Design of Flexible Fronthaul Featuring Per-UE Granularity and RU-level Puncturing for URLLC Applications

Yahya Alfadhli[1], Shuang Yao[1], Muhammad S. Omar[1], Shang-Jen Su[1], Shuyi Shen[1], Rui Zhang[1], You-Wei Chen[1], Peng-Chun Peng[2] and Gee-Kung Chang[1]

[1]*School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA*

[2]*Department of Electro-Optical Engineering, National Taipei University of Technology, Taipei 10608, Taiwan*

yalfadhli@gatech.edu

Abstract: We propose and experimentally verify a fine-grained, Per-UE, flexible fronthaul where different applications are transported over different function splits (*i.e.,* URLLC over A-RoF-based fronthaul, Option-9, and other traffic over Option-7), exploiting two RU-level puncturing methods.

OCIS codes: (060.4256) Networks, network optimization; (060.4258) Networks, network topology.

1. Introduction

The concept of flexible function split, that concerns dynamic assignment of functions in radio access networks (RAN), has been investigated heavily in literature [1]. There are different granularities of flexibility, discussed in [2], including flexibility per central unit (CU), per distributed unit (DU) and per user equipment (UE). However, Fig. 1(a) depicts two levels of flexibility: (*i*) Per-remote unit (RU) flexibility, wherein the RU can support different splits at different times but only one is activated at any point of time [3, 4]. This kind of flexibility can facilitate dynamic placement of processing loads in the network, energy saving at the RU and adaptation to varying fronthaul bandwidth and latency capabilities. A common use-case is to run Option-7 under normal fronthaul traffic conditions. Then, once the network is getting congested, the flexible network will switch to a less bandwidth-demanding function split (FS) such as Option-6; (*ii*) Per-UE flexibility, wherein the RU can support several FSs simultaneously. Each of these splits is dedicated to a specific envisioned 5G application category, such as enhanced mobile broadband (eMBB), ultra-reliable low latency (URLLC) and massive machine type communications (mMTC). While the former flexibility type has been searched in literature, the latter lacks published research results, particularly experimental work.

To reduce the scheduling latency for URLLC, eMBB traffic is punctured by URLLC. Conventionally, this multiplexing procedure is performed by MAC layer at DU. However, the major pitfall is that, after multiplexing, all traffic (*i.e.,* URLLC and others) will traverse the same fronthaul without any quality of service (QoS) awareness. For example, in Option-7.1 (wherein the FFT stage is located at the RU), the frequency domain IQ samples should be synchronously delivered to RU. As this scenario is suboptimal for URLLC, we are proposing a new scheme wherein multiplexing of applications occurs at the RU instead of DU (*i.e.,* RU-level puncturing). Once RU-level multiplexing is realized, different QoS measures can be applied such as the use of different FS options for different applications. FS such as Option-7 induces additional latency and jitter due to additional fronthaul-related processing including compression, packetizing and queuing. Therefore, to maintain low latency fronthauling of URLLC traffic, we make use of Option-9 FS, which has been experimentally verified, in previous work, to exhibit lower fronthaul latency than other possible FSs [5]. Option-9 is based on analog radio over fiber (A-RoF) technology wherein all PHY processing, including RF layer, is consolidated at the DU, and RU only performs optical/electrical conversion. For all other applications categories, Option-7 is a widely accepted FS as it lowers the fronthaul bandwidth requirement and keeps the RU relatively simple. In this paper, we propose and experimentally demonstrate a Per-UE flexible FS fronthaul design using two novel RU-level puncturing techniques, namely, coordinated and uncoordinated puncturing methods.

2. Design and experimental setup of coordinated RU-level puncturing

The main challenge in designing RU-level puncturing is that the MAC layer, which is responsible for URLLC/eMBB multiplexing, resides in the DU for the most foreseen deployment scenarios. In this section, we discuss the coordinated

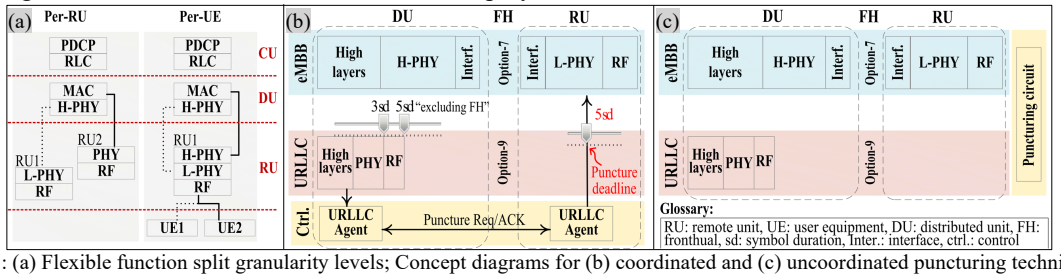

Fig.1: (a) Flexible function split granularity levels; Concept diagrams for (b) coordinated and (c) uncoordinated puncturing techniques.

puncturing, which relies on exchanging control messages between DU and RU, as depicted in Fig. 1(b). As discussed earlier, the URLLC is sent over Option-9 and other traffic is carried over Option-7. At the DU, eMBB and URLLC network slices are identified by the MAC layer. Upon URLLC presence, MAC layer will send the puncturing metadata snippet to the URLLC agent including indices for the targeted resource elements for puncturing. Then, the puncturing agent will generate a puncturing request packet that contains all necessary information for a successful puncturing such as current time stamp, the targeted resource elements and the puncture deadline. In order to have a successful puncturing at RU, the puncture deadline should not be violated. The puncture deadline can be written as:

$$Puncture\ deadline = URLLC\ scheduling + DU_{URLLC\_Proc.} + DU_{URLLC\_Align.} + DU_{URLLC\_TTI} \quad (1)$$

where $URLLC\ scheduling$ is time consumed to schedule the corresponding mini-slot, which depends on the amount of ingress URLLC traffic. The values of the three remaining components depend on the adopted numerologies including subcarrier spacing (SCS) and mini-slot size, which in our case are 15 kHz and 2-symbols duration (sd), respectively. Consequently, the processing delay at DU ($DU_{URLLC\_Proc.}$) is 2-sd, the alignment delay ($DU_{URLLC\_Align.}$) is 1-sd, and the mini-slot transmission time interval (DU_{URLLC_TTI}) is 2-sd. All these delay components contribute 350 us to the 500 us one-way latency limit, which leaves 150 us for UE processing [6]. The fronthaul latency should be also accommodated within this 500 us latency budget, and particularly, it is deducted from $DU_{URLLC\_Proc.}$ budget.

The experimental setup is depicted in Fig. 2(a) where Open Air Interface (OAI) is used to implement the LTE stack including RAN and core network [7]. At the DU, the output of H-PHY layer is split into two streams, one fed to the interface (for Option-7.1) and another fed to L-PHY (for Option-9). In order to exploit the reduced latency of Option-9, both FSs should not be synchronized. However, in our setup, a GPS-based clock source (Clk) is used as a master clock merely to obtain proper puncturing timing alignment. The fact that Clk is connected to the RF layer, makes the Clk acquisition a time-consuming step (avg. 60 us). However, in real implementation, the already existing clock source that is required for Option-7 operation can be used, which can reduce this delay component to few microseconds.

Once the URLLC agent at the RU receives the puncturing request, it triggers the puncturing process and sends a puncturing ACK to the DU agent. Figure 2(b) shows the Wireshark snapshot of puncturing messaging between URLLC agents at DU and RU where the delay is about 210 us. On the other hand, Fig. 2(c) presents the one-way end-to-end latency of URLLC puncturing for different fronthaul fiber lengths (back-to-back, 500 m and 1 km). The results show that only few counts of puncturing request outliers miss the puncturing deadline which is 350 us in this scenario. However, if the clock acquisition delay is reduced as discussed earlier, all of the puncturing requests would meet the deadline. Given that puncturing request deadline is met, puncturing can be easily implemented prior to FFT stage which provides higher fine-grained puncturing control. Figure 2(d) shows three punctured eMBB signals when: (*i*) all subcarriers are punctured for the mini-slot duration, (*ii*) selected few subcarriers are punctured during the whole frame duration, and (*iii*) specific resource elements are punctured.

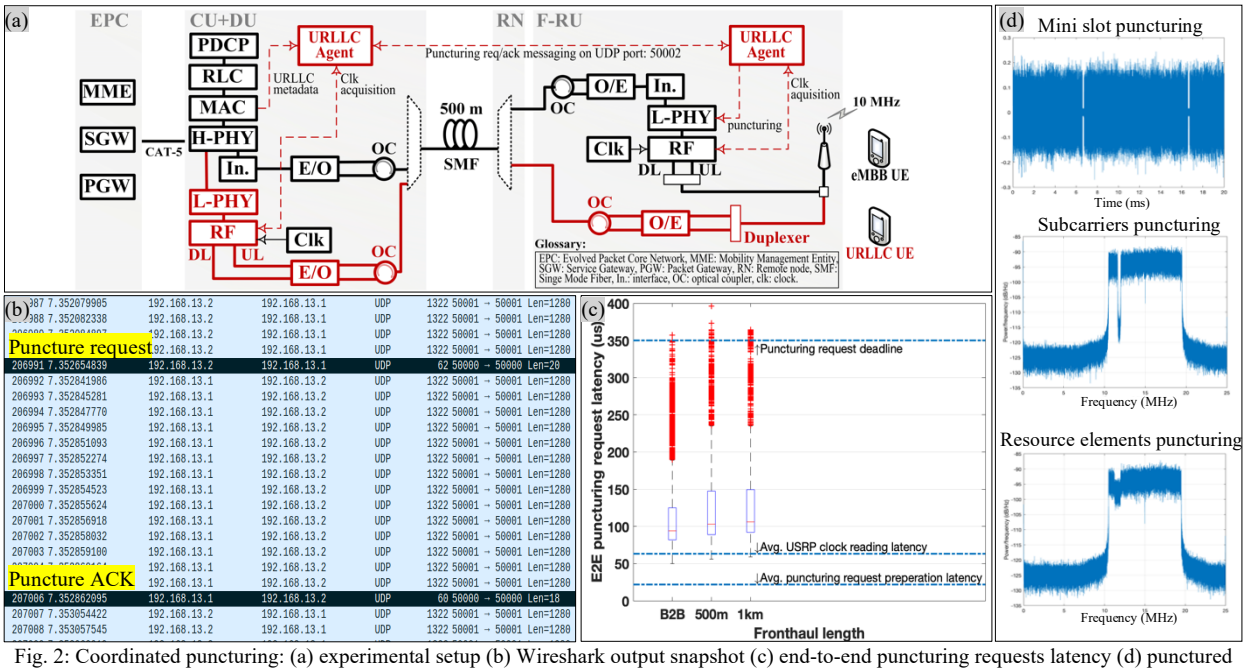

Fig. 2: Coordinated puncturing: (a) experimental setup (b) Wireshark output snapshot (c) end-to-end puncturing requests latency (d) punctured eMBB signal under different puncturing configurations

3. Design and experimental setup of uncoordinated RU-level puncturing

Even though coordinated puncturing method enables fine-grained puncturing of specific resource elements, the main limitation is the limited puncturing budget. Therefore, depending on the network implementation scenarios, some mini-slot durations (*e.g.,* 2-sd at 120 kHz SCS) cannot be realized with the coordinated puncturing method, as their puncturing deadline cannot be met. Therefore, we present another method where RU-level puncturing is performed without any coordination between DU and RU, as shown in Fig. 1(c). In this puncturing method, RU has a puncturing circuit that is responsible for detecting the presence of URLLC traffic and performing the puncturing accordingly.

The experimental setup is shown in Fig. 3(a), wherein two independent application flows are implemented. The eMBB flow is presented by a pair of National Instrument PXI equipment that is implemented on FPGA following LTE standard for a 20 MHz signal with 15 kHz SCS. On the other hand, the URLLC traffic is presented by an OAI LTE signal where the transmitter side is the OAI DU and the receiver is a commercial off-the-shelf phone. Under the unoccupied LTE signal condition, the LTE waveform contains reference symbols that are around 71 us wide (which resembles the duration of a 4-sd mini-slot at 60 kHz SCS). The puncturing circuit includes an envelope detector (ED) that takes the URLLC signal as input and detects the presence of the URLLC traffic. Then, the ED signal is connected to a comparator to generate a control ON/OFF signal that will be controlling the RF switch.

All waveforms generated by the puncturing circuit are shown in Fig. 3(b) with the corresponding yellow-shaded labeling. The inset figure shows the fast response of the puncturing circuit (in few nanoseconds range), which proves the capability to support even the smallest mini-slot configuration defined in 5G standard (18 us). Figure 3(c) illustrates the waveforms of the URLLC, punctured eMBB and the resultant mixed applications signal that is transmitted over the air. To study the impact of the puncturing mechanism, *ping* test is used through URLLC traffic. As Fig. 3(d) shows, the *ping* test results are not impacted by the puncturing process (*i.e.,* URLLC latency performance is not degraded by the puncturing process), which proves the efficiency of the uncoordinated puncturing method. Three sets of *ping* tests, where each has 1000 packets, are used for this test. The packet loss rate is equal in both cases (1/3000), which proves that the puncturing process does not impact the reliability of URLLC traffic. The constellation diagrams for eMBB traffic in the cases of standalone and mixed traffic flows are shown in Fig. 3(e).

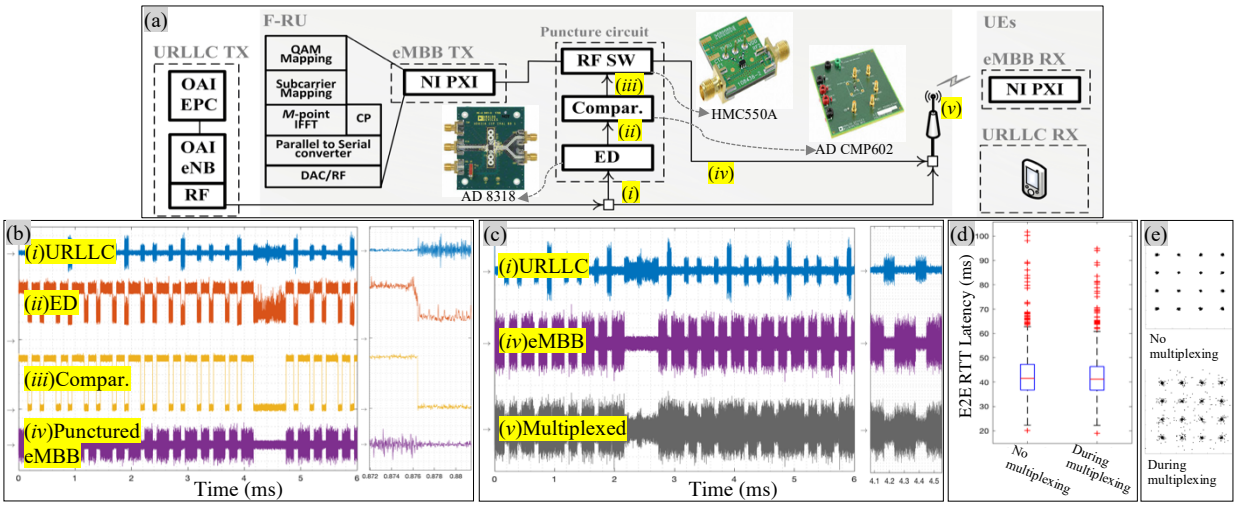

Fig. 3: Coordinated puncturing (a) experimental setup, (b) Waveforms at puncturing circuit, (c) waveforms of URLLC and eMBB before and after multiplexing, (d) impact of puncturing on URLLC ping results, and (e) impact of puncturing on eMBB signal constellation

4. Conclusion

We present and experimentally verify two new methods enabling service differentiation at the fronthaul where URLLC is carried over an A-RoF-based fronthaul (Option-9) and other traffic is transported using Option-7. Both methods use RU-level puncturing to enable multiplexing of downlink URLLC data with other traffic at the RU instead of DU. The results show that the coordinated puncturing achieves fine-grained puncturing of resource elements, whereas uncoordinated puncturing has a nanosecond-scale response, which makes it able to support all 5G mini-slots durations.

References

[1] L. M. Larsen, et al., IEEE Commun Surv Tutorials (2018).
[2] Y. Yoshida, OFC (2018), pp. 1–85.
[3] Y. M. Alfadhli, et al., OFC (2018), pp. Th2A–48.
[4] A. M. Alba, et al., IEEE INFOCOM WKSHPS-3rd (2019).
[5] Y. Alfadhli, et al., Computer Networks Vol. **162**, 106865 (2019).
[6] R1-1608844, "Support of URLLC in DL," 3GPP TSG RAN (2016).
[7] [online] https://www.openairinterface.org, accessed Oct (2019).