On enabling 5G Dynamic TDD by leveraging Deep Reinforcement Learning and O-RAN

Karim Boutiba*, Miloud Bagaa†, and Adlen Ksentini*

* EURECOM, France. firstname.lastname@eurecom.fr

† CSC-IT Center for Science Ltd, Espoo, Finland. miloud.bagaa@csc.fi

***Abstract*—Dynamic Time Duplex Division (D-TDD) is a promising solution to accommodate the new emerging 5G and 6G services characterised by asymmetric and dynamic Uplink (UL) and Downlink (DL) traffic demands. D-TDD dynamically changes the TDD configuration of a cell without interrupting users' connectivity, hence balancing the bandwidth for UL or DL communication according to the traffic pattern. However, 3GPP standard does not specify algorithms or solutions to derive the TDD configuration, i.e., the number of slots to dedicate to UL and DL. In [1], we have proposed a Machine Learning (ML)-based solution relaying on Deep Reinforcement Learning (DRL) to allow the base station (or gNB) to self-adapt to the traffic pattern of the cell by periodically adapting the number of slots dedicated to UL and DL. In this work, we implemented the DRL algorithm on top of an open-source gNB based on OpenAirInterface (OAI) [2] to demonstrate its efficiency. To this end, we relied on the O-RAN architecture [3], where the proposed DRL algorithm is deployed as xApp at the Near Real-time RAN Intelligent Controller (RIC) and communicates with the base station using O-RAN E2 interface. We developed xTDD Service Model (SM) following the E2SM standard [3], allowing the DRL solution to monitor DL and UL buffers from the gNB to deduce the optimal TDD configuration that accommodates the current traffic. Then, the decision (i.e., TDD configuration) is pushed to the base station. We implemented the solution on top of the OAI 5G StandAlone (SA) platform and Flexric RIC [4]. To the best of our knowledge, this is the first demonstration of a ML-based D-TDD on top of a real 5G network, showing the advantage of O-RAN architecture to building Self Organized Network (SON) function for dynamic configuration of D-TDD.**

I. INTRODUCTION

In recent years, the small cell deployment scenario has been experiencing increasing growth, especially for industry 4.0 use cases and 5G private networks. While 4G networks were designed to accommodate DL-dominant traffic, 5G networks need to adapt to the newly emerging services, such as high-quality video streaming captured by drones for building surveillance that requires more UL traffic than DL. In this context, Dynamic TDD (D-TDD) is a promising solution introduced in 5G to satisfy the dynamic traffic pattern of small cell deployment. In D-TDD the allocation of the DL/UL ratio (i.e., the number of DL and UL slots in a frame) can be dynamically adjusted according to the UL and DL traffic demands. D-TDD allows the base station to change the TDD pattern dynamically without interrupting users' connectivity. Introduced flexibility by D-TDD allows the base station to adapt the frame configuration according to the traffic pattern by selecting the number of slots dedicated to UL and DL. However, the 5G NR specifications only cover the mechanism allowing the base station to inform the UE about the UL/DL slots pattern in a TDD frame, leaving the algorithm deriving the pattern UL/DL open. In [1], we have filled this gap by proposing a novel algorithm, namely, Deep Reinforcement Learning (DRL)-based 5G RAN TDD Pattern (DRP), which allows deriving the UL/DL pattern of TDD frames according to the existing cell traffic whatever it is DL or UL dominant. DRP monitors the DL and UL traffic and derives the percentage of the frame (number of slots) dedicated to UL and DL, aiming to avoid the overflow of DL and UL buffers to guarantee the optimal quality of service (QoS) [5] whatever the pattern of traffic, UL or DL dominant. In this demo, we leveraged O-RAN architecture [6] by executing the DRL inference at the near RT RIC as a xApp, while the dynamic TDD mechanism is implemented in OAI. xTDD Service Model (SM) is introduced to plug DRP in OAI following the E2SM standard [3]. xTDD xApp receives UL and DL buffer fullness ratio from the base station via E2 indication messages periodically and executes DRP to derive the TDD pattern. Finally, xTDD sends the TDD pattern via E2 control message to the base station. The latter will update the cell configuration allowing to ensure SON function for D-TDD.

II. SYSTEM DESIGN AND IMPLEMENTATION

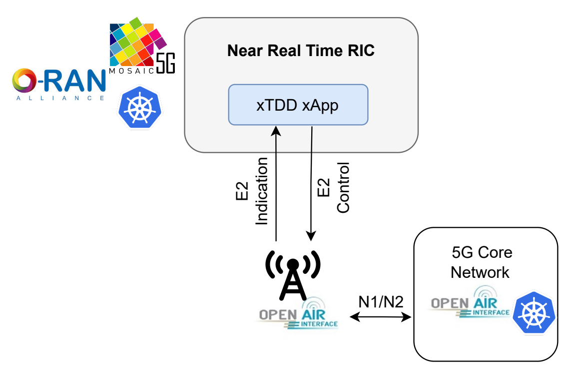


Figure 1: O-RAN aligned architecture of xTDD

In this demo, we improved the design introduced in [1] by aligning it with O-RAN architecture as depicted in Figure 1. xTDD xApp executes the DRP agent, responsible for deriving the TDD pattern. xTDD receives E2 indication messages each T ms from the base station containing the DL and UL buffer status. The DL buffer status is extracted from the RLC layer by summing the amount of remaining data (in bytes) after each scheduling process over all the Logical Channels (LC). In contrast, the UL buffer status is estimated at the MAC layer by summing the Buffer Status Report (BSR) MAC Control Element (CE) received over all the LCs. xTDD derives the TDD pattern taking buffer status history as input, and sends a E2 control message containing the ratio of UL slots. The xTDD SM at the base station updates the TDD configuration of the cell at the first slot of the next TDD period upon the reception of the E2 control message.

At the gNB side, we have implemented D-TDD in OAI by: (i) removing the TDD pattern information from the periodic SIB1 broadcast to UEs, which leads the latter to figure out the direction of the slot dynamically via the Downlink Control Information (DCI); (ii) changing the MAC and PHY layers context to change the TDD pattern at gNB. (iii) sending multiple DCI with different K2 parameters in order to schedule multiple UL slots in the same DL slot (to enable more UL slots than DL slots).

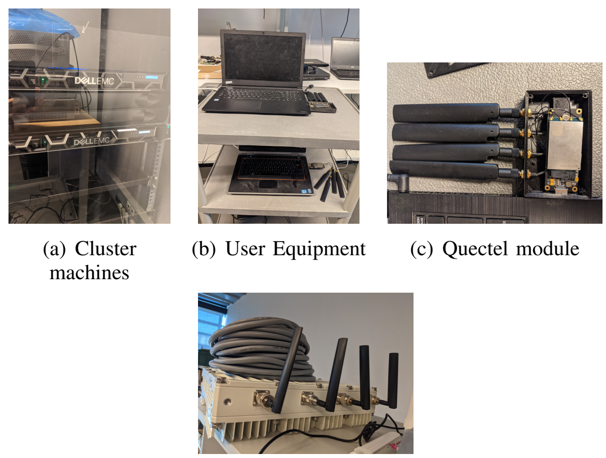
At the RIC side, we have implemented DRP using the Deep Deterministic Policy Gradient (DDPG) Algorithm [7]. The DRL hides the complexity of the environment, which helps DRP to make efficient and quick decisions that adapt according to traffic patterns. We define the DRP design by:
State: The DRP agent considers $\mathcal{K}$ previous observations before taking any action. Each observation depicts the buffer fullness ratio of both DL and UL.
Action: The DRP agent has only one continuous action a_t that presents the percentage of slots that should be reserved for the UL traffic.
Reward: The DRP agent receives a positive reward when the buffers do not exceed their threshold (i.e., the maximum size of the buffers). Moreover, the emptiest the buffers are, the highest reward becomes. The agent receives a penalty when one of the buffers exceeds its capacity. This strategy will force the DRP agent to empty all buffers (DL and UL) as much as possible and prevent their overflow, positively impacting the QoS.

We leveraged Flexric SDK [4] to create the xTDD SM and xApp and integrate it with Flexric RIC. xTDD xApp gains the ability to learn with time and adapts to different and unseen situations. We have designed xTDD to be lightweight to ensure real-time interaction with OAI. Also, we have designed xTDD to ensure generality and then work in an unseen environment. xTDD has been designed in a way to work independently from the number of slots (which makes xTDD suitable for multiple numerologies) and the number of UEs. Further, it considers the variation and correlation in the buffer states to predict traffic patterns.

III. DEMONSTRATION

A. Equipment and Settings

Our setup, depicted in Figure 2, is composed of two machines with 36 CPUs. Each CPU is an Intel(R) Xeon(R) Gold 6154 CPU @ 3.00GHz. One machine is used to run gNB based on OAI. The latter is connected to AW2S Radio Unit (RU) [8]. The second machine is used as a single-node cluster based on Kubernetes. It hosts the 5G Core Network based on OAI and the Flexric RIC. (ii) two laptops with Ubuntu Operating System (OS), each connected to a Quectel RM500Q-GL module [9], considered 5G UEs. At the gNB side, we are using a 20 MHz bandwidth at the frequency of 3.5 GHz (band n78) in an indoor environment. We are using a static Aggregation Level (AL) that allows us to schedule up to 8 DCIs (for both UL and DL) per DL slot. We are fixing the maximum Modulation and Coding Scheme (MCS): (i) for DL: 28 (ii) for UL: 20. We reduced the UL MCS due to a limitation in OAI implementation at the time of writing this paper.



(a) Cluster machines (b) User Equipment (c) Quectel module

(d) AW2S Radio unit

Figure 2: Dynamic TDD setup hardware

B. Experiment Scenario

We connect two UEs to the 5G network. The base station uses numerology 1 and a TDD period of 5ms. For instance, a TDD period has 10 slots. DRP, leveraging DRL, will automatically and periodically allocate these 10 slots to UL or DL, i.e., how many slots are dedicated to UL, and the rest are dedicated to DL. In this experiment, we execute three different scenarios:

- **UL dominant traffic:** We run the Iperf tool [10] as a client on the UE side, requesting 28 Mbps (i.e., the maximum achievable UL rate on the current 5G configuration) or on two UEs requesting 14 Mbps for each UE. For DRP, both cases are identical because xTDD aggregates the buffer sizes of all UEs to obtain the overall cell traffic size. The server side of Iperf was, run on the single node cluster in the same subnet as the User Plane Function (UPF) N6 interface (the interface that connects the UPF to the data network). We used the User Datagram Protocol (UDP) to have less DL traffic (i.e., no ACK messages). The initial TDD pattern is (5DL, 1M, 4UL). A Mixed slot (M) is composed of DL symbols, guard-band symbols (i.e., unused symbols), and UL symbols. The Mixed slot is required between UL and DL slots to provide enough time for DL delayed signal to arrive due to path distance and also gives enough opportunity for UE to receive UL timing advance command from the gNB [11]. When DRL observes that the UL buffer fullness ratio is higher than DL, it modifies the TDD pattern by allocating more UL slots. We observe that DRL allocates the TDD pattern (2DL, 1M, 7UL). The three DL slots are needed to send DCIs for UL transmission scheduling. Therefore, the Iperf server receives all the data sent by the clients (i.e., 28 Mbps).

- **DL dominant traffic:** In this scenario, we are using the same network configuration as the UL-dominant traffic scenario. The only difference is the location of the Iperf client and server. We use two Iperf servers on the UE side and two Iperf clients on the cluster side. The clients request 40 Mbps (i.e., the maximum achievable DL throughput on the current 5G configuration). We notice that the DRP changes the TDD pattern to (7DL, 1M, 2UL). We argue this by the fact that DRP notices that the DL fullness ratio becomes more important than the UL fullness ratio. Three UL slots are required to send the Hybrid Automatic Repeat-Request (HARQ) information to acknowledge DL transmissions in each slot since the gNB uses the PUCCH 0 format (i.e., 2 bits are available in each UL slot, where each bit is used to acknowledge a DL transmission). Therefore, the Iperf server on the UE side receives the entire data sent by the cluster clients (i.e., 40 Mbps).
- **UL/DL equilibrate traffic:** In this scenario, we run an Iperf client on one UE and an Iperf server on the second UE. An Iperf client is deployed in the single node cluster to send data to the second UE, while an Iperf server is deployed on the same cluster to receive data from the first UE. Both clients send data at a rate of 20 Mbps. The DRP agent observes that the UL buffers are full and therefore allocates the TDD pattern (2DL, 1M, 7UL) to avoid UL buffer overflow. Then, the UL buffer fullness ratio will decrease, and at the same time, the DL buffer fullness ratio will increase. Hence, DRP will allocate more DL slots (i.e., 7DL, 1M, 2UL). When the UL and DL buffers exceed 0.5 (i.e., half the buffer is full), the DRP will allocate the TDD pattern (5DL, 1M, 2UL). Therefore, the DRP can keep the DL and UL buffers of the gNBs/UEs as small as possible and avoid buffer overflow.

We summarize the results of the three scenarios as follows: (i) UL traffic only: we observe that xTDD selects more UL slots; (ii) DL traffic only: we observe that xTDD selects more DL slots; (iii) UL and DL traffic: we observe that the base station changes the pattern dynamically to avoid buffers overflow and to satisfy the requested data rate from Iperf clients.

CONCLUSION

In this demo, we implemented DRP, a Deep Learning Reinforcement (DRL)-based solution that allows to derive and adjust the TDD pattern in 5G NR. We leveraged OpenAirInterface (OAI) and O-RAN to build a Self Organized Network (SON) solution for dynamic configuration of TDD. The proposed demo clearly showed that DRP is able avoid buffers overflow and dynamically adapt the TDD pattern according to the cell traffic.

ACKNOWLEDGMENT

This work was partially supported by the European Union's Horizon Program under 6GBricks project (Grant No. 101096954).

REFERENCES

[1] Miloud Bagaa, Karim Boutiba, and Adlen Ksentini. "On using Deep Reinforcement Learning to dynamically derive 5G New Radio TDD pattern". In: *GLOBECOM 2021*. 2021.

[2] Florian Kaltenberger et al. "The OpenAirInterface 5G new radio implementation: Current status and roadmap". In: *WSA 2019, 23rd ITG Workshop on Smart Antennas, Demo Session, 24-26 April 2019, Vienna, Austria*. 2019.

[3] Open RAN alliance. *O-RAN specifications*. 2022. URL: https://www.o-ran.org/specifications. (accessed: 08.08.2022).

[4] Robert Schmidt, Mikel Irazabal, and Navid Nikaein. "FlexRIC: An SDK for next-Generation SD-RANs". In: *Proceedings of the 17th International Conference on Emerging Networking EXperiments and Technologies*. CoNEXT '21. Virtual Event, Germany: Association for Computing Machinery, 2021, pp. 411–425. ISBN: 9781450390989. DOI: 10.1145/3485983.3494870. URL: https://doi.org/10.1145/3485983.3494870.

[5] Tarik Taleb and Adlen Ksentini. "QoS/QoE predictions-based admission control for femto communications". In: *Proceedings of IEEE International Conference on Communications, ICC 2012, Ottawa, ON, Canada, June 10-15, 2012*. IEEE, 2012, pp. 5146–5150.

[6] Bouziane Brik, Karim Boutiba, and Adlen Ksentini. "Deep Learning for B5G Open Radio Access Network: Evolution, Survey, Case Studies, and Challenges". In: *IEEE Open Journal of Communication Society* 3 (2022), pp. 228–250.

[7] OpenAI. *Deep Deterministic Policy Gradient*. 2022. URL: https://spinningup.openai.com/en/latest/algorithms/ddpg.html. (accessed: 08.08.2022).

[8] AW2S. *Radio products for wireless networks*. 2023. URL: https://www.aw2s.com/electronic-engineering/engineering-services/. (accessed: 07.03.2023).

[9] Quectel. *Quectel RM500Q-GL specifications*. 2023. URL: https://www.quectel.com/wp-content/uploads/2021/03/Quectel_RM500Q-GL_5G_Specification_V1.3.pdf. (accessed: 07.03.2023).

[10] Jon Dugan, Seth Elliott, Bruce A. Mah, Jeff Poskanzer, Kaustubh Prabhu. *Iperf: The ultimate speed test tool for TCP, UDP and SCTP*. 2022. URL: https://iperf.fr/. (accessed: 08.08.2022).

[11] Techtrained. *TDD : Why a Guard Period only in DL-to-UL Switching ?* 2023. URL: https://www.techtrained.com/lte-tdd-why-a-guard-period-only-in-dl-to-ul-switching/. (accessed: 07.03.2023).