A 5G multi-Slice cell capacity framework

Gabriela Pereyra, Claudina Rattaro and Pablo Belzarena
pereyra.gab@gmail.com;crattaro@fing.edu.uy;belza@fing.edu.uy
Facultad de Ingeniería, Universidad de la República, Montevideo, Uruguay

ABSTRACT

In this work we present Py5cheSim, a flexible and open-source simulator based on Python and specially oriented to simulate cell capacity in 3GPP 5G networks and beyond. To the best of our knowledge, Py5cheSim is the first simulator that supports Network Slicing at the Radio Access Network, one of the main innovations of 5G.

1 INTRODUCTION

The different types of services to be supported by 5G networks (enhanced Mobile Broad Band, Ultra Reliable and Ultra Low Latency and massive Mobile Type Communications) have vastly heterogeneous traffic characteristics, quality of service requirements and even energy consumption associated [3]. The unified management of these services in Layer 2, particularly designing the algorithms related to scheduling, access control, interference management and power control, is a daunting task. Although several mechanisms have been standardized in 5G to consider these services' requirements (e.g. Flexible Radio Frame Structure, mMIMO, Flexible HARQ, Network Slicing), the standard does not define how they should be used. Equipment providers will be faced thus with the challenge of designing the algorithms mentioned above. Operators will have to evaluate and choose among several possibilities. In this context, a tool which allows to generate different types of scenarios in 5G networks and to easily test different scheduling algorithms is essential.

As scheduling is always a delicate vendor topic, few free tools simulate cell capacity in 3GPP networks. Existing tools represent only a selected set of features presented in the 5G standard. Most importantly, none of the existing known simulators have the specific features and flexibility needed to implement and evaluate a complete cell capacity analysis. Even more, no one implements Network Slicing at the RAN (Radio Access Network) level which is one of the main key new features.

On one hand, we have network simulators (e.g. ns3 with 5G-LENA module [6]) which often implement layer by layer most of the procedures described in the 3GPP standard, so the simulation turns hard to configure and implies high processor loads. On the other hand, System-Level Simulators (e.g. Vienna [4] and Simu5G [5]) often has a high degree of simplification to cover a wide range of cells with affordable resource use. To tackle these problems, we have developed a new simulator in Python platform, named Py5cheSim. The general design goal of the developed simulator is to keep it as simple as possible, trying to maintain the biggest freedom degree possible for scheduler implementation. Py5cheSim allows analyzing inter and intra-slice scheduling. Also, as there is no need to implement layer by layer all the procedures defined in the standard, this new simulator is more straightforward, lighter, and quicker than many of the existing free tools. Furthermore, but not less important, as Python offers a vast choice of libraries for Artificial Intelligence (AI) development, Py5cheSim allows to implement easily and test AI-based algorithms.

The rest of the paper is structured as follows. In Section 2 we briefly describe Py5cheSim characteristics.Then, in Section 3 we present the validation process. Finally, Section 4 discusses our roadmap and plans and concludes the work. It is important to remark that a complete description of Py5cheSim architecture is presented in an article that is actually in at peer review process at CLEI XLVII conference[1]. Py5cheSim v1.0 is available at our web page[2].

2 SYSTEM CHARACTERISTICS

Py5cheSim implements RAN Slicing as a core feature using a two-level scheduler composed of an Intra Slice Scheduler and an Inter Slice Scheduler. The first one is oriented to solve resource allocation between different UEs of the same Slice, and the second to allocate resources between the different Slices. Each Slice has a set of requirements and a configuration. Configuration is set automatically depending on Slice requirements in terms of delay, band, the number of UEs to serve, traffic profile, UE capabilities, and availability. For each Slice, numerology/SCS (Sub-carrier Spacing)/TTI (Transmission Time Interval), duplexing mode, scheduler algorithm to use, signaling load, and allocated PRBs (Physical Resource Blocks) are set at the initializing of the simulator. Slice allocated PRBs can change according to interSlice scheduler decision with a TTI granularity.

Py5cheSim supports multiple numerologies, FDD (Frequency Division Duplex) and TDD (Time Division Duplex) frame (depending on the cell band set for the simulation),

[1] https://clei2021.cr

[2] https://iie.fing.edu.uy/investigacion/grupos/artes/proyectos/inteligencia-artificial-aplicada-a-redes-5g/

SIGCOMM N2Women'21, 2021
2021.

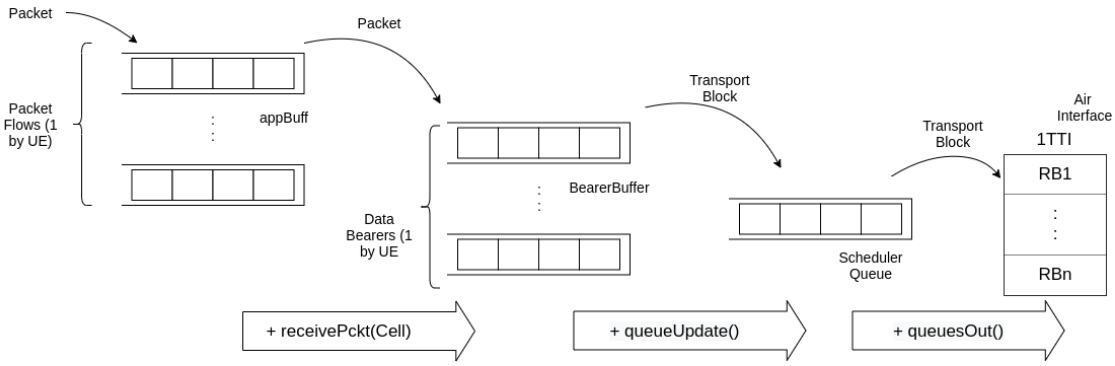

Figure 1: Intra Slice Scheduler Queues operative.

uplink and downlink bearers, and a basic implementation of Carrier Aggregation and Single-User/Multi-User MIMO functionalities. Transport Block Size (TBS) calculation, which depends on both the number of allocated PRBs and the MCS (Modulation and Coding Scheme) is based on 3GPP Technical Specifications [1, 2]. Py5cheSim also supports different traffic profiles configuration by groups of UEs that can emulate the different 5G services. The traffic profile is set in terms of average packet size and inter-arrival times.

The simulator basic operation can be seen in the Figure 1. The application generates a packet flow through the *queueAppPckt* method. Each packet is stored in an application queue, the first which appears in Figure 1. Then, when the UE reaches the connected state in the cell, the DRB (Data Radio Bearer) is established and its packets go to the bearer queue through the *receivePckt* method. Then, the scheduler assigns resources for all the active bearers and takes packets from there to make TB (Transport Blocks) with an appropriate MCS according the UE SINR at the moment and put them in the Scheduler queue through the *queueUpdate* method. Finally the scheduler takes the TB from the queue at each TTI and send them through the air interface. The TB are successfully received with a probability of (1-BLER (Block Error Rate)). Please note that BLER can be set overwriting the *setBLER* method.

3 VALIDATION PROCESS

A partial validation was made through the 5G-LENA module and the throughout calculator web tool from https://5g-tools.com/5g-nr-throughput-calculator/ (the last one represents a quick comparison with known analytical results). We said "partial" because there is no tool to compare ourselves in multi-slice scenarios. The general idea behind this validation was to test the developed simulator operation and compare performance results with the references in terms of the main KPI considered, using the same configuration scenarios. The validation and calibration process has been a comprehensive check verifying in a wide variety of scenarios.

4 CONCLUSIONS AND FUTURE WORK

This article presented Py5cheSim, a new discrete event Python simulator focused on cell capacity analysis. We are working on a second version of the simulator improving some features and including others. In particular we are working on a associated library to simplify the development and testing of new 5G scheduler algorithms (inter and intra Slice).

ACKNOWLEDGEMENTS

This work has been supported by the Agencia Nacional de Investigación e Innovación, Uruguay, Project FMV_1_2019_1_155700, "Artificial Intelligence applied to 5G networks".

REFERENCES

[1] 3GPP. 2021. *NR; Physical layer procedures for data.* Technical Specification (TS) 38.214. 3rd Generation Partnership Project (3GPP). https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3216 Version 16.5.0.

[2] 3GPP. 2021. *NR; User Equipment (UE) radio access capabilities.* Technical Specification (TS) 38.306. 3rd Generation Partnership Project (3GPP). https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3193 Version 16.4.0.

[3] ITU-R. 2015. *IMT Vision – Framework and overall objectives of the future development of IMT for 2020 and beyond.* Recommendation M.2083-0. International Telecommunication Union, Geneva.

[4] Martin Klaus Müller, Fjolla Ademaj, Thomas Dittrich, Agnes Fastenbauer, Blanca Ramos Elbal, Armand Nabavi, Lukas Nagel, Stefan Schwarz, and Markus Rupp. 2018. Flexible multi-node simulation of cellular mobile communications: the Vienna 5G System Level Simulator. *EURASIP Journal on Wireless Communications and Networking* 2018, 1 (Sept. 2018), 17. https://doi.org/10.1186/s13638-018-1238-7

[5] Giovanni Nardini, Giovanni Stea, Antonio Virdis, and Dario Sabella. 2020. Simu5G: A System-level Simulator for 5G Networks. https://doi.org/10.5220/0009826400680080

[6] Natale Patriciello, Sandra Lagen, Biljana Bojovic, and Lorenza Giupponi. 2019. An E2E simulator for 5G NR networks. *Simulation Modelling Practice and Theory* 96 (2019), 101933. https://doi.org/10.1016/j.simpat.2019.101933

A 5G multi-Slice cell capacity framework

Gabriela Pereyra, Claudina Rattaro, Pablo Belzarena
Universidad de la República, Uruguay

N2Women'21

Introduction

The different types of services to be supported by 5G networks have vastly heterogeneous traffic characteristics, quality of service requirements and even energy consumption associated (ITU-IMT2020). The unified management of these services in Layer 2, particularly designing the algorithms related to scheduling, access control, interference management and power control, is a daunting task. Although several mechanisms have been standardized in 5G to consider these services' requirements (e.g. Flexible Radio Frame Structure, mMIMO, Flexible HARQ, Network Slicing), the standard does not define how they should be used. Equipment providers will be faced thus with the challenge of designing the algorithms mentioned above. Operators will have to evaluate and choose among several possibilities. In this context, a tool which allows to generate different types of scenarios in 5G networks and to easily test different scheduling algorithms is essential. In this work we present Py5cheSim, a flexible and open-source simulator based on Python and specially oriented to simulate cell capacity in 3GPP 5G networks and beyond. To the best of our knowledge, Py5cheSim is the first simulator that supports Network Slicing at the Radio Access Network.

System Characteristics

Py5cheSim implements RAN Slicing as a core feature using a two-level scheduler composed of an Intra Slice Scheduler and an Inter Slice Scheduler. The first one is oriented to solve resource allocation between different UEs of the same Slice, and the second to allocate resources between the different Slices. Each Slice has a set of requirements and a configuration. Configuration is set automatically depending on Slice requirements in terms of delay, band, the number of UEs to serve, traffic profile, UE capabilities, and availability. For each Slice, numerology/SCS/TTI, duplexing mode, scheduler algorithm to use, signaling load, and allocated PRBs (Physical Resource Blocks) are set at the initializing of the simulator. Slice allocated PRBs can change according to interSlice scheduler decision with a configurable granularity.

Py5cheSim supports multiple numerologies, FDD and TDD frame (depending on the cell band set for the simulation), uplink and downlink bearers, and a basic implementation of Carrier Aggregation and Single-User/ Multi-User MIMO functionalities. Transport Block Size calculation, which depends on both the number of allocated PRBs and the MCS is based on 3GPP Technical Specifications. It also supports different traffic profiles configuration by groups of UEs that can emulate the different 5G services.

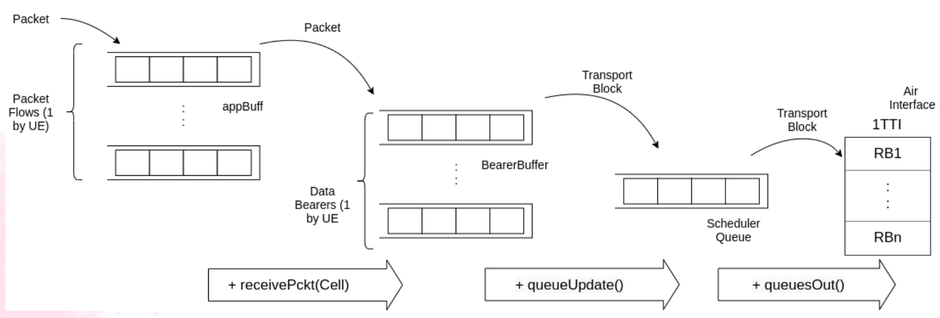

Fig.1. Intra Slice Scheduler Queues operative.

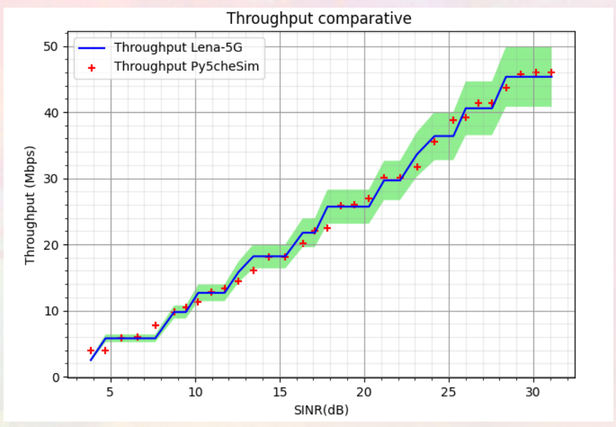

Fig.2. Throughput of Py5cheSim vs 5G-LENA.

Conclusions

Py5cheSim is a new discrete event Python simulator focused on cell capacity analysis. Its validation and calibration process has been a comprehensive check verifying in a wide variety of scenarios.

We are working on a second version of the simulator improving some features and including others. In particular we are working on a associated library to simplify the development and testing of new 5G scheduler algorithms (inter and intra Slice).

FACULTAD DE INGENIERIA

UNIVERSIDAD DE LA REPÚBLICA URUGUAY