An Extension of the ns-3 LTE Module to Simulate Fractional Frequency Reuse Algorithms

Piotr Gawłowicz\textsuperscript{1} Nicola Baldo\textsuperscript{2} Marco Miozzo\textsuperscript{2}

\textsuperscript{1}AGH University of Science and Technology
Kraków, Poland

\textsuperscript{2}Centre Tecnològic de Telecomunicacions de Catalunya
Barcelona, Spain

The Workshop on ns-3, 2015
Outline

1. Motivation
2. Frequency Reuse in cellular networks
3. FFR implementation
4. Implemented FFR algorithms
5. Configuration
6. Downlink Power Control
7. Uplink Power Control
8. Testing
9. Examples
10. Visualization
11. Summary
12. GSoC experience
Motivation

- FFR algorithms fit in the general category of Self Organized Network algorithms
- The LTE standard does not provide the design of FFR algorithms
- Design is left open for LTE equipment vendors to create their own solutions
- FFR solutions receive a significant attention in the industry and within academia
- Simulation tool is needed to implement and compare the performance of the new FFR algorithms
Frequency Reuse in cellular networks

- In LTE each eNB use the same carrier frequency and system bandwidth to serve all of its users; FRF = 1
- It leads to very high throughput in cell center, but very low at the cell edge (due to high interferences)
- FFR divide available bandwidth into sub-bands with different FRF and different TX power setting
- FFR in LTE technology is possible thanks to its dynamic MAC scheduling and power control functionalities
FFR implementation — Data Plane
FFR implementation — API

- **FFR-RRC SAP**
  - AddUeMeasReportConfigForFfr
  - SetPdschConfigDedicated
  - SendLoadInformation

- **eNodeB RRC**

- **FFR-RRC SAP**
  - SetCellId
  - SetBandwidth
  - ReportUeMeas
  - RecvLoadInformation

- **Fractional Frequency Reuse Algorithm**

- **FFR-MAC SAP**
  - GetAvailableDlRbg
  - IsDlRbgAvailableForUe
  - GetAvailableUlRbg
  - IsUlRbgAvailableForUe
  - ReportDlCqiInfo
  - ReportUlCqiInfo
  - GetTpc
  - GetMinContinuousUlBandwidth

- **MAC Scheduler**
FFR implementation — Scheduling

MAC schedulers supporting FFR: PF, PSS, CQA, TD-TBFQ and FD-TBFQ
Implemented FFR algorithms

- Full Frequency Reuse (no-op)
- Hard Frequency Reuse
- Strict Frequency Reuse
- Soft Frequency Reuse (two versions)
- Soft Fractional Frequency Reuse
- Enhanced Fractional Frequency Reuse
- Distributed Frequency Reuse Scheme
Manual configuration

- FFR algorithm type needs to be specified in LteHelper
- Default algorithm: *no-op*
- Each algorithm provides different set of attributes, that have to be configured by user
- Default configuration for each FFR algorithm is provided (the same for each cell)
Automatic configuration

- *Manual* configuration is quite complex
- *Automatic* solution was implemented to avoid problems with sub-bands configuration
- User needs to set only FrCellTypeId, which can take value of \{1,2,3\}
- *Note*: only sub-bands will be automatically configured; in most cases, this is enough to perform a meaningful simulation
Configuration example

```c
lteHelper->SetFfrAlgorithmType("ns3::LteFrSoftAlgorithm");

// Option 1: Manual Configuration:
lteHelper->SetFfrAlgorithmAttribute("DlEdgeSubBandOffset", UintegerValue(8));
lteHelper->SetFfrAlgorithmAttribute("DlEdgeSubBandwidth", UintegerValue(8));
lteHelper->SetFfrAlgorithmAttribute("UlEdgeSubBandOffset", UintegerValue(8));
lteHelper->SetFfrAlgorithmAttribute("UlEdgeSubBandwidth", UintegerValue(8));

// Option 2: Automatic Configuration:
lteHelper->SetFfrAlgorithmAttribute("FrCellTypeId", UintegerValue(2));

lteHelper->SetFfrAlgorithmAttribute("AllowCenterUeUseEdgeSubBand", BooleanValue(false));
lteHelper->SetFfrAlgorithmAttribute("RsrqThreshold", UintegerValue(20));
lteHelper->SetFfrAlgorithmAttribute("CenterPowerOffset", UintegerValue(LteRrcSap::PdschConfigDedicated::dB0));
lteHelper->SetFfrAlgorithmAttribute("EdgePowerOffset", UintegerValue(LteRrcSap::PdschConfigDedicated::dB3));
```
Downlink Power Control

- DPC is essential for FFR algorithms
- Requirements in 3GPP, TS 36.213 Physical Layer Procedures
- Only P_A values implemented
- DPC mechanism is partially embedded in FFR algorithms implementation

Source: www.sharetechnote.com
Uplink Power Control

- UPC allows to adjust the transmission power, thus reduce interferences and power consumption
- Requirements in 3GPP, TS 36.213 Physical Layer Procedures
- New class LteUePowerControl is responsible for computing and updating the power levels of UL channels (PUSCH and SRS)
- Two UPC mechanisms implemented:
  - Open Loop
  - Closed Loop (with two modes: Absolute and Accumulation)
- Several attributes available
- Trace sources for collection of TX power uplink channels
- FFR algorithm in eNB is responsible for determining the proper values of TPC commands for each UE
Testing

New test suites:

- **lte-frequency-reuse**
  - three test cases types suitable for different FFR algorithms
  - test vector comprises configurations for all FFR algorithms
  - each FFR algorithm is tested with all the schedulers which support FFR
  - tests pass if the UE is served in DL and UL using the RBs and power levels expected for the FR algorithm being tested

- **lte-downlink-power-control**
  - test case for SpectrumValue creation (PHY layer)
  - test case to check power level difference between PDCCH and PDSCH
  - test case for RRC Reconfiguration

- **lte-uplink-power-control**
  - test case for UPC with Open Loop
  - test case for UPC with Closed Loop in Absolute mode
  - test case for UPC with Closed Loop in Accumulation mode
Examples

- Two examples are provided to show basic FFR algorithms functionalities:
  - *lena-frequency-reuse*
  - *lena-distributed-reuse*
- `RadioEnvironmentMapHelper` was extended to be able to generate radio environment map on a per RB basis
- *lena-dual-stripe* example now supports FFR algorithms; `LteHexGridEnbTopologyHelper` extended
Visualization

Spectrum Analyzer trace; Soft FFR

REM for RB 1; Soft FR
Summary

Outcome:

- FFR API and 7 algorithms are available
- Power Control in DL and UL are implemented
- Test and examples are provided
- Documentation can be found on project official webpage
- The code was merged to official ns-3 repository and is available from version 3.21
- The project was funded by the Google Summer of Code 2014 program

Future work:

- Compare performance of implemented FFR algorithms
- More advanced mechanism for UPC
GSoC experience

- Thanks to Nicola and Marco for mentorship
- Opportunity to see how open source community works
- Good communication (telco, mail) = fruitful cooperation
- Perfect organization: proposal, weekly reports and code reviews
Thank you