, Mar. 2005.

[15] A. F. Ribadeneira, "An analysis of the MOS under conditions of delay, jitter and packet loss and an analysis of the impact of introducing piggybacking and reed solomon FEC for VoIP," Master's thesis, Georgia State University, 2007.

[16] R. Favraud *et al.*, "QoS guarantee in self-backhauled LTE mesh networks," in *IEEE GLOBECOM*, 2017.