Simulation of scheduling algorithms for femtocells in an LTE environment

Examensarbete utfört i Kommunikationssystem vid Tekniska högskolan i Linköping
av

Kristoffer Roberg

LiTH-ISY-EX--10/4396--SE

Linköping 2010
Simulation of scheduling algorithms for femtocells in an LTE environment

Examensarbete utfört i Kommunikationssystem vid Tekniska högskolan i Linköping
av

Kristoffer Roberg

LiTH-ISY-EX--10/4396--SE

Handledare: Dr. Jonas Eriksson
ISY, Linköpings universitet
Dr. Kwang-Cheng Chen
GICE, National Taiwan University

Examinator: Dr. Jonas Eriksson
ISY, Linköpings universitet

Linköping, 22 September, 2010
Simulering av schemaläggningsalgoritmer för femtoceller i en LTE miljö

Simulation of scheduling algorithms for femtocells in an LTE environment

Kristoffer Roberg

The new mobile standard *Long Term Evolution* delivers high data rates, small delay and a more efficiently utilized RF spectrum. A solution to maintain this performance in user dense areas or areas with bad reception is the deployment of so-called femtocells. Femtocells are small base stations that are deployed indoors and share the RF spectrum with the whole mobile network. The idea is that femtocells will increase mobile operators network coverage and capacity while it at the same time increase users data throughput. There are several challenges with femtocells, both technical and economical ones. The most debated issues is how femtocells should schedule users while operating in an environment where other femtocells and base stations are interfering. In this work we developed a simulation tool to simulate the scheduling interaction between femtocells and base stations in order to show the performance of radio resource schedulers. This rapport also aims to evaluate an approach to a femtocell scheduler to solve this issue in a satisfying way. The report gives a description of the structure of the implemented simulation tool together with some reflections on how future designs of similar or more complex simulation environments could be done.

Keywords LTE, femtocells, FAP, simulering, OFDMA, RRM, scheduling, resource allocation
Abstract

The new mobile standard *Long Term Evolution* delivers high data rates, small delay and a more efficiently utilized RF spectrum. A solution to maintain this performance in user dense areas or areas with bad reception is the deployment of so-called femtocells. Femtocells are small base stations that are deployed indoors and share the RF spectrum with the whole mobile network. The idea is that femtocells will increase mobile operators network coverage and capacity while it at the same time increase users data throughput. There are several challenges with femtocells, both technical and economical ones. The most debated issues is how femtocells should schedule users while operating in an environment where other femtocells and base stations are interfering. In this work we developed a simulation tool to simulate the scheduling interaction between femtocells and base stations in order to show the performance of radio resource schedulers. This rapport also aims to evaluate an approach to a femtocell scheduler to solve this issue in a satisfying way. The report gives a description of the structure of the implemented simulation tool together with some reflections on how future designs of similar or more complex simulation environments could be done.

Sammanfattning

I want to thank my Taiwanese advisor Dr. Kwang-Cheng Chen for being so kind to allow me to write this thesis in his lab during my stay at National Taiwan University. I also want to thank my fellow students, Mr. Shao-Yu Lian and Mr. Chu Feng-Sen for our many discussions. Further, I want to thank my Swedish advisor, Dr. Jonas Eriksson, for giving me straight and fast replies on my many mails and for valuable discussions at the finalizing of the thesis.

I thank my senior high school math and physics teacher, Mr. Gunnar Gardström, for inspiring me to educate myself in this area. I am uttermost grateful for my parents constant support and appreciation in whatever I do. Most of all I want to thank Chu Ying-Chieh and her family for the kindness and support during my stay in Taiwan.
# Contents

1 Abbreviations and Terms ........................................... 3

2 Introduction ....................................................... 7
   2.1 Goals of the thesis ........................................... 8

3 Introduction to a OFDMA based mobile system ................... 9
   3.1 Downlink transmission ....................................... 9
   3.2 User Downlink Scheduling ................................... 12

4 Long Term Evolution ............................................... 15
   4.1 Background .................................................. 15
   4.2 Technology overview ........................................ 16
   4.3 System level ................................................ 17
   4.4 Evolved Packet Core ....................................... 18
   4.5 Evolved Universal Terrestrial Radio Access .............. 18
   4.6 Long Term Evolution Advanced ............................ 26

5 Femtocells .......................................................... 27
   5.1 Concept overview ........................................... 27
   5.2 Business challenges ......................................... 27
   5.3 Technology challenges ...................................... 28
   5.4 Resource scheduling for femtocells ...................... 28

6 QoS scheduling for LTE femtocells ................................ 31
   6.1 Previously proposed QoS aware schedulers .............. 31
   6.2 Reference schedulers ....................................... 32
   6.3 Proposed QoS-aware scheduler ............................. 34
   6.4 Simulation results .......................................... 37

7 System simulation tool ............................................. 45
   7.1 Goals ......................................................... 45
   7.2 Framework .................................................. 45
   7.3 Implementation environment ................................ 48
   7.4 Overview .................................................... 48
   7.5 Initiating the environment ................................ 48
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.6 Channel model</td>
<td>52</td>
</tr>
<tr>
<td>7.7 Generate CQI reports</td>
<td>55</td>
</tr>
<tr>
<td>7.8 Traffic models</td>
<td>57</td>
</tr>
<tr>
<td>7.9 Radio Resource Management</td>
<td>59</td>
</tr>
<tr>
<td>7.10 Data Transmission</td>
<td>60</td>
</tr>
<tr>
<td>7.11 HARQ functionality</td>
<td>64</td>
</tr>
<tr>
<td>7.12 Evaluation</td>
<td>64</td>
</tr>
<tr>
<td>7.13 Simulation results</td>
<td>67</td>
</tr>
<tr>
<td>8 Conclusions and future work</td>
<td>71</td>
</tr>
<tr>
<td>8.1 FAP scheduling</td>
<td>71</td>
</tr>
<tr>
<td>8.2 The simulator</td>
<td>71</td>
</tr>
<tr>
<td>8.3 Future work</td>
<td>72</td>
</tr>
<tr>
<td>Bibliography</td>
<td>75</td>
</tr>
<tr>
<td>A Implementation class list</td>
<td>79</td>
</tr>
<tr>
<td>A.1 SimEnvironment</td>
<td>80</td>
</tr>
<tr>
<td>A.2 Node</td>
<td>83</td>
</tr>
<tr>
<td>A.3 Enode</td>
<td>85</td>
</tr>
<tr>
<td>A.4 UE</td>
<td>87</td>
</tr>
<tr>
<td>A.5 FAP</td>
<td>91</td>
</tr>
<tr>
<td>A.6 BS</td>
<td>92</td>
</tr>
<tr>
<td>A.7 ChannelModel</td>
<td>95</td>
</tr>
<tr>
<td>A.8 CQI</td>
<td>101</td>
</tr>
<tr>
<td>A.9 UEscheduling</td>
<td>102</td>
</tr>
<tr>
<td>A.10 EnodeScheduling</td>
<td>103</td>
</tr>
<tr>
<td>A.11 Scheduler</td>
<td>104</td>
</tr>
<tr>
<td>A.12 Transportation</td>
<td>106</td>
</tr>
<tr>
<td>A.13 EPSstream</td>
<td>113</td>
</tr>
<tr>
<td>A.14 Packet</td>
<td>114</td>
</tr>
<tr>
<td>A.15 Logger</td>
<td>115</td>
</tr>
</tbody>
</table>
7.12 Simulation speed ............................................. 69
A.1 Properties of the SimEnvironment class ...................... 80
A.1 Properties of the SimEnvironment class ...................... 81
A.2 Structs defined in the Node class file ........................ 83
A.3 Properties of the Node class ................................. 83
A.4 Structs defined in the Node class file ........................ 85
A.5 Properties of the Enode class ................................. 86
A.6 Properties of the UE class .................................... 88
A.7 Properties of the FAP class ................................... 91
A.8 Structs defined in the BS class file .......................... 92
A.9 Properties of the BS class ..................................... 93
A.10 Structs defined in the ChannelModel class file .......... 95
A.11 Properties of the ChannelModel class ...................... 96
A.12 Properties of the CQI class ................................ 101
A.13 Structs defined in the UEscheduling class file ........... 102
A.14 Properties of the UEscheduling class ...................... 103
A.15 Structs defined in the EnodeScheduling class file ........ 103
A.16 Properties of the EnodeScheduling class .................. 104
A.17 Structs defined in the Transportation class file .......... 106
A.18 Properties of the Transportation class .................... 107
A.19 Properties of the EPSstream class ........................ 113
A.20 Properties of the Packet class .............................. 114
A.21 Properties of the Logger class .............................. 115
Chapter 1
Abbreviations and Terms

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3GPP</td>
<td>Third Generation Partnership Program system.</td>
</tr>
<tr>
<td>AMBR</td>
<td>Aggregate Maximum Bit Rate</td>
</tr>
<tr>
<td>ARP</td>
<td>Allocation and Retention Priority</td>
</tr>
<tr>
<td>AWGN</td>
<td>Additive White Gaussian Noise</td>
</tr>
<tr>
<td>BS</td>
<td>Base Station, also referred to as eNodeB</td>
</tr>
<tr>
<td>BW</td>
<td>BandWidth</td>
</tr>
<tr>
<td>CB</td>
<td>Code Blocks</td>
</tr>
<tr>
<td>CP</td>
<td>Cyclic Prefix</td>
</tr>
<tr>
<td>CQI</td>
<td>Channel Quality Indicator</td>
</tr>
<tr>
<td>CRC</td>
<td>Cyclic Redundancy Check</td>
</tr>
<tr>
<td>CSI</td>
<td>Channel State Information</td>
</tr>
<tr>
<td>EDGE</td>
<td>Enhanced Data rates for GSM Evolution</td>
</tr>
<tr>
<td>Enode</td>
<td>Is referring to a node in the LTE network that is sending at downlink, for example BS and FAP.</td>
</tr>
<tr>
<td>EPC</td>
<td>Evolved Packet Core</td>
</tr>
<tr>
<td>EPS</td>
<td>Evolved Packet System</td>
</tr>
<tr>
<td>FDD</td>
<td>Frequency Division Duplex</td>
</tr>
<tr>
<td>FAP</td>
<td>Femtocell Access Point and is also synonym for femtocell or HeNodeB.</td>
</tr>
<tr>
<td>GBR</td>
<td>Guaranteed Bit Rate</td>
</tr>
<tr>
<td>GPRS</td>
<td>General Packet Radio Service</td>
</tr>
<tr>
<td>GPS</td>
<td>Global Positioning System</td>
</tr>
<tr>
<td>HARQ</td>
<td>Hybrid Automatic Repeat Request</td>
</tr>
<tr>
<td>HSPA</td>
<td>High Speed Packet Access</td>
</tr>
<tr>
<td>IDFT</td>
<td>Inverse Discrete Fourier Transform</td>
</tr>
<tr>
<td>IFFT</td>
<td>Inverse Fast Fourier Transform</td>
</tr>
<tr>
<td>ISI</td>
<td>Inter Symbol Interference</td>
</tr>
<tr>
<td>ISP</td>
<td>Internet Service Provider</td>
</tr>
<tr>
<td>LTE</td>
<td>Long Term Evolution and is sometimes referred to as 3GPP’s Rel.8, 3G+ or as 4G (even though it is NOT defined as 4G)</td>
</tr>
<tr>
<td>LTE-A</td>
<td>Long Term Evolution Advanced and is the evolution of LTE and is defined as a 4G technology.</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>MIMO</td>
<td>Multiple Input Multiple Output</td>
</tr>
<tr>
<td>MME</td>
<td>Mobility Management Entity</td>
</tr>
<tr>
<td>NAS</td>
<td>Non-Access Stratum</td>
</tr>
<tr>
<td>OFDM</td>
<td>Orthogonal Frequency Division Multiplexing</td>
</tr>
<tr>
<td>PDCP</td>
<td>Packet Data Convergence Protocol</td>
</tr>
<tr>
<td>PDN</td>
<td>Packet Data Network</td>
</tr>
<tr>
<td>PDU</td>
<td>Packet Data Units</td>
</tr>
<tr>
<td>PER</td>
<td>Packet Error Rate</td>
</tr>
<tr>
<td>PF</td>
<td>Proportional Fair</td>
</tr>
<tr>
<td>P-GW</td>
<td>Packet data network GateWay</td>
</tr>
<tr>
<td>PHY-layer</td>
<td>PHYsical-layer</td>
</tr>
<tr>
<td>QAM</td>
<td>Quadratic Amplitude Modulation</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>QPSK</td>
<td>Quadratics Phase Shifting Key</td>
</tr>
<tr>
<td>RAN</td>
<td>Radio Access Network</td>
</tr>
<tr>
<td>RE</td>
<td>Resource Element</td>
</tr>
<tr>
<td>RF</td>
<td>Radio Frequency</td>
</tr>
<tr>
<td>ROHC</td>
<td>RObust Header Compression</td>
</tr>
<tr>
<td>RRM</td>
<td>Radio Resource Management</td>
</tr>
<tr>
<td>RS</td>
<td>Reference Signal</td>
</tr>
<tr>
<td>SAE</td>
<td>System Architecture Evolution</td>
</tr>
<tr>
<td>S-GW</td>
<td>Serving Gateway</td>
</tr>
<tr>
<td>SINR</td>
<td>Signal to Noise plus Interference Ratio</td>
</tr>
<tr>
<td>TB</td>
<td>Transport Block</td>
</tr>
<tr>
<td>TDD</td>
<td>Time Division Duplex</td>
</tr>
<tr>
<td>TTI</td>
<td>Transmission Time Interval</td>
</tr>
<tr>
<td>UE</td>
<td>User Equipment and is a naming convention for all connected devices such as cell phones, laptops and so on.</td>
</tr>
<tr>
<td>VoIP</td>
<td>Voice over IP</td>
</tr>
<tr>
<td>WCDMA</td>
<td>Wideband Code Division Multiple Access</td>
</tr>
</tbody>
</table>
Chapter 2

Introduction

Long Term Evolution (LTE) is a third generation (3G) mobile system defined by the organization 3G Partnership Project (3GPP). Often LTE is referred to as 4G in the daily media, probably to enable people without a technical background to differ between the first 3G systems (WCDMA and HSPA) and the later (LTE).

2010 is the year when the new mobile standard LTE is prepared to conquer the mobile market. With the two first live LTE networks deployed in Sweden and Norway already in 2009, there is another 20 worldwide operators scheduling to deploy LTE LTE networks during 2010[17]. By 2012 there is confirmed to be at least 45 networks worldwide. The spendings on LTE are expected to proceed the ones on WiMAX by 2011 and in total there is 110 mobile operators with LTE commitments [17].

The standard of the fourth generation (4G) mobile systems from 3GPP, Long Term Evolution Advanced (LTE-A), may be finished by the end of the year and will by the time it is deployed deliver the peak downlink rates of amazing 1Gbps[10]. Even though the technologies seem to be able to deliver unbelievable opportunities there are some issues that need to be resolved for these technologies to succeed.

LTE is a IP-packet based system with both higher data rates and lower packet delay than in previous technologies. Some years ago it was impossible to ensure that a IP-packet sent through the internet to arrive within a certain time frame, but today more or less it is possible. Today there is also more and more applications that demand that the delay is not too large and one example of that is Voice over IP (VoIP) with applications such as Skype. In the previous mobile standards such as GSM or CDMA there is no mechanisms to ensure that data is not delayed more than a certain limit. In LTE on the contrary, there is full support to ensure this through so-called Quality of Service commitments which is deployed between the outer gateway of the mobile network and the user equipment(UE)[5]. The unit that is responsible to ensure these commitments is the radio resource scheduler in the base stations. The functionality of the resource scheduler is not mentioned in LTE standard and is therefore up to the system providers and operators to design. When designing the resource scheduler it has to be decided on how to combine different priorities and how to use the resources into satisfying solution.
The scheduling problem is further explained in section 3.2.

If the reader is new to mobile systems it is suggested to first read the introduction to mobile systems in chapter 3.

2.1 Goals of the thesis

The main goal of this thesis is to provide an approach to construct a simplified but yet valid LTE system simulator for evaluation of scheduling algorithms in an environment containing both femtocells and base stations. The target technology of the thesis is LTE Rel.8 and Rel.9 and the reason to this is that these specifications lack features regarding femtocell, which on the contrary the final release Rel.10 will include. Therefore, deploying femtocells in a LTE environment is likely to be far more challenging than in LTE-A (Rel.10) due to the lack of supporting features in the standard.

Many believes femtocells are a key technology for the success of LTE and especially LTE-A. That makes it uttermost important that femtocells are good enough already for LTE; due to of the mediocre success in previous technologies indicated by the positioning of the mobile system giant Ericsson to not develop new femtocells for 3G technologies[32]. A failure in commercially deploying LTE femtocells would risk to hurt the adoption willingness among both users and network providers when it is time to deploy them for LTE-A, where it is far more important to achieve the target data rates. Therefore this thesis also aims to provide a view on scheduling for femtocells which provides a promising solution that focus on trying to guarantee the user experience and at the same time in an efficient manner utilizes the available network resources.
Chapter 3

Introduction to a OFDMA based mobile system

In this chapter a more general description of mobile systems and especially those using OFDMA. Most mobile systems includes base stations which communicates with the users equipment. These base stations are somehow connected to one or several gateway(s) which handles certain data transformation, security routines, packet mapping and hand-offs between base stations. This gateway also handles the communication with the fixed phone network and internet. A general system is shown in Figure 3.1. The description of base stations’ placement is usually generalized by that all base stations are deployed in a hexagonal pattern, as in Figure 3.2. The placement of base stations in reality is also influenced by the environment and user density.

3.1 Downlink transmission

The transmission from a base station (BS) to an user equipment (UE), such as a mobile phone, is referred to as downlink while the transmission from a UE to a BS is called uplink. When data is about to be transmitted it is first going through several steps to become ready to be sent, a simple illustration of these steps are shown in Figure 3.3. First error detection and correcting codes are added which increases the size of the data. It is common that some punctuation is also performed, which means that some bits are removed in such a pattern that they can still be retrieved from the help of the remaining bits. The bits are then modulated into complex symbols using a mapping pattern. For example if Quadratic Phase Shifting Key (QPSK) is used two bits are mapped to one of four possible complex symbols. When opportunistic transmission is used the used modulation is depending on the quality of the channel. The complex symbols are then mapped into different sub-carriers, in frequency, and slots in time. A sub-carrier can be seen as one channel that data can be transmitted on and in an OFDMA system several adjacent sub-carriers (in frequency) are transmitted on
simultaneously. If multiple antennas is used, so called multiple input multiple output (MIMO), this process is even more complicated. The symbols on the different sub-carriers are transformed from frequency domain into time domain by an inverse discrete fourier transform (IDFT) which often is implemented as a inverse fast fourier transform (IFFT) as in equation 3.1, where $X(k)$ are the modulated complex symbols in frequency domain, $\Delta f$ is the sub-carrier spacing and $N$ is the number of sub-carriers [19].

$$s(t) = \sum_{k=0}^{N-1} X(k) \times e^{j2\pi k\Delta f t}$$

The parallel streams of complex symbols are then serialized and a cyclic prefix (CP) is added. Before the data is transmitted there are numerous RF-front end operations which are not that important in the context of this study. Once the data has been transmitted the signal is distorted by the propagation channel. In simulations the channel model tries to mimic the behavior of the reality with buildings blocking and reflecting the signal which fades the channel. This fading is often modeled by three parts, the fast fading which is from superpositioning of delayed versions of the signal, the slow fading which is from the physical environment such as buildings and the pathloss which is the power loss the signal suffer from the path it propagates through. When receiving the signal the time slots where the signal is measured are called taps. The number and size of taps depends on the

**Figure 3.1.** A simple illustration of a mobile phone network. Note that the "Gateway server" entity often consists of several gateways.
Figure 3.2. The standard deployment pattern for base stations. The numbers indicate the transmission sectors.
delay of the signal coming from the multi-path channel and decides the frequency
dependent fading. It is here the cyclic prefix helps to prevent delayed versions of
the transmitted signal to interfere with the next transmission. The received signals
$y(n)$, where $L$ is number of taps, $z(n) \sim N(0, \sigma^2)$ is additive white Gaussian noise
(AWGN), $s'$ the signals with CP added and $N$ is the number of sub-carriers and
$h_i$ is the channel coefficients for each tap[19].

$$y(n) = \sum_{i=0}^{L} h_i s'(n - i) + z(n) \quad n = 0, 1, \ldots, (N - 1)$$  (3.2)

After some mathematic operations and applying the DFT-transform on the re-
ceived signal the following can be shown, where $Y$ is column vector with the
symbols received on each RB, $H$ is a diagonal matrix with channel coefficients in
frequency domain, $z''$ is $z$ with some operations added (but not important) and
$X$ the transmitted symbols (without CP) on each sub-carrier.[19]

$$Y = HX + z''$$  (3.3)

It is also possible that $z$ consists of the sum of non-wanted signals, background
noise plus interfering transmission from other sources. The channel coefficients
and the variance of the noise can be measured by help from pilot signals and/or
reference signals. Pilot signals are set transmissions that come first in a trans-
mision while the reference signals are continuously sent on specific sub-carriers.
When the channel and the noise has been estimated it is possible to estimate the
transmitted symbols $X$.

$$H_i^{-1}Y_i = \hat{Y}_i = H_i^{-1}H_iX_i + H_i^{-1}z_i'' = X_i + H_i^{-1}z_i''$$  (3.4)

This $\hat{Y}$ is then demodulated into soft bits. If we assume that all modulation
constellations have the same average energy, a larger constellation such as 64QAM
will have smaller distance between the possible complex values. This increases
the error probability because the adding term $H^{-1}z''$ is more likely to shift the
transmitted symbol into another constellation point and thereby the wrong bits
will be retrieved. The soft bits represent each bit by an integer which describes
how close to the used constellation point the demodulated symbol was. The soft
bits created by the demodulator increases the performance of the decoding of the
punctuated convolutional code at a certain SINR level.

### 3.2 User Downlink Scheduling

Since the BS cannot transmit to all its UEs at the same time it has to use some
kind of rule how to divide the transmission opportunities among the UEs. This
task is taken care of by the radio resource management (RRM) unit, also called
radio resource scheduler, resource allocator or just scheduler, in the BS. Since
every mobile operator wants each BS to transmit as much data to as many UEs as
possible it is important that the scheduler is efficient. The problem the scheduler
Figure 3.3. Simple data flow before and after downlink transmissions.
face is to decide which IP packets from which users should be transmitted with what settings (such as modulation and transmitting power); all these should be decided out of the current information which usually is the, from UEs, fed back channel status, each UEs history and also how full is the buffer of packets waiting to be transmitted. A very basic approach is Round-Robin scheduling which take turns to allow the UEs without taking any of the available information into consideration. The scheduling can be coordinated between BS in order to decrease the interference between them. One way is to let the BS make use of sectors and only use a part of the available frequency spectrum in each sector. More efficiently, each BS can use different sub-carriers in each sector for transmitting to UEs close to the cell edge (which requires high transmitting power). Figure 3.2 illustrates the dividing into sectors and Figure 3.4 shows how the transmission spectrum can be divided[19], where $N$ is the number of sub-carriers in used spectrum.

![Figure 3.4](image-url). Illustration of the scheduled power for different parts of the spectrum on different sectors.
Chapter 4

Long Term Evolution

4.1 Background

The first steps towards Long Term Evolution (LTE) and LTE-A was taken back in 2-3 November 2004 at the RAN evolution workshop. The workshop was held by the organization 3GPP for both its members and non-members. During this workshop a number of contributions to take UMTS Terrestrial Radio Access Network (UTRAN) to the next generation, Evolved UTRAN was presented. Soon thereafter, in Dec 2004, a feasibility study for UTRA & UTRAN LTE was launched with goals to define its development strategy[2]. One year later, in Dec 2005, most of technologies for the physical layer was agreed upon[14]. 3GPPP itself was created in 1998 as a agreement between actors in the mobile communication industry with the purpose to prepare, approve and maintain globally applicable specifications for Global System for Mobile communication (GSM), General Packet Radio Service (GPRS) and Enhanced Data rates for GSM Evolution (EDGE)[7, 1]. Later the 3GPP agreement was expanded to also incorporate co-operation to evolve 3G and beyond 3G mobile systems based on the 3GPP core networks[2].

The purpose of the LTE development was to ensure that the 3G systems are competitive also in the future with closer connection to internet and applications requiring high data rates; this through a less complex IP-packet based system with high throughput, low latency and flexible bandwidth usage[8, 14]. In September 2006 the study phase of E-UTRAN was concluded[1], in December 2008 the release 8 of the LTE standard was frozen, including the radio network(E-UTRAN) and the core network System Architecture Evolution (SAE). The LTE standard was by this time considered stable and only minor enhancements was added to the release 9 which was planned to be frozen in December 2009[8]. During Mobile World Congress 2008 the first end-to-end call was demonstrated [15], and in December 2009 the first LTE networks were deployed in the Scandinavian cities of Oslo and Stockholm.
4.2 Technology overview

The main requirements for LTE on higher throughput (especially at cell edges), lower latency and less complex system architecture translates into requirements of higher spectrum efficiency and a flatter system architecture. The main technologies that was chosen to achieve these goals includes orthogonal frequency division multiplexing (OFDM) downlink, multiple-input-multiple-output (MIMO), channel sensitive scheduling, link adaptation, power control, packet oriented network architecture, QoS guarantees and more network mechanisms situated in the base station[19].

As mentioned earlier one of the key technologies in LTE is OFDM which enables parallel transmission over variable bandwidth, on so called sub-carriers, up to 20MHz. For more on OFDM see chapter 3 or [14, 19, 31]. Wideband OFDM do not only increase the throughput but also simplifies interference mitigation, lowers complexity and increase scalability[19]. The base station (BS) is capable of receiving channel state information (CSI) from the user equipments (UEs) on different parts of the bandwidth; this enables the channel sensitive scheduling, link adaptation and power control which in an unstandardized way allocates resources to users on certain sub-carriers which are experiencing good channels, this is so called channel sensitive scheduling or opportunistic transmission. The reason to why this scheduling is not mentioned in the LTE standard is that it is considered to be the equipment designers’ task to design this because there are several kinds of prioritizing when doing so. The link adaptation and power control used are dependent on the received CSI and on the used scheduling algorithm[19]. When a base station is communicating with user equipment (UE), i.e. mobile phone, there is a choice of how many antennas to use for both transmitter and receiver. This is the previously mentioned MIMO technology which increases the degrees of freedom in the transmissions and this can be used to achieve higher throughput or less transmission errors[31].

As a whole the LTE technology consists of two parts, the LTE radio access network (RAN) and the SAE: evolved packet core (EPC). In the LTE-RAN the functionality of UEs and BSs are described while in the EPC the functionality of the mobile management entity/gateway (MME/GW) is described[19]. The UE includes several layers to perform a number of operations in order to translate received IP-packets into OFDM symbols including encryption and error-detecting and correcting codes, for the antennas to transmit. The layers of the UE are the following: Packet Data Convergence Protocol (PDCP), Radio Link Control (RLC), Medium Access Control (MAC), PHYsical layer (PHY). The PDCP layer provides services to UE applications and the PHY receives and sends data to the radio frequency (RF) front-end also just referred to as the antenna(s)[19]. The BS has both the same functions as the UE and further functionality; inter cell radio resource management, resource block (RB) control, connection mobility control, radio admission control, E-UTRAN nodeB (eNB) measurement configuration and provision, dynamic resource allocation and radio resource controller (RRC)[19]. The only entity in the EPC is the MME/GW which includes a number of functions which in previous mobile standards was divided into several entities. This
is one of the reasons to why the system architecture in LTE is flatter and therefore supports shorter latency. The MME/GW includes three parts; the mobility management entity (MME), serving gateway (S-GW) and the packet data network gateway (P-GW). The MME has the following functions: NAS security, idle state mobility handling and evolved packet system (EPS) bearer control, while the S-GW includes the mobility anchoring, handles user movement between cells. The P-GW handles the communication towards outer networks such as internet and therefore has the two functions of UE IP address allocation and packet filtering. In the standard it is also defined what interfaces the network use to communicate, communication between eNBs use the X2 interface while communication between eNB and MME/GW uses the S1 interface[14], these are shown in figure 4.1.

4.3 System level

As mentioned before, the LTE system consists of three main entities the MME/GW, BS and UE. A simplified illustration of a LTE system is shown in figure 4.1. The MME/GWs are connected to the Internet through the mobile operator’s network.

Figure 4.1. An illustration of the system structure for LTE.
4.4 Evolved Packet Core

The Evolved Packet Core (EPC) contains one type of entity, the MME/GW. The MME/GW consists, as already mentioned, of three parts: the P-GW, the S-GW and the MME. The P-GW is the part that is connected to the Internet. When the P-GW receives IP packets from the Internet it classifies them and associates them to the correct evolved packet system bearer (EPS-bearer) and UE. The EPS-bearer concept will be explained further in the next section[19]. The P-GW also takes care of negotiation with outer applications to establish new EPS-bearers. The S-GW is connected to the P-GW and handles the communication with the BS through the S1 interface, which is a protocol for the communication. The S-GW also incorporates the mobility anchoring which is a mechanism to help to transfer a UE to a new BS when it moves into the coverage area of a new BS. The MME controls the non-access stratum (NAS) protocol which is a protocol used between the MME and UEs for security, authentication and EPS bearer control.

4.4.1 EPS bearers and QoS

When applications on the Internet want to send data to a UE it first has to negotiate with the P-GW to set up an IP-CAN session with the UE represented by an IPv4 and/or IPv6 address[5]. The P-GW then maps this session to an internal evolved packet system bearer (EPS-bearer) which is a session between the UE and the P-GW. Each of these EPS-bearers hold a set of requirements which the P-GW has agreed to follow, these set of requirements are called quality of service (QoS) requirements which include performance characteristics of the stream. When packets arrive at the P-GW they are classified and mapped to the correct EPS bearer[19]. There are three QoS parameters for each EPS-bearer defined in the LTE standard; QoS Class Identifier (QCI), Allocation and Retention Priority (ARP) and the Guaranteed Bit Rate (GBR)[16]. The QCI decides the delay budget, packet error loss rate and priority of the EPS stream[5]. The ARP translations are showed in table 4.1; only the data stream types used in the simulation are displayed in table 4.1 and it is the delay budget, packet error loss rate and priority that are used by the simulation tool. The ARP is used when resources are limited to decide which users are going to be accepted, refused or dropped; while the GBR is only provided for those applications that requires a certain bit rate to function, such as voice calls. Further, there is also a requirement, aggregate maximum bit rate (AMBR), on the group of EPS-bearers held by a single user in order to be able to limit the throughput for each user[16].

4.5 Evolved Universal Terrestrial Radio Access

The evolved universal terrestrial radio access network (E-UTRAN) includes BS and UEs and the purpose of it are the transmissions between these entities. The EUTRAN architecture consists of several layers. These layers are shown in figure 4.2 which illustrates downlink in E-UTRAN and uplink is performed in a similar way.
Figure 4.2. Illustration of the EUTRAN downlink transmissions[14].
<table>
<thead>
<tr>
<th>Requirement</th>
<th>QCI</th>
<th>Priority</th>
<th>Delay Budget</th>
<th>Packet Error Rate</th>
<th>Stream Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VoIP</td>
<td>1</td>
<td>2</td>
<td>80ms</td>
<td>$10^{-2}$</td>
<td>GBR</td>
</tr>
<tr>
<td>Live video</td>
<td>2</td>
<td>4</td>
<td>130ms</td>
<td>$10^{-3}$</td>
<td>GBR</td>
</tr>
<tr>
<td>Video streaming</td>
<td>4</td>
<td>5</td>
<td>280ms</td>
<td>$10^{-6}$</td>
<td>GBR</td>
</tr>
<tr>
<td>HTTP</td>
<td>6</td>
<td>6</td>
<td>280ms</td>
<td>$10^{-6}$</td>
<td>Non-GBR</td>
</tr>
<tr>
<td>FTP</td>
<td>6</td>
<td>6</td>
<td>280ms</td>
<td>$10^{-6}$</td>
<td>Non-GBR</td>
</tr>
</tbody>
</table>

Table 4.1. QoS requirements on different data types. Observe that a delay of 20 ms between the PCEF and the base station has already been deducted[5].

### 4.5.1 Packet Data Convergence Protocol

The Packet Data Convergence Protocol (PDCP) compresses the headers of the arriving IP packets using the algorithm robust header compression (ROHC). Unnecessary bits are removed because one want to transmit a minimal amount of bits. The PDCP layer also cipher the data to increase security. The PDCP also do the opposite on the receiving side, decipher received data and expand IP headers[14].

### 4.5.2 Radio Link Control

IP packets can be quite large (even after header compression), therefore it is the radio link control layer’s (RLC) task to segment the large IP-packets into smaller pieces, so called RLC packet data units (PDU). The RLC give each of these PDU a identity so the RLC at the receiver can arrange them into the correct order and then deliver to the upper layers. Further, the RLC has a retransmission mechanism in order to perform error free delivery to upper layers and if some of the PDU are missing it will be sent a request to resend those PDU. The sizes of the PDU depends on the size of the transmission to a single UE[14].

### 4.5.3 Medium Access Control

The medium access control layer has three functions, hybrid automatic repeat request (HARQ), downlink scheduling and to map logical channels from the RLC layer into transport channels for the physical (PHY) layer. These channels carry different kinds of data: control data, data from the IP packets etc. Because there are more logical channels than transport channels a mapping is required.

#### Hybrid Automatic Repeat Request

The Hybrid Automatic Repeat Request (HARQ) functionality keeps track of the sent data until it has been delivered successfully. When the sender receives an acknowledgement (ACK) from the receiver the data is considered delivered successfully. When a negative acknowledgement (NACK) is received from the receiver the redundancy version is increased and data is sent again. The redundancy version affects which bits are sent and at the receiver the received soft bits are combined
with previous received soft bits which increase the chance for a successful decoding with every retransmission. In LTE a maximum of three retransmission is allowed, after that the transmission is considered to have failed[3].

**Downlink scheduler**

In LTE there are two ways to schedule resources, Time-Division-Duplex (TDD) and Frequency-Division-Duplex (FDD). When using TDD resources are only scheduled in time while in FDD resources are scheduled both in time and frequency. In this report only FDD is considered and in FDD one radio frame is 10ms long and consists of 10 subframes á 1ms. The scheduler schedules one subframe, also called transmission time interval (TTI), per scheduling. These subframes then each consists of two slots á 0.5ms, where each slot includes a number \(N_{RB}\) of resource blocks (RB) depending on the used bandwidth. See table 4.2 for possible number of RB. These RBs are the smallest element that can be scheduled. The RB consists of 7 (or 6 if extended CP is used) time slots and 12 subcarriers; each of these \(7 \times 12\) slots are called resource elements and can be filled with a OFDM symbol. The RBs are 180 kHz wide with a 15kHz spacing between subcarriers while the OFDM symbols have a duration of \(\sim 66.6\mu s\) (plus CP of \(\sim 4.7\mu s\)). The sum of all subcarriers bandwidth \(BW\) is called transmission bandwidth \(BW_{config}\) and the sum of all subcarriers and spacings is called channel bandwidth \(BW_{channel}[14]\).

<table>
<thead>
<tr>
<th>(BW_{channel}[\text{MHz}])</th>
<th>(BS_{config}[\text{MHz}])</th>
<th>(N_{RB})</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4</td>
<td>1.08</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>2.7</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>4.5</td>
<td>25</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>50</td>
</tr>
<tr>
<td>15</td>
<td>13.5</td>
<td>75</td>
</tr>
<tr>
<td>20</td>
<td>18.0</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 4.2. The available bandwidths and number of resource blocks[14].

Continuously, or when ordered to, the UE sends reports back to the BS describing how good the channel is on each RB with a value, so called channel quality indication (CQI). The UE calculates the CQI value out from the received Signal-to-Noise-plus-Interference (SINR) and at the BS it translates into which modulation and how much punctuation shall be performed. The CQI values can be reported for groups of RBs, 1-4 RB in each resource block group (RBG), one value for each RB or one value for all RBs. The CQI values can be delivered as offsets from an average or as absolute values[4].

So what the downlink scheduler has to decide are which UE should be scheduled on which RBs, what CQI value should be used for each UE and which packets from the scheduled UEs should be sent. In MIMO mode the scheduler also has to chose MIMO mode and which pre coding to use in the antenna mapping. The goals
Figure 4.3. Resource frame structure for FDD in LTE.
the scheduler should have are that the QoS rules of the EPS-bearers should be followed, UEs with good channels should be prioritized but at the same time keep fairness among UEs, one CQI value must be chosen out from the different reported ones and at the same time you want a as high throughput as possible for the BS. To be able to satisfy all of these targets one probably has to count on some kind of trade off between them.

Once UEs has been allocated to each RB and CQI values set for the UEs; the amount of data available to be sent for each UE is picked depending on the CQI value and the number of RBs allocated to that UE[4]. The data to each UE forms a transport block.

4.5.4 Physical layer

The physical (PHY) layer receives the data to be sent from the MAC layer in transport blocks (TB) which then are prepared to be transmitted. The TBs are also called codewords and normally only one is transmitted per transmission time interval (TTI) but if several transmitting- and receiving antennas are used, up to two codewords are possible. This implies that the same UE can use different modulation and code rate on different codewords.

Cyclic redundancy check

A 24 bit long cyclic redundancy check (CRC) code is added to the TB to enable the receiver to verify that the correct bits has been received. The TB is then divided into smaller blocks, so called code blocks (CB). The code blocks have a maximum size of 6144 bits and another 24 bit CRC code is added to each CB[3].

Turbo Code

Each CB is run through a turbo encoder which consists of two recursive systematic convolutional encoders with parallel concatenation[14]. Each of the convolutional encoders has the stream into the encoder and a parity stream. The input into the second encoder is a interleaved version of the input into the first encoder. Only the two parity streams, the input to the first encoder and the termination bits(last four bits) into the second encoder are used and concatenated into three streams[3]. These three streams are then interleaved and concatenated according to [3] and stored in a cyclic buffer.

Rate matching

In order to get the code rate that the CQI number indicates all data cannot be sent; therefore rate matching is performed. The CQI number indicates how much data is to be sent and this amount of data is read out from the cyclic buffer in a sequence starting from a position depending on the current redundancy version. The redundancy version is increased for every transmission, therefore different data will be sent every retransmission[3]. In figure 4.4 the first steps of the PHY layer is shown.
Figure 4.4. First steps of the PHY layer in E-UTRAN.
4.5 Evolved Universal Terrestrial Radio Access

Scrambling and modulation

A cell unique scrambling is applied to the data to ensure that interfering signals after descrambling are totally randomized [14]. The scrambled data are then modulated with the modulation indicated by the chosen CQI number; where QPSK, 16QAM and 64QAM are the possible modulations.

Layer mapping and precoding

When using several antennas, so called MIMO, it is possible in theory to send number of transmitting antennas times the number of receiving antennas more data compared to one transmitting- and one receiving antenna but in reality this is not possible. In LTE the maximum number of allowed antennas is four transmitting and four receiving[10], this setup makes it possible to use up to four layers, which means that at least four times as much data can be scheduled. The codewords are mapped into layers and then mapped into antenna ports by the precoding. This codeword to layer to antenna port not only depends on the number of antennas but also which of the eight MIMO modes that is used. The precoding is a matrix operation where the matrix is decided out from the precoding matrix indication (PMI) provided by the UE[14].

Resource element mapping

When the codewords have been mapped into antenna ports the complex symbols (OFDM symbols) are mapped into the correct resource element, in time and frequency, which they were scheduled to by the resource scheduler.

Transmitting preparations

Before the the symbols are transmitted they are transformed into time domain by IFFT, CP is added to the symbol streams, the signal is upconverted from baseband to the carrier frequency and amplified to get the correct transmitting power.

Receiver

The steps in the receiver PHY layer are quite straight forward, doing the opposite of the transmitter’s PHY layer. One big difference is the demodulation which need the state of the channel and the characterization of the interference. These are estimated out of from the reference signals (RS) which are transmitted on certain REs. The demodulator requires the estimated channel and then produces soft bits (float numbers) instead of bits. One way to allocate the soft bits are that positive values mean ones, negative values zeros and zeros mean the value is unsure. A large positive value means that it is very certain that it is a one while a very large negative value mean a sure zero. When restoring the rate out from the turbo encoder zeros are filled in those locations where no bits were received. These soft-bits help the turbo decoder to chose the correct path in the trellis decoding which increases its performance[3]. The remaining steps are also very straight forward
but it could be noted that a faulty turbo decoding will fail one of the CRC checks in the last steps and trigger a retransmission.

4.6 Long Term Evolution Advanced

Long term evolution advanced (LTE-A) is a true 4G mobile system and is basically just an extension of LTE. The final version of LTE-A is not finished yet (August, 2010) but is expected to be so in the end of 2010 or beginning of 2011. The goal of LTE-A is to increase both the peak data rate and the peak spectrum efficiency. The difference between LTE and LTE-A is shown in table 4.3 [10].

<table>
<thead>
<tr>
<th></th>
<th>Rel.8 LTE</th>
<th>LTE-A</th>
</tr>
</thead>
<tbody>
<tr>
<td>DL peak data rate</td>
<td>300 Mbps</td>
<td>1 Gbps</td>
</tr>
<tr>
<td>UL peak data rate</td>
<td>75 Mbps</td>
<td>500 Mbps</td>
</tr>
<tr>
<td>DL peak spectrum efficiency</td>
<td>15</td>
<td>30</td>
</tr>
<tr>
<td>UL peak spectrum efficiency</td>
<td>3.75</td>
<td>15</td>
</tr>
<tr>
<td>Max bandwidth</td>
<td>20 MHz</td>
<td>100 MHz</td>
</tr>
<tr>
<td>Max downlink layers</td>
<td>4</td>
<td>8</td>
</tr>
</tbody>
</table>

Table 4.3. Comparison of LTE and LTE-A.

There are several extra mechanisms that are proposed to increase the performance of LTE-A. One is coordinated multi-point (CoMP) transmissions, which basically means that several BS coordinate and can all transmit to a certain UE[10]. Another important part of LTE-A is other types of transmitting nodes; relay nodes and femtocell nodes. Relay nodes are either deployed within the macro cell to increase the signal strength at cell-edges or outside the service area of the BS and from the UE forms a new macro cell. In both cases the relay is connected to the BS through a wireless connection outside the normal transmission spectrum[6]. Femtocells are described in the next chapter.
Chapter 5

Femtocells

This chapter mainly describes the femtocell access point (FAP) concept from a technical point of view but also include some remarks on the business challenges and opportunities. The business view is important because of the expected limited profit margin on femtocells [21, 12] which will make the cost be the deciding factor in the final solution.

5.1 Concept overview

Femtocells has been a helping solution in most previous mobile systems [12]. Even though the femtocell concept is not new and its somehow mediocre penetration in the mass market [21] in the previous mobile technologies it is expected to be an important mechanism in the LTE networks and a must for LTE-A.

The femtocell is a home base station which is meant to be deployed in homes or companies to increase the mobile network capacity or give mobile network coverage where none exists. Most likely will all femtocells coverage only be available to its owners and the femtocell will work much like a WiFi router but will share the radio spectrum with the rest of the mobile network [22]. The backhaul for femtocells in LTE will most likely be an already owned broadband connection, like ADSL or fiber.

FAPs are considered cheap ways for mobile operators to increase their capacity and coverage of their networks but at the same time increase users’ data throughput [22]. The benefits from femtocells is often described as a "win-win" situation between mobile operators and users [22, 24, 12].

5.2 Business challenges

There are many challenges in order to make FAPs for LTE a commercial success in the broad consumer market. One fundamental obstacle is the manufacturing cost. Even though LTE FAPs would be far more advanced than a wifi router but to customers the delivered values are similar. Therefore would customers probably
not want to pay much more for a LTE FAP than a wifi router. Since many customers with a broadband connection already has a wifi router it is believed that a FAP deployment to the mass market will require the mobile operators to subsidize the FAP units in order to attract customers [21, 12]. In addition it is believed that it is important that calls made through the femtocell are somehow cheaper than using the mobile network because it is hard to motivate customers to pay the same amount for calling through their own FAP and broadband connection as if they were using the mobile network. This means that it will be a tough challenge for the mobile operators to get a high return on the investment when subsidizing femtocells.

5.3 Technology challenges

There are several challenges with femtocells for LTE that has to be addressed before they can be successfully deployed. One challenge is how will the femtocell synchronize with the rest of the mobile network using its broadband backhaul due to its lack of exactness[12]. Another difficulty the broadband backhaul is causing is that it makes it hard to in a reliable manner keep the QoS rules due to the unreliable delay and bandwidth[20]. This problem cease to exist if the mobile operator and the internet service provider (ISP) have an agreement or are the same company so the delays can be monitored and controlled [12]. BS keeps a list of its neighbors for handover purposes but how will the FAP do; how does it retrieve its position to send to others and how does it construct its neighbor list. It is proposed to use a GPS to acquire the FAP’s position [12]. AT&Ts currently offered femtocell for their 3G network (not LTE) uses a GPS to retrieve the position and therefore is required to be situated close to a window due the weak GPS signal’s lack of ability to penetrate house walls[9]. Many mobile operators would probably like the FAP to incorporate several generations of mobile systems, how could this be arranged? In LTE-A the FAP concept is considered a part of the system while other mobile systems require them to be added on top of the existing system. This means that the design of the femtocell has to adapt to the mobile system not the other way around. This leads us to the probably most challenging issue for LTE FAPs and what is focused on in this study, resource scheduling, and is described in the next section. The reason to why its so challenging is that most problems in BS scheduling also applies to FAP scheduling, in addition the FAP has additional issues and restrictions.

5.4 Resource scheduling for femtocells

There are two main strategies to what resources the FAPs should use, either the FAPs use an additional bandwidth spectrum or the FAPs use the same spectrum as the rest of the mobile network [24]. Since additional spectrum is very expensive to acquire for the mobile operator the general opinion is that FAPs should share the spectrum with the rest of the network. The serving radius of a LTE FAP is expected to be around 20-40m, this short distance will enable the FAP to provide
a strong signal to the users even with low transmitting power, due to small path losses. The strength of the signal from the FAP to the UE is limited by interference from other FAPs and BS. Even more important is that the FAPs transmissions do not interfere with the transmission from the BS to its UEs. These interference scenarios are illustrated by figure 5.1. In figure 5.1 the BS is transmitting to its UEs while one of them is within the coverage area of the FAP1 which will interfere the transmission to the UE. The way to avoid interfering with the transmissions from the BS in LTE includes a strategy on which RBs to use and how much power to use. This planning is possible to perform either through a central unit planning several FAPs’ scheduling or each FAP planning its own resource usage [24]. The later is regarded as a more practical solution since it becomes a very complex problem to schedule a large number of FAPs and also these scheduling decision would be more delayed and therefore inaccurate. In [24] it is presented that the available spectrum can be divided into segments, then each FAP randomly chose one of the segments. This scheme decrease the chance adjacent FAPs to schedule on the same resources and interfere each other. When performing scheduling while sharing the spectrum the FAP first has to retrieve the channel state information (CSI) somehow. The two options are either that the UEs sends CQI reports to

![Figure 5.1. The different interference scenarios with FAPs involved.](image-url)
the FAP, exactly like they do to the BS, or the FAP itself senses the current CSI. This sensing could be performed with either the same antenna as for transmitting which decreases capacity or another antenna could be used but would increase the manufacturing cost of the FAP[24]. However it is possible that a UE is interfered but the sensing FAP does not sense this interference and will then faulty schedule the interfered UE.

The are several algorithms to schedule the resources for FAPs [24, 33], most are heavily influenced by simple schedulers for the BS[31]. A very simple scheduler is the Round-Robin scheduler which just in turn schedules the UEs. Even this simple scheduler can serve its purpose if the channel on each RB fluctuates a lot, because then the reported CSI is invalid when it is about to be used. Another popular scheduler in mobile systems is the proportional fair scheduler. The basics of it is simple, the users with good channel and small historic throughput are prioritized[31]. The implementation on an LTE system requires more mechanisms which will affect the performance of the scheduler, like how to choose the used CQI or if one should be able to schedule all RBs to one UE. Therefore can schedulers named as proportional fair schedulers have widely different performance in LTE. Further, there are many proposed schedulers aimed for BS that tries to ensure the QoS requirements, for example [25, 30, 34]. Ensuring the QoS requirements is a very important feature in a LTE schedulers. Therefore this study focus on schedulers that are considering QoS requirements to schedule users for femtocells. This is described further in the next chapter.
Chapter 6

QoS scheduling for LTE femtocells

As mentioned in the previous chapter there are several approaches to scheduling in LTE but in this study we focus on schedulers which consider the QoS requirements when scheduling. There has been some contributions when it comes to QoS aware schedulers for BS, [25, 30, 34], but when it comes to femtocells there is a limited number of contributions [24, 33]. The reason to this might be that femtocells are a minor area within LTE or it might be that scheduling in the femtocell seems to be the same problem as for BS but on the contrary there are a couple of fundamental differences between the two. One difference is that it is very difficult for the FAP to retrieve information about who are interfering with its transmissions and on which resources and how should the FAP schedule its users without interfering with other users.

6.1 Previously proposed QoS aware schedulers

Almost all QoS aware schedulers is somehow inspired by the classical proportional fair (PF) scheduler. So to understand how most QoS aware schedulers work it is important to understand the PF scheduler. It aims to provide fairness among users and to utilize the available RBs efficiently by choose the user with the largest metric[31].

\[ M_{PF} = \frac{R_k[m]}{T_k} \]  

(6.1)

\( R_k \) is the available throughput and the \( T_k \) the historical throughput averaged over a certain window size in time and the index \( k \) ranges over the number of UEs served by the base station while \( m \) indicates the current RB being scheduled.

One QoS aware proportional fair scheduler was presented in [30]. It was pro-
posed that in each RBG schedule the user with the largest metric:

\[ M_{QOS} = \frac{C_{s,k}}{B_k^\kappa} \]  

\[ B_k = \frac{A_k}{T_k R_k} + \epsilon \]  

where \( A_k \) is the historical throughput over \( T_k \) which is the QoS delay target. The \( R_k \) is the QoS target bit rate, \( \epsilon \) is a small positive regularization term.\(^{[30]} \), \( C_{s,k} \) is the minimum supported rate for the GBR and \( \kappa \) is a additional fairness parameter. \( B_k \) works as a measurement on how good the QoS requirements are fulfilled to make sure EPS-streams below their target rate to be prioritized while at the same time users with very good current channel conditions are prioritized. This metric seems very good and there is several similar scheduling metrics proposed, for example in \([26, 18]\) but none of them differentiate the all QoS requirements between streams with guaranteed bit rate and those without. One contribution with a similar metric which consider the high priority for streams with guaranteed bit rate is \([34]\). What they all fail to give a solution to is how to decide which modulation and code rate should be used. This is a problem when each UE report different CQI numbers on each RBG but still one single modulation and code rate must be used for each UE. In scheduling for femtocells the problems faced for base stations remains but additional issues appear, where the most critical issues are to know which RBs are available for the FAP to use and how much power should be used in transmission without interfering other users.\(^{[24, 22]}\).

### 6.2 Reference schedulers

In order to show the performance of the proposed scheduling algorithm a reference scheduling algorithm has been implemented. The reference algorithm is based on the previously mentioned proportional fair scheduler together with some design decisions to make it work in LTE.

#### 6.2.1 LTE Proportional fair scheduler

The implemented LTE proportional fair (LPF) scheduler uses the metric described in 6.4 where \( C_k \) is the CQI value for user \( k \) and \( T_k \) is the historical throughput for user \( k \). The reason behind using the CQI value is that it is relatively proportional to the amount of data that can be sent. The LPF is shown in pseudo code below in algorithm 1. The HARQ buffer’s packets are scheduled first using algorithm 2, then the user with the highest PF metric is scheduled on each RB using algorithm 3 and finally the packets are scheduled using algorithm 4. On line 1 in algorithm 2 the CQI values used in the previous transmission is loaded, this enables that a retransmission never uses higher CQI value than a previous transmission which gives a lower risk for a failed transmission. On line 5 in algorithm 3 limits the BS to not schedule more than 75% of the RBs to one single user, this makes sure that
a retransmission is almost always possible even with a lower reported CQI value.

\[ M_{LPF} = \frac{C_k \times 0.5 + 1}{T_k} \]  \hspace{1cm} (6.4)

Algorithm 1 LTE Proportional fair scheduler

for all HARQ_TBs do
    Schedule HARQ retransmissions using algorithm 2
end for

for all RB do
    Schedule the best UE using algorithm 3
end for

for all UE do
    Schedule packets using algorithm 4
end for

Algorithm 2 Scheduling of retransmissions

\[ CQI\_LEVEL = CQI \] value used in last transmission
\[ NR\_OF\_USABLE\_RBs = 0 \]

while \( CQI\_LEVEL >= 0 \) do
    for all RB do
        if \( CQI < CQI\_LEVEL \) \& RB not used then
            \( NR\_OF\_USABLE\_RBs = NR\_OF\_USABLE\_RBs + 1 \)
        end if
        if \( CQI\_LEVEL \) \& \( NR\_OF\_USABLE\_RBs \) enables transmission of the TB then
            Schedule the UE on the RBs counted for in \( NR\_OF\_USABLE\_RBs \)
        end if
    end if

    Schedule the packets in the current TB
    \( CQI\_LEVEL = -1 \)
end if
end for

if TB not scheduled then
    \( CQI\_LEVEL = CQI\_LEVEL - 1 \)
end if
end while
6.3 Proposed QoS-aware scheduler

In this section we present our proposed scheduling algorithm which has been designed to consider both the needs from GBR EPS-bearers and opportunistic scheduling. The core consists of the metric $M_{LTE\ QOS,k}$ which influence which user to be scheduled on each RB and the mechanism that chooses which packets to schedule. The metric of the proposed scheduler is inspired by the by the metric presented in [34] and is presented in equation 6.5 where $\tau$ is delay in ms, $R$ is rate in bps, $D_k$ is the number of non-GBR EPS-bearers for $k^{th}$ UE, $C_{rep}$ is reported CQI value for current RB, $C_{avg}$ is CQI average over time, $k$ which UE and $N$ are the numbers of RB allocated so far this round. The sum $A_k$ gives high values
to UEs with EPS-streams (with GBR) that lies below requirements on data rate and delay. Here the $\gamma$ constant is to give different weight the two requirements but is initially set to $\gamma = 0.5$. $N$ ensures that UEs that currently have few RBs assigned are prioritized before those with many. The $B_k$ term is similar to a PF metric but uses CQI instead of data rate in order to favor users with better channel conditions (than their used average). The usefulness of such a CQI fractal is comparable with a data rate fractal because the CQI values are almost linear to the amount of data you can transmit. In addition using the CQI values requires a lot less computations. The $\beta$ constant is to balance the metric between resource need ($A_k$) and opportunistic scheduling ($B_k$), it is initially set to $\beta = 0.5$.

$$A_k = \sum_{l=0}^{\forall GBR} \left( \frac{s_{l}^{avg}}{s_{l}^{max}} \gamma + \frac{R_{l}^{min}}{R_{l}^{avg}}(1-\gamma) \right) + D_k$$  \hfill (6.5) \\

$$B_k = \frac{C^{rep}_k}{C^{avg}_k}$$  \hfill (6.6) \\

$$M_{LTE \ QOS,k} = \frac{A_k}{N} \beta + B_k(1-\beta)$$  \hfill (6.7)

**Algorithm 5** Proposed QoS-aware scheduler

```plaintext
for all HARQ_TBs do
    Schedule HARQ retransmissions using algorithm 2
end for
for all RB do
    Schedule the best UE using algorithm 6
end for
for all Scheduled UE do
    Schedule packets using algorithm 7
end for
```
Algorithm 6 Finding the best UE for QoS scheduler.

\[ \text{BEST\_METRIC} = -2 \]

\[ \text{SMALLEST\_CQI} = (28, \ldots, 28), \text{size} = \text{NR\_OF\_UE} \]

for all \text{UE} do

\[ \text{if } \text{UE is not retransmitting then} \]

5: \[ \text{if } \text{ALLOC\_RES(UE)} < \text{size of all waiting packets then} \]

if \text{UE has not been scheduled on more than 75\% of the RBs then} \n
\[ \text{if } \text{reported CQI on RB } \geq \text{ average of reported CQI then} \]

\[ \text{TEMP\_METRIC} = M_{LTE\ QOS,k} (6.5) \]

end if

10: \[ \text{if } \text{TEMP\_METRIC} > \text{BEST\_METRIC then} \]

\[ \text{BEST\_METRIC} = \text{TEMP\_METRIC} \]

\[ \text{BEST\_UE} = \text{UE} \]

end if

end if

end if

end for

\[ \text{ALLOC\_RES(BEST\_UE)} = F(\text{SMALLEST\_CQI(BEST\_UE)}, \text{NR\_OF\_ALLOC}) \]
Algorithm 7 Schedule packets for QoS scheduler.

if $UE$ is not retransmitting then
  if $ALLOC\_RES(UE) > 0$ then
    for all $WAITING\_PACKETS$ do
      if $WAITING\_PACKET\_remaining\_delay\_budget < 5\text{ms}$ then
        Schedule $WAITING\_PACKET$
        $ALLOC\_RES(UE) = ALLOC\_RES(UE) - WAITING\_PACKET\_size$
      end if
    end for
  end if
end if

for all $EPS - streams$ with GBR do
  if $\frac{Delay_{avg}}{Rate_{avg}} > 0.9$ or $\frac{Rate_{avg}}{Rate_{min}} > 0.9$ then
    Schedule $WAITING\_PACKET$
    $ALLOC\_RES(UE) = ALLOC\_RES(UE) - WAITING\_PACKET\_size$
  end if
end for

for all $WAITING\_PACKETS$ do
  if $WAITING\_PACKET < ALLOC\_RES(UE)$ then
    Schedule $WAITING\_PACKET$
    $ALLOC\_RES(UE) = ALLOC\_RES(UE) - WAITING\_PACKET\_size$
  end if
end for


6.4 Simulation results

In this section the results from three simulations using the two FAP schedulers and one simulation without FAPs are presented. In the comparison we use the simulation settings in table 6.1 and the nodes was placed as in figures 6.1 and 6.2. The three simulations are compared at three levels; for the whole macro cell, for the BS and for the average FAP. The reason to why only the last half of the simulation is showed in plots is that it takes some time before the buffer sizes converge to stable sizes; which can also explain the slight constant increase/decrease in some results.

6.4.1 Throughput

Here we study three types of throughput, total for macro cell, for BS and average among FAPs, these are shown in figure 6.3. The total throughput in macro cell is interesting from a mobile operator perspective, while the throughput for BS and average FAP are mostly interesting from the UEs served by BS respectively UEs served by FAPs.
### Table 6.1. The simulation settings for comparison of the two FAP schedulers.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>LPF</th>
<th>LTE QoS</th>
<th>No FAPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>#BS</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>#FAP</td>
<td>8</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>#UE served by BS</td>
<td>32</td>
<td>32</td>
<td>48</td>
</tr>
<tr>
<td>#UE served by FAP</td>
<td>2/FAP</td>
<td>2/FAP</td>
<td>0</td>
</tr>
<tr>
<td>BW</td>
<td>5 MHz</td>
<td>5 MHz</td>
<td>5 MHz</td>
</tr>
<tr>
<td>#RB</td>
<td>25</td>
<td>25</td>
<td>25</td>
</tr>
<tr>
<td>Noise level</td>
<td>-150</td>
<td>-150</td>
<td>-150 dBm</td>
</tr>
<tr>
<td>BS max power</td>
<td>46dBm</td>
<td>46dBm</td>
<td>46dBm</td>
</tr>
<tr>
<td>FAP max power</td>
<td>20dBm</td>
<td>20dBm</td>
<td>20dBm</td>
</tr>
<tr>
<td>Simulation time</td>
<td>1200 ms</td>
<td>1200 ms</td>
<td>1200 ms</td>
</tr>
</tbody>
</table>

**Figure 6.1.** The positions of all nodes in the first two simulations. The five-cornered star is the BS, the circles are UEs served by BS, the six-cornered stars are FAPs and the crosses are UEs served by FAPs.
6.4 Simulation results

In figure 6.3 we can observe that the use of FAPs almost doubles the total throughput in the macro cell while at the same time the two schedulers give about the same throughput for the macro cell. In the second plot in figure 6.3 it is clear that when some UEs are served by FAPs the throughput of the BS is decreasing. Whether this decrease depends on that the BS serves less UEs, perform less optimal scheduling with fewer UEs or if it is because of the interference from the FAPs is hard to tell without further more thorough studies. Another interesting observation is that the proposed QoS scheduler provides higher throughput for the FAPs than the LPF do but at the same time when using the LPF the BS is providing higher throughput than when using the proposed QoS scheduler. The largest surprise was that the throughput of the proposed QoS scheduler was almost identical to the LPF scheduler.

6.4.2 Delay

The delay is depending on the throughput, the packet arrival rates and the buffer sizes. This means that if a scheduler provides high throughput it is likely to provide low delay but only when there is a controlled packet arrival rate in the buffers. Figure 6.4 shows that when not using FAPs the delays are lower; this can be explained by that when using FAPs the throughput is larger so more EPS-bearers are admitted which leads to larger buffer sizes and therefore higher delays. The difference in delay between the two schedulers is small and therefore it is hard to
Figure 6.3. Throughput at macro cell, BS and average FAP for the three simulations.
draw any conclusions regarding it.

![Graphs showing packet delay across different cells and FAPs for three simulations.](image)

**Figure 6.4.** Average packet delay at macro cell, BS and average FAP for the three simulations.

### 6.4.3 PER

Since the PER in the all the simulations were zero which is quite reasonable since the maximum allowed PER is between $10^{-3}$ and $10^{-6}$. The packet error rates are of course mainly depending on the SINR, which we cannot control but the factors we can control are the used demodulator, turbo decoder and what mapping between CQI numbers and SINR are used. It is possible that the SINR levels were mapped too conservatively which means that the throughput and PER are lower than what would be the case with a more moderate mapping.

### 6.4.4 UE satisfaction

In this part we find the most surprising result because when designing the QoS scheduler the idea was to enable more UEs to be satisfied. On the contrary, the simulation results in figure 6.5 show that the LPF scheduler is able to provide higher satisfaction than the proposed QoS scheduler. The case without FAPs that received even higher user satisfaction is likely to be correlated with the lower throughput and less filled packet buffers.
Figure 6.5. User satisfaction at macro cell and BS for the three simulations.
The simulations results showed us that FAPs can provide a substantial increase in macro cell throughput while at the same time not interfere UEs served by the BS (since PER was zero in all simulations). The proposed QoS scheduler did not give the anticipated results but was still providing performance quite similar to the implemented proportional fair scheduler. The reasons behind the unexpected results are hard to prove without a wider and more detailed study, however one possible explanation for the surprising result is that the EPS-bearer admission mechanism is not properly adjusted which could give unbalanced packet arrival rates and buffer sizes.
Chapter 7

System simulation tool

This chapter describes the implementation of the system simulation tool for LTE with femtocells. This simulation environment enables good comparisons between different types of FAP schedulers. In addition, it is capable of illustrating the effectiveness of BS schedulers. For more details than provided in this chapter see Appendix A and the source code. It should be noted that this simulation tool is based on both 3GPP’s rel.8 and rel.9 documents. Rel.8 is the final release of the LTE standard but lack any FAP features. Rel.9 which is considered a slight extension of the Rel.8 and a preparation for LTE-A do include some support for FAPs.

7.1 Goals

One can perform LTE system simulations with many different goals and one of the most common one is probably just to measure the throughput at different points in the system. The goal of this simulator is rather to enable studies of the coexistence of FAPs(HeNodeB) and BS(eNodeB). Further, another important goal with this simulator is to enable comparisons of different schedulers, even with different schedulers in different nodes. One important part in the study of the FAP-BS coexistence is to monitor not only the throughput but also the user satisfaction. One should notice that user satisfaction is no absolute term but it is mentioned in sections 7.8 and 7.12.1 how user satisfaction is defined in this study. The user should be aware of that the purpose of the scheduler is for comparable simulations not to simulate the absolute performance of an environment with FAPs.

7.2 Framework

In order to simulate the real world we must create models to describe its behavior, but doing so in a very precise manner often demands quite complex models. In order to keep the complexity to a minimum we therefore need to make assumptions and simplifications about the real world, these are presented in 7.2.1.
7.2.1 Assumptions and simplifications

All the assumptions found in this section tries to simplify the behavior of the system and the environment. This is important in order to find a reasonable amount of functions which could be implemented within the time frame of this project while at the same time have enough functions and preciseness to be sure that the results have value in the real world.

The environment

Interference
Except for the other Enodes (refers to both FAPs and BS in this report) and a background noise there is no other source of interference in the environment. This is reasonable since it is the problem of FAPs and BS interfering each other that is studied here.

Movement
All UEs and all Enodes are stationary even though movement is partially supported in the code. Movement would not increase the value of the results.

Coverage area
The coverage areas of several BS are not overlapping and there are no UEs positioned in the area not covered by the BS. The difference is close to nonexisting since BS still interfere outside their coverage area.

BS Positions
All BS are place in hexagon patterns, i.e. seven BS are placed in a hexagonal pattern in the environment, which means each BS has six neighbors.

UE Positions
The positions of the UEs are limited to within the coverage area (x,y-coordinates) of BSs and FAPs. Further, the z-coordinate is limited between the height of the BS and the specified ground level for UEs.[6]

FAP Positions
FAPs are only placed within the one third of the coverage area of the BS which is furthest away since it is a reasonable assumption to think that users close to the BS would not need a FAP due to an already good coverage. Even if this is set as default it is possible to run simulations if the FAPs positions are changed.

EPC

Backhaul
The backhaul of the FAPs does not have a delay greater than the accounted delay in the EPS-bearers delay budgets, 20ms. See section 7.8 for more on this.
7.2 Framework

Information sharing
The BS is assumed to be able to share its scheduling spectrums to the FAP and therefore the FAP has this information available.

Hand-overs
Since there is no movement there is no need for hand-overs. Hand-overs are not expected to have any large impact on the scheduling performance.

Packets
The size of the packets inside the mobile network are smaller than reality due to that we receive the fragmented packets from the RLC layer and also the simulator was forced to use smaller packets because of the smaller capacity when only using SISO and the stray for a smooth flow of packets in the system. If the packets were too big they would become hard to schedule in an efficient manner.

Link layer

Computation capacity
The computation capacity of the BS, FAP and UE nodes have no impact on the link layer performance.

PDCP layer
The PDCP layer is considered to have no impact on the performance of the simulation.

Synchronization
All BS and FAPs are synchronized and also all transmission are automatically synchronized.

Resource mapping
The modulated data is mapped into RE without considering the placement of control signal and reference signals. They are only accounted when calculating how much data can be mapped.

Channel
The channel coefficients are generated in frequency domain therefore no IFFT or FFT is needed on the transmitted symbols. In addition, the channel estimation is considered to be ideal. There is always a background noise present.

Data flow
Only downlink is considered in this simulation therefore will data only flow from the applications to the UE.

CQI reports
The CQI reports are ideal and available at BS after a delay of 8ms as in [23].
Antennas

Only the SISO, one transmitting antenna and one receiving antenna, case is considered. The use of MIMO would increase the value of the results in this report but since MIMO would also add too much coding complexity it has been left out.

7.3 Implementation environment

The simulation tool is implemented in C++ in the Microsoft Visual Studio 2008 developing environment. This developing environment includes text editor, compiler, linker and debugger. In addition to C++’s standard libraries, the IT++ library [27] has been used. The IT++ library contains mathematical, signal processing and communications classes and functions,[27]. The IT++ itself is using either Intel’s or AMD’s mathematically libraries which also has to be installed which is described in [27].

7.4 Overview

7.4.1 Class overview

The simulation tool is built of numerous C++-classes where each class tries hold a certain mechanism in the system simulation with the class’ properties and functions. All classes are listed in table 7.1 and each class is furhter described in Appendix A.

7.5 Initiating the environment

When initiating a simulation an object of the SimEnvironment class is created with the following inputs: number of BS, number of FAPs within the coverage area of each BS and number of total users within BS coverage area (BS UEs plus FAP UEs). The process is illustrated in Figure 7.1 and the modeled environment in Figure 7.2. When the SimEnvironment object is created it sets its properties, for example it fills its lists of the BS in the environment with given number of BS and in the same way for FAPs. Table 7.2 shows the values of the most important properties and see Appendix A for a complete property list. Each Enode created then also sets numerous properties such as a list containing all UEs which belong(are connected) to them, the antenna configuration of the Enode, and positions. The most important properties for the BS and FAP are showed in Table 7.3 and Table 7.4. Each UE that the Enodes create is also given properties and the key values set in each UE is displayed in Table 7.5.
<table>
<thead>
<tr>
<th>Name in code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AntConfig</td>
<td>Handles the antenna configurations in each node. This class lack importance when the IT++ channel is used.</td>
</tr>
<tr>
<td>BS</td>
<td>Handles the functionality of the BS and is derived from the Enode class.</td>
</tr>
<tr>
<td>ChannelModel</td>
<td>Calculates the channel used in transmissions, one version is based on [6] and [11]. Due to the amount of computations in the first model a simpler fading generator from IT++’s library is used in simulation.</td>
</tr>
<tr>
<td>CQI</td>
<td>Holds the information in a CQI-report and some translations from CQI number.</td>
</tr>
<tr>
<td>Enode</td>
<td>Base class for all nodes transmitting in DL, such as BS and FAP. Derived from the Node class.</td>
</tr>
<tr>
<td>EnodeScheduling</td>
<td>Holds all information and functions connected to a round of scheduling for one Enode.</td>
</tr>
<tr>
<td>EPSstream</td>
<td>Holds the properties of the EPS streams.</td>
</tr>
<tr>
<td>FAP</td>
<td>Handles the functionality of the FAP and is derived from the Enode class.</td>
</tr>
<tr>
<td>Node</td>
<td>Holds all properties and functions in common for all nodes. Base class for Enode and UE.</td>
</tr>
<tr>
<td>Packet</td>
<td>Keeps the properties of a packet such as how much is left of the delay budget.</td>
</tr>
<tr>
<td>Scheduler</td>
<td>All schedulers are implemented as functions in the scheduler class.</td>
</tr>
<tr>
<td>SimEnvironment</td>
<td>Top level class (except for main) which hold the properties of the environment, the instantiations of the other classes and runs the simulation loop.</td>
</tr>
<tr>
<td>Transportation</td>
<td>This class includes all functions in the physical layer including for applying the channel and also keeps the values at each step of the transmission and reception.</td>
</tr>
<tr>
<td>UE</td>
<td>Keep the properties and functionality of the UE. Derived from the Node class.</td>
</tr>
<tr>
<td>UEScheduling</td>
<td>Holds all information and functions connected to a round of scheduling for one UE.</td>
</tr>
</tbody>
</table>

Table 7.1. The classes of the simulation tool.
SimEnvironment(#BS,#FAP/macrocell,#UE/macrocell)

Set global parameters

Create #BS

Create #Femtocells

Create # of UEs where

#UE/FAP = P(UE_in_FAP) * (#UE/macrocell) / FAP/macrocell

Create # of UEs where

#UE/BS = #UE/macrocell - #UE/FAP * FAP/macrocell

Set parameters for each UE

Set initial transmitting power on each RB (RS)

Generate EPS-bearers

Fill packet queue with packets from the EPS-bearers

**Figure 7.1.** The initiation process of the simulation environment.

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bandwidth</td>
<td>1.4-20 GHz</td>
</tr>
<tr>
<td>Number of RB</td>
<td>6-110</td>
</tr>
<tr>
<td>Ideal CQI</td>
<td>Yes</td>
</tr>
<tr>
<td>Ideal SINR</td>
<td>Yes</td>
</tr>
<tr>
<td>Noise level</td>
<td>-150dBm</td>
</tr>
<tr>
<td>Max interference distance - BS(^a)</td>
<td>2000m</td>
</tr>
<tr>
<td>Max interference distance - FAP(^a)</td>
<td>500m</td>
</tr>
<tr>
<td>Max interference distance - UE(^a)</td>
<td>500m</td>
</tr>
</tbody>
</table>

\(^a\) The nodes are considered to not interfere beyond this distance.

**Table 7.2.** The set values of the most important properties of the SimEnvironment class.
Figure 7.2. The positioning of the nodes in the simulation. The inner circle around the base stations describes the minimum distance to the BS a UE can be positioned. The middle dotted line describes the minimum distance to a BS a FAP can be positioned. The alignment of the BS sectors is also shown in the figure.

<table>
<thead>
<tr>
<th>Position</th>
<th>So that all BS nodes form a hexagonal pattern.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Antenna configuration</td>
<td>3-sector antenna</td>
</tr>
<tr>
<td>Maximum allowed power</td>
<td>46dBm</td>
</tr>
<tr>
<td>BS radius ($R_{BS}$)</td>
<td>500m</td>
</tr>
<tr>
<td>RB Group (RBG) size</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 7.3. The set values of the most important properties of the BS class.

<table>
<thead>
<tr>
<th>Position</th>
<th>At distance $\sim Uni[0.75R_{BS}, R_{BS}]$ from BS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum allowed power</td>
<td>20dBm</td>
</tr>
<tr>
<td>FAP radius ($R_{FAP}$)</td>
<td>40m</td>
</tr>
<tr>
<td>RB Group (RBG) size</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 7.4. The set values of the most important properties of the FAP class.
<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Position</td>
<td>At distance $\sim \text{Uni}(BS/FAP_{\text{min \ dist}}, R_{BS/FAP})$</td>
</tr>
<tr>
<td>Delay of CQI reports$^a$</td>
<td>8ms</td>
</tr>
<tr>
<td>EPS-streams</td>
<td>$\sim N(k(UE/Enode), 2)$ number of EPS-streams of uniformly distributed kind, a UE has either GBR or non-GBR EPS-bearers but not both.</td>
</tr>
</tbody>
</table>

$^a$ The time it takes from that a signal is transmitted until the quality of that signal is known.

Table 7.5. The set values of the most important properties of the UE class.

### 7.6 Channel model

This simulation environment includes two channel models, one is based on the channel model defined by 3GPP in [6] which is based on the one in the WINNERII-Project[11] The other one is the TDL channel from IT++ using the ITU Pedestrian A channel model. Due to the extensive computations required by the implemented 3GPP channel it is preferred to use IT++’s fading channel for large simulations. Therefore it is the IT++ channel that has been used in simulations throughout this report.

#### 7.6.1 Path loss

Both channel models use the same path loss model and model to generate if there is line of sight (LOS) which both are defined in [6]. The path loss depends on the distance between Enode and UEs, scenario and antenna heights. The different pathloss functions (depending on scenario) is given in Table B.1.2.1-1 in [6] with a addition a further path loss of 20dB from walls and it is assumed to be a wall in the way if the transmitter is outside and the UE inside as well as the reverse. As default the indoor hotspot (InH) environment is used for FAPs while the urban macro (UMa) is used as environment for the BS.

#### 7.6.2 3GPP Channel

Even though the complete simulation environment only support SISO many parts do support MIMO, the channel is one of them. In addition to the MIMO support, the implemented channel model supports all scenarios described in 3GPP’s channel model [6] and a 20 cluster reception. The channel is however calculated in several steps and these steps are shown in Figure 7.3. The features of the channel implementation also makes it usable in other implementations and as well if this tool is to be expanded to include MIMO.
Figure 7.3. The steps of the channel generation.
7.6.3 Line of Sight

Whether there is line of sight (LOS) or no line of sight between the Enode and each UE is decided by a random variable with probability depending on the used scenario and distance, described in Table B.1.2.1-2 in [6].

Large-scale parameters

The large-scale parameters generated in this step is the rms delay spread, rms azimuth spread of departure angles, rms azimuth spread of arrival angles, shadow fading and Ricean K-factor. In case of the InH scenario the parameters are Laplacian distributed otherwise Normal distribution with mean and variance depending on scenario, given in Table B.1.2.2.1-4 in [6]. These random variables are then first correlated with the same parameters of other UEs and then cross-correlated according to the cross-correlation parameters in Table B.1.2.2.1-4 in [6].

Generate delays

Depending on scenario and if the UE is in LOS, a number of delays $\tau_n$ is generated with cluster index $n$. The delays are generated as followed where $r_\tau$ is the delay distribution proportionality factor, $\sigma_\tau$ is the delay spread calculated in 7.6.3 and $X_n \sim \text{Uni}(0,1)$.[6]

$$\tau'_n = -r_\tau \sigma_\tau ln(X_n)$$  \hspace{1cm} (7.1)

Then the smallest of the $\tau'_n$ are subtracted to all other and sorted in descending order.

$$\tau_n = \text{sort}(\tau'_n - \min \tau')$$  \hspace{1cm} (7.2)

If the current path is in LOS the delays will also be scaled where $K$ is the Ricean-K factor.

$$\tau_n^{\text{LOS}} = \frac{\tau_n}{D}$$  \hspace{1cm} (7.3)

$$D = 0.7705 - 0.0433K + 0.0002K^2 + 0.000017K^3$$  \hspace{1cm} (7.4)

Generate cluster powers

The cluster powers depend on the delays generated in the previous step and is given by the following equation where $Z_n \sim N(0,\zeta)$[6]:

$$P_n' = \exp -\tau_n \frac{r_\tau - 1}{r_\tau \sigma_\tau} \times 10^{-\frac{z_n}{10}}$$  \hspace{1cm} (7.5)

Each cluster is containing $M$ number of rays and each of these rays are allocated the power $P_n/M$ where $P_n$ is the normalized power.

$$P_n = \frac{P_n'}{\sum_{\forall n} P_n'}$$  \hspace{1cm} (7.6)
7.7 Generate CQI reports

Departure and arrival angles

The last part is to determine the departure and arrival angles of each cluster which mainly depends on the already determined cluster powers and if LOS. Then each ray is given a departure and arrival angle as set offset from the cluster angles. The departure and arrival angles of the rays are then randomly coupled together. The rays are also given an initial phase which is uniformly distributed \((-\pi, \pi)\). For the exact procedure of generating departure and arrival angles see [6].

Generate channel coefficient $H$

The channel coefficients $h(t)$ in the channel model given in [6] has three dimensions; transmitter antenna, receiver antenna and cluster (also called taps in some literature [31]). Since only SISO is used in this tool there is only one transmitting antenna and one receiving antenna. Also a time frame of one millisecond is used. The calculation of channel coefficient $h(1\,\text{ms})$, given that $\phi$ is the angle from transmitter to receiver, $\varphi$ is the angle from receiver to transmitter, $F$ the antenna field pattern, $\Phi$ the phase of the ray, $\lambda_0$ the carrier frequency, $d_s$ transmitter antenna separation, $d_u$ receiver antenna separation, $\kappa$ the cross polarization power ration, $\upsilon$ the doppler frequency component and $n$ the cluster index.

\[
    h_n(t) = \sqrt{P_n} \sum_{m=1}^{M} \begin{bmatrix} F_V(\varphi_{n,m}) \\ F_H(\varphi_{n,m}) \end{bmatrix}^T \begin{bmatrix} \exp(\imath \Phi_{n,m}^{vv}) & \sqrt{\kappa} \exp(\imath \Phi_{n,m}^{hh}) \\ \sqrt{\kappa} \exp(\imath \Phi_{n,m}^{hv}) & \exp(\imath \Phi_{n,m}^{hh}) \end{bmatrix} \begin{bmatrix} F_V(\phi_{n,m}) \\ F_H(\phi_{n,m}) \end{bmatrix} \\
    \times \exp(\imath d_s 2\pi \lambda_0^{-1} \sin(\phi_{n,m})) \exp(\imath d_u 2\pi \lambda_0^{-1} \sin(\varphi_{n,m})) \exp(\imath 2\pi \upsilon_{n,m} t)
\]

(7.7)

7.7 Generate CQI reports

To be able to generate CQI reports to use in the scheduling there are several steps to go through. The first step is to generate the channel coefficients, next we use the channel in combination with assigned transmission power and calculated pathloss. From this we can then calculate a SINR ratio at each UE. This SINR ratio then translates directly into a certain CQI number.

7.7.1 Calculate Channel

The calculations of all receiving and interfering links are described in algorithm 8. For each link the received power and the total channel is calculated.

7.7.2 Calculate SINR and translate into CQI reports

The SINR or more precisely the "Signal-to-noise-plus-interference"-ratio for each UE and each RB is calculated as following with the parameters from algorithm 8:

\[
    \text{SINR} = \frac{P_{Rx,UE}}{P_{I,UE}}
\]

(7.8)
Algorithm 8 Calculate channel parameters

for all Enodes do
    Generate channel coefficients $h$ in time domain between Enode and all UEs within max range.
    Apply pathloss to channel coefficients.
    Transform $h(t)$ into $H(f)$ using $\forall RB$ calculate $H_{RB} = \sum_{n=0}^{\nu_{taps}} h_ne^{j2\pi RB180000}$
    for all RBs do
        for all UEs used in channel calculation do
            if UE owned by Enode then
                $H_{total,UE} = H_{RB} \sqrt{10^{\frac{\text{Allocated power(dBm)}}{10}}}$
                $P_{Rx,UE} = \left| H_{RB} \sqrt{10^{\frac{\text{Allocated power(dBm)}}{10}}} \right|^2$
            else
                $I_{Rx,UE} = \sum_{i=0}^{\nu_{\text{interfering Enodes}}} H_{RB} \sqrt{10^{\frac{\text{Allocated power(dBm)}}{10}}}$
            end if
        end for
    end for
    for all UE do
        $I_{Rx,UE} = I_{Rx,UE} + CN(0, 10^{\frac{\text{Noise level}}{10}})$
        $P_{I,UE} = \frac{|I_{Rx,UE}|^2}{2}$
    end for
end for
The SINR values are then translated into CQI values by direct mapping. The mapping was initially similar to values found in [34], [29] and [28] but the required SINR values for certain CQI values had to be increased due to a too high error rate in simulations. This means that this mapping lack anchoring in reality but is required for the simulation to work properly.

<table>
<thead>
<tr>
<th>SINR level [dB]</th>
<th>CQI number</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt; −2</td>
<td>0</td>
</tr>
<tr>
<td>[−2, −1]</td>
<td>1</td>
</tr>
<tr>
<td>[−1, 0]</td>
<td>2</td>
</tr>
<tr>
<td>[0, 1]</td>
<td>3</td>
</tr>
<tr>
<td>[1, 2]</td>
<td>4</td>
</tr>
<tr>
<td>[2, 3]</td>
<td>5</td>
</tr>
<tr>
<td>[3, 4]</td>
<td>6</td>
</tr>
<tr>
<td>[4, 5]</td>
<td>7</td>
</tr>
<tr>
<td>[5, 6]</td>
<td>8</td>
</tr>
<tr>
<td>[6, 7]</td>
<td>9</td>
</tr>
<tr>
<td>[7, 8]</td>
<td>10</td>
</tr>
<tr>
<td>[8, 9]</td>
<td>11</td>
</tr>
<tr>
<td>[9, 10]</td>
<td>12</td>
</tr>
<tr>
<td>[10, 11]</td>
<td>13</td>
</tr>
<tr>
<td>[11, 12]</td>
<td>14</td>
</tr>
<tr>
<td>[12, 13]</td>
<td>15</td>
</tr>
<tr>
<td>[13, 14]</td>
<td>16</td>
</tr>
<tr>
<td>[14, 15]</td>
<td>17</td>
</tr>
<tr>
<td>[15, 16]</td>
<td>18</td>
</tr>
<tr>
<td>[16, 17]</td>
<td>19</td>
</tr>
<tr>
<td>[17, 25]</td>
<td>20</td>
</tr>
<tr>
<td>[25, 40]</td>
<td>21</td>
</tr>
<tr>
<td>[40, 45]</td>
<td>22</td>
</tr>
<tr>
<td>[45, 65]</td>
<td>23</td>
</tr>
<tr>
<td>[65, 73]</td>
<td>24</td>
</tr>
<tr>
<td>[73, 78]</td>
<td>25</td>
</tr>
<tr>
<td>[78, 88]</td>
<td>26</td>
</tr>
<tr>
<td>[88, 93]</td>
<td>27</td>
</tr>
<tr>
<td>&gt; 93</td>
<td>28</td>
</tr>
</tbody>
</table>

Table 7.6. Which SINR levels is mapped into what CQI numbers.

7.8 Traffic models

One part in reaching good and reliable simulation results is to make sure that the data traffic is similar to reality. This should be made through assuming a mix
of different kind of data streams similar to what a normal user can be expect to have. For example imagine that a user is surfing the web, downloading a file using a FTP program and also at the same time carrying out a video call on skype. These three applications will create three data steams which have quite different characteristics and requirements. For example; the FTP transfer will require high throughput, because the files are usually quite large, while it is not crucial how the packages arrive in terms of delay. On the other hand, the video call application will have strict requirements on delay to keep the response time low, which otherwise would ruin the call experience. The requirements on data transmissions in order to provide a sufficient level of service between the application and the user leads to that a number of guarantees has to be set up, so called Quality of Service (QoS) requirements. The different EPS-bearer types described in table 4.1 also has some characterizations which is important for the traffic model and these is showed in table 7.7. In the 3GPP standard [5] it is stated that in simulations the EPS-bearers

<table>
<thead>
<tr>
<th>Type of requirement:</th>
<th>GBR(kbps)</th>
<th>Packet size</th>
<th>Arrival type</th>
</tr>
</thead>
<tbody>
<tr>
<td>VoIP(^a)</td>
<td>11.2</td>
<td>224 bits</td>
<td>Continuously</td>
</tr>
<tr>
<td>Live video(^b)</td>
<td>80</td>
<td>500 bits</td>
<td>Continuously</td>
</tr>
<tr>
<td>Video streaming(^b)</td>
<td>80</td>
<td>500 bits</td>
<td>Continuously</td>
</tr>
<tr>
<td>HTTP</td>
<td>-</td>
<td>800 bits</td>
<td>Cluster(^c)</td>
</tr>
<tr>
<td>FTP</td>
<td>-</td>
<td>500 bits(^d)</td>
<td>Cluster(^e)</td>
</tr>
</tbody>
</table>

\(^a\) Based on the codec G.711 with 8 kbps sample rate.\(^{[13]}\)
\(^b\) 160 packets are sent per second and a packet size of 500 bits is assumed, influenced by \(^{[19]}\).
\(^c\) Based on HTTP traffic model in \(^{[19]}\). Object sizes chosen to approximately mean (=80000 bits) and divided into 10 packets. Number of objects is set to mean of distribution in \(^{[19]}\) (=6), which gives 60 packets per cluster. Cluster spacing in seconds is exponentially distributed with \(\lambda = 0.033\).
\(^d\) File sizes are truncated lognormally distributed with mean 16Mbit and deviation of 5.776 Mbit with a maximum size of 40Mbit.\(^{[19]}\) It is assumed that each file is divided into 3200 packets.
\(^e\) The file spacing is exponentially distributed with \(\lambda = 0.006\).\(^{[19]}\) A packet spacing of 10ms is assumed.

Table 7.7. QoS requirements on different data types. Observe that a delay of 20 ms between the PCEF and the base station has already been deducted.\(^{[5]}\)

which are of Non-GBR type may experience congestion related packet drops and of those packets that are not dropped, it is expected that 98% of the packets do not exceed the delay budget. While those EPS-bearers that are of GBR type should not experience any congestion and also needs to deliver 98% of the packets within delay budget.\(^{[5]}\) A breach of these requirements are according to \(^{[5]}\) assumed to make a user unsatisfied, further, a packet error rate (after a certain time) above the requirement is assumed in this study to make an user unsatisfied. No packet drops due to full packet buffer is implemented in this simulation environment except for the packets that are dropped when a EPS-stream is dropped when a UE has more EPS-streams than can be supported.
7.8.1 EPS-bearer admission

The number of EPS-bearers allocated to each UE is adjusted continuously depending on how good the current EPS-bearers are doing. If the packet buffer of a UE is larger than 10,000 times the number of EPS-bearers bits, the packet buffer is on average increasing with more than 500 bits per ms and the number of EPS-bearers are more than 70% of average the UE is qualified to remove one EPS-bearer. If no UE served by an Enode is dropping EPS-bearers it will performed a check to see if any UE can add EPS-bearers. The UE with the lowest expected arrival rate of bits, which also has a packet buffer with less than 1000 bits and no more than 30% more EPS-bearers than average, is qualified to add an EPS-bearer.

7.9 Radio Resource Management

Which scheduler each Enode will use is a input when each Enode performs scheduling. What is common for all scheduler implementations is that they need to set a number of properties in the Enode’s EnodeScheduling and UEscheduling objects, further each UE’s UEscheduling objects need to be set. In general one can say that the EnodeScheduling class includes the scheduling information for the whole Enode while the UEscheduling holds the UE specific scheduling information. The reason behind that the Enodes also have a UEscheduling object is because it belongs to the base class Node. In tables 7.8 and 7.9 the most important parameters that needs to be set is listed.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>modulation</td>
<td>The used modulation (Not used for Enode).</td>
</tr>
<tr>
<td>CQI number</td>
<td>The used CQI number (Not used for Enode).</td>
</tr>
<tr>
<td>scheduledPacktes</td>
<td>References to all scheduled packets.</td>
</tr>
<tr>
<td>scheduledRBs</td>
<td>Which layers are used and how much power [mW] is allocated for each RB.</td>
</tr>
<tr>
<td>totalTXpower</td>
<td>The total amount of scheduled transmitting power [mW].</td>
</tr>
<tr>
<td>totalNrUsedRBs</td>
<td>Number of RB that is scheduled.</td>
</tr>
<tr>
<td>transportBlockSize</td>
<td>The size of the TB (Not used for Enode).</td>
</tr>
<tr>
<td>totalUsedCapacity</td>
<td>The total size of the scheduled packets (Not used for Enode).</td>
</tr>
<tr>
<td>HARQretransmission</td>
<td>True if it is a HARQ retransmission (Not used for Enode).</td>
</tr>
<tr>
<td>HARQTBID</td>
<td>The ID number of the TB in the HARQ buffer (Not used for Enode).</td>
</tr>
</tbody>
</table>

Table 7.8. The parameters that needs to be set in the UEscheduling class.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBAllocation</td>
<td>The scheduling information for each RB. Includes the following:</td>
</tr>
<tr>
<td></td>
<td>• Allocated UEs Which UEs are allocated to this RB.</td>
</tr>
<tr>
<td></td>
<td>• CQInumber What CQI number is used for respective UE.</td>
</tr>
<tr>
<td></td>
<td>• usedLayers What layers in the RB is used.</td>
</tr>
<tr>
<td></td>
<td>• alocatedPower How much power [mW] is allocated to the RB.</td>
</tr>
<tr>
<td>scheduledPackets</td>
<td>all packets scheduled at this Enode and this round.</td>
</tr>
</tbody>
</table>

Table 7.9. The parameters that needs to be set in the EnodeScheduling class.

7.9.1 Frequency usage

BS using the implemented PF scheduler are using frequency reuse by using two thirds of the RB to the closest UEs and the last third to the cell edge UEs, as described in 3.2. The RBs used for cell edge UEs are different for each sector. The FAPs, using the two implemented schedulers, situated at cell edge uses this frequency reuse by only schedule on RBs that are used by the BS for the close UEs. This mechanism decreases the interference from FAPs to UEs served by the BS but also decreases the resources available for the FAPs.

7.9.2 Power allocation

BS allocates twice as much power to UEs at cell edge compared to the ones close. If no transmissions are scheduled on a RB only $\frac{1}{21}$ power is allocated to represent the power always allocated to the RS. FAPs allocates power that is scaled depending on the redundancy version of the TB according to table 7.10. This mechanism further decreases interference to UEs served by other Enodes.

<table>
<thead>
<tr>
<th>Redundancy</th>
<th>Scaling</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-10dB</td>
</tr>
<tr>
<td>1</td>
<td>-5dB</td>
</tr>
<tr>
<td>2</td>
<td>-2dB</td>
</tr>
<tr>
<td>3</td>
<td>0dB</td>
</tr>
</tbody>
</table>

Table 7.10. The scaling of the transmitting power for FAPs.

7.10 Data Transmission

In this part the data transmission is described which consists of that the data going through the PHY-layer on the transmitter side, then is applied to the channel and
finally being decoded in the PHY-layer on the receiver side. An overview of the process is presented in Figure 7.4. Each part is further described in the subsections below.

**Figure 7.4.** The packet transmission process.

### 7.10.1 Data generation

The transport process starts with generation of a uniformly random bit vector of the same size as all scheduled packages for each UE.
7.10.2 CRC adding

A 24-bit long CRC is added to the TB. If the original data is larger than 6144 bits, the TB is divided into equally big parts that is smaller than 6144 bits, these parts is referred to as Code Blocks. A additional 24-bit long CRC is added to each of these Code Blocks. This procedure is illustrated in Figure 7.4.

7.10.3 Turbo code encoding

One CB at the time is fed to the Turbo Code Encoder. The encoder provided by ITPPs Turbo Code class is used with the transfer function, $G(D)$, given from the generation polynomials given in formulas below for the two 8-state constituent encoders [3].

$$G(D) = \left[1, \frac{g_1(D)}{g_0(D)}\right]$$

(7.9)

$$g_0(D) = 1 + D^2 + D^3$$

(7.10)

$$g_1(D) = 1 + D + D^3$$

(7.11)

The Turbo Code interleaver sequence used is generated using the method described in Section 5.1.3.2.3 and with values from Table 5.1.3-3 in [3]. The Turbo Code encoder gives two data streams and two parity streams out. The tail bits of the second data stream are added to the end of the data stream and to the two parity streams.

7.10.4 Rate matching

Each of the three streams are interleaved according to their specific method described in Section 5.1.4.1.1 in [3]. The three streams are then concatenated. Data is picked from this concatenated vector by the indexes generated by the function in Section 5.1.4.1.2 in [3] to form the final bits to be transmitted.

7.10.5 Modulation

The data is modulated using the scheduled modulation and placed in one of the resource elements within a RB that has been scheduled to the UE that the current TB belongs to.

7.10.6 Channel Usage

Since the channel estimations only have the granularity of RB, the same channel is applied to each resource element within each RB. We model the received interference, $I_{RE}$, as $CN(0, 1)$ distributed because the demodulator requires one interference variance for the whole spectrum. Since the interference does not have the variance one and therefore in order to get that we need to divide the received
symbols with the interference $I_{RE}$. The received OFDM symbol $Y$ can then be expressed as:

$$Y_{RE} = \frac{X_{TR} \times H_{RB} + I_{RE}}{I_{RE}} = X_{TR} \times \hat{H}_{RE} + 1$$  \hfill (7.12)

### 7.10.7 Demodulation

The transmitted symbols are mapped back into TBs and then demodulated using IT++ QAM demodulator[27]. The used channel and the interference variance is fed to the demodulator and soft bits are given from it. The higher positive value the more secure 0 while the bigger negative value the more sure 1.

### 7.10.8 Derate matching

The received soft bits of the code blocks are placed back at the index they were picked from at rate matching, the rest of the positions is filled out with insecure soft bits (0). The derate matched data is divided into three streams, each of them are deinterleaved. The three streams are then mapped into four streams to feed the turbo code decoder.

![Diagram](image)

**Figure 7.5.** The decoding process of the PHY layer, exclusive CRC checks.

### 7.10.9 Turbo Code Decoder

The decoder uses the same settings on the IT++ functions as the encoder but decodes instead.
7.10.10 CRC check

If any of the CBs fail the CRC check the TB transmission is considered to have failed. If all CBs pass the CRC check they are concatenated into the original TB and another CRC check is performed on the whole TB; if this CRC check is passed the TB with all the packets it includes is considered successfully delivered.

7.11 HARQ functionality

A received TB is put in the HARQ process buffer no matter if it was successfully transmitted or not. If the transmission failed some vital information to enable HARQ re-combining for the next transmission is stored otherwise only the most vital information is stored in the HARQ process buffer. The TBs in the queue are delayed to present the delay of the transmission and the return of the ACK/NACK from the UE to tell the base station if the transmission was successful. Each round this queue is checked to see if any ACK/NACK has been received. If a NACK has been received at the transmitter the packets of the TB are put into the HARQ retransmission packet buffer to be rescheduled in the next transmission. Further, at a NACK the TB information remains in the queue for the receiving of the next transmission of that TB. If a ACK is received the TB is removed from the HARQ process buffer. For both received ACKs and NACKs the statistics for packet and TB transmissions are updated. This process is illustrated in Figure 7.6.

7.12 Evaluation

7.12.1 Performance metrics

There is a number of performance metrics for LTE system simulations proposed in [6] however the following are available at different points in this simulation tool. Further, the following notations are used; R means throughput in bits per second, B means number of bits, P amount of packets, T means the time to average over, min of (simulated time,1000ms), t is time index and D means an amount of delay in milliseconds.

Whole Environment

The performance metrics in this section are calculated average over all BS respectively all FAPs.

Average data throughput

- \( R_{\text{avg},t} = \frac{R_{\text{BS},t-1}(T-1)+B_{\text{total in round}}\times 1000}{T} \)
- Available for both all BS and all FAP.

Average delay

- \( D_{\text{avg},t} = \frac{D_{\text{avg},t-1}\times P_{\text{total successful},t-1}+D_{\text{total in round}}}{P_{\text{total successful},t}} \)
- Available for both all BS and all FAP.
Figure 7.6. The movement of packets between different buffers throughout the simulation.
Average packet error rate (PER)

\[ PER = \frac{P_{\text{total failed}}}{P_{\text{total sent packets}}} \]

- Available for both all BS and all FAP.

**Each Macro cell**

The macro cell includes one BS and its UEs but also all FAPs and their UEs inside the coverage area of the BS. The same performance metrics as in 7.12.1 are available for each macro cell.

**Each BS**

In addition to the metrics described in 7.12.1 each BS has the following performance metrics available. Sector specific performance metrics in form of average throughput, expected throughput and average delay. The average throughput and average delay are calculated in the same manner as in 7.12.1. The expected throughput is an expected rate out from the current EPS-streams connected to the UEs of the BS.

Utilization of available resources

\[ \text{Utilization of available resources} = \frac{\sum_i \forall RB_{\text{assigned}} \text{ used CQI number}}{\sum_i \forall RBs \text{ best available CQI number}} \]

- Utilization of available resources shows how close to an optimal radio resource usage the base station was. Here the best available is the highest CQI value among all UEs on the RB.

Utilization of best resources

\[ \text{Utilization of best resources} = \frac{\sum_i \forall RB_{\text{assigned}} \text{ used CQI number}}{\text{total number of RBs} \times \text{highest CQI number}} \]

- Utilization of available resources shows how much the channels and sub-optimal channel usage affects the resource utilization for the base station.

User satisfaction

- Percentage of users at each base station that is not unsatisfied according to the criterions mentioned in section 7.8. Also a user is considered unsatisfied if the QoS requirements on the EPS-streams are not followed.

**Each FAP**

The performance metrics described in 7.12.1, the "Utilization of available resources" and "Utilization of best resources" are available for each FAP.

**Each UE**

In addition to the performance metric in 7.12.1 the following are available for each UE.
7.13 Simulation results

SINR
- The min and max SINR among the RBs and the average are shown.

CQI
- The average of the reported and the used values is shown.

Sent data
- The amount of data sent each turn.

Buffer size
- The total size of all packets waiting to be sent (not retransmissions).

Number of EPS-bearers
- The number of EPS-bearers the UE currently use.

Sent packets
- The total amount of successfully delivered, delay and failed packets. Note that a packet can be both successfully delivered and delayed.

Each EPS-bearer

Together with the same performance metrics as in 7.12.1 the QoS-requirements are shown as a comparison.

7.13 Simulation results

In this section it shown how the results from simulations are saved. All performance metrics described in 7.12.1 are saved to M-files which plots the data if run in Matlab. These files are stored in the running directory in a map named by date and time, the content of that folder is then organized as in table 7.11. Many of the performance metrics are also printed in the console at end of simulations.

7.13.1 Simulation speed

In table 7.12 the time it takes to simulate different scenarios can be seen and note that one third of the UEs in the macro cell will be allocated to FAPs. When keeping the number of UEs per cell constant and changing the number of FAPs do not change the simulation times noticeable. On the other hand increasing the number of BS is increasing the simulation times almost linearly. This is also the case for the used RBs, twice the number of RBs takes about the double time to simulate. Further, the number of rounds (ms) simulated also seems to scale the simulation time linearly, which one could expect.
Table 7.11. File organization of simulation results.
### Table 7.12. The simulation time at different settings.

<table>
<thead>
<tr>
<th>#BS</th>
<th>#FAP/BS</th>
<th>#UE/Cell</th>
<th>#RB</th>
<th>Rounds [ms]</th>
<th>Time [hh:mm:ss]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>6</td>
<td>25</td>
<td>50</td>
<td>00:15:43</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>6</td>
<td>25</td>
<td>50</td>
<td>00:14:48</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>6</td>
<td>25</td>
<td>50</td>
<td>00:38:02</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>30</td>
<td>25</td>
<td>50</td>
<td>00:39:56</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>30</td>
<td>25</td>
<td>100</td>
<td>00:58:28</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>30</td>
<td>50</td>
<td>50</td>
<td>01:11:15</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>30</td>
<td>25</td>
<td>50</td>
<td>01:37:18</td>
</tr>
</tbody>
</table>
Chapter 8

Conclusions and future work

8.1 FAP scheduling

When we studied the FAP scheduling problem we mainly looked at the scheduling algorithm which decides which UE should be scheduled on each RB, as in previous works. After running simulations we have noticed that EPS-bearer admission and the traffic models have a major impact on the performance. Thus, we conclude that it is important that the UE admission, the EPS-bearer admission and the scheduling are handled as one big problem and therefore also solved as one. In order to solve this problem in a satisfying way it is believed that statistics from real data traffic and the opinions from the mobile operators on how UEs should be served is needed. Statistics from real data traffic could enable modeling of the impact from the PHY-layer and HARQ functionality without implementing them completely, which should vastly decrease the simulation time. Opinions from the mobile operators could be a sound foundation when making assumptions regarding how many EPS-bearers should be allocated to each UE and when should a UE be dropped.

Even though the proposed QoS scheduler did not provide a better user satisfaction as intended we still believe that with a better UE- and EPS-bearer admission the idea of differentiating different QoS requirements will be able to give a higher satisfaction in the LTE networks. Further, we conclude that FAPs at cell edge using the frequency reuse pattern of the BS do not interfere UEs served by BS. On the contrary, FAPs close to the BS is expected to substantially interfere with UEs served by BS and this implies that a noble power allocation algorithm is needed for FAPs operating close to BS.

8.2 The simulator

In this section we present the conclusions regarding the implemented simulator. We created a simulator for scheduling of BS and FAP in an LTE environment. Even if only two types of schedulers were implemented, the environment enables
implementations of further schedulers in order to evaluate their performance in this kind of scenario. The simulator is considered to contain the major parts of LTE influencing the scheduling problem, with exception for MIMO, which should enable trustworthy studies of scheduling in a FAP/BS coexisting environment.

8.2.1 Coding

As seen in the simulation results the simulation times are increasing fast with the number of BS,FAP and UEs. This makes the size of the simulated environment, used BW and number of rounds the three ways to control the time it takes to run the simulation. When developing something new it is always difficult to imagine exactly what parts are needed. Which was the case here, therefore many classes became very nested into others which makes it more difficult for new people to understand the program. Therefore it is important that if another simulator is to be implemented it is recommended to study the parts that are included in this one but to make sure that each class become more isolated to keep the program modular and easy to understand.

In order to achieve satisfying simulation times it would be beneficial to be familiar with how to parallelize code in order to run the program efficiently on multi-core computers. Because there is a high degree of possible parallelism in this problem, for example when calculating channels and transmission to each UE, it should be possible to write simulators that can utilize several processor cores. Furthermore it should of course also be important to optimize the code as far as possible which most likely would require extensive experience in this kind of coding problem.

8.2.2 Functionality

When solving this kind of problem there is a trade off between simulation times and functionality. This simulator it is believed to include the functionality which have the greatest impact on the scheduling problem but still the simulation times are slightly too long. Therefore before adding more functionality as mobility, MIMO and uplink, it is important to make a more efficient code base and to run on multiple cores, as described. The functionality that is believed to influence the results the most is the MIMO since you then receive another degree to schedule, which MIMO mode to use , which has impact on capacity and error rate. MIMO transmission will also substantially increase the number of required PHY-layer operations, which means that it becomes a challenge to add MIMO without increasing simulation times.

8.3 Future work

There are several interesting ways of expanding this work and the most promising are listed below. This work can be seen as a tool to understand the problems when implementing a simulator. When solving further problems in LTE it is suggested
to implement a new simulator while remembering lessons learned learned in this one.

Parameter evaluation

- The proposed scheduler includes several weighting factors and may also need more, however it is believed that if these are set properly the performance of the proposed metric will increase. Therefore it could be interesting to evaluate the performance of the proposed scheduler at different parameter settings.

Efficient coding

- One way is to focus on speeding up the simulations by a more efficient coding and to utilize several processor cores, as mentioned above. This would enable studies on the impact of FAPs in a larger environment with several macro cells.

Expand problem

- Another way to expand this study could be to harder integrate UE- and EPS-bearer-admission into the scheduler, as mentioned above. Further, one could utilize MIMO transmissions and focus on how to chose MIMO transmission modes. It would also be very interesting to develop a more sophisticated power allocation algorithm, which was considered important for FAPs in [22], together with the other parts of the scheduler.

LTE-A

- The third interesting way to expand this work is to make a similar simulator but for LTE-A. The largest differences are that LTE-A includes other types of nodes, like for example relays, and that the maximum utilized bandwidth will be five times as large. This large bandwidth together with new nodes will make the necessity for optimized coding and simplifications even greater.
Bibliography


Appendix A

Implementation class list

Here the main classes described with their properties and methods.
A.1 SimEnvironment

A.1.1 Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bandwidth</td>
<td>double</td>
<td>The bandwidth in GHz used in simulation.</td>
</tr>
<tr>
<td>nrOfRB</td>
<td>int</td>
<td>The number of RB that belongs to the chosen bandwidth.</td>
</tr>
<tr>
<td>ownedBS</td>
<td>vector</td>
<td>Vector with references to all BS in simulation.</td>
</tr>
<tr>
<td>ownedFAP</td>
<td>vector</td>
<td>Vector with references to all FAPs in simulation.</td>
</tr>
<tr>
<td>centerFrequency</td>
<td>double</td>
<td>Carrier frequency in the simulation.</td>
</tr>
<tr>
<td>wavelength</td>
<td>double</td>
<td>The wavelength of the carrier in the simulation.</td>
</tr>
<tr>
<td>bgrNoise</td>
<td>double</td>
<td>The level of the background noise in dBm for the simulation.</td>
</tr>
<tr>
<td>maxInterferenceDistance</td>
<td>int[3]</td>
<td>The maximum distances that each type of nodes are interfering.</td>
</tr>
<tr>
<td>nrOfBSUE</td>
<td>int</td>
<td>Number of UEs connected to all BS.</td>
</tr>
<tr>
<td>nrOfSatisfiedBSUE</td>
<td>int</td>
<td>Number of satisfied UEs that are connected to BS.</td>
</tr>
<tr>
<td>BSaverageRate</td>
<td>double</td>
<td>Average data throughput for among BS.</td>
</tr>
<tr>
<td>BSavgCellRate</td>
<td>double</td>
<td>Average data throughput for each macro cell.</td>
</tr>
<tr>
<td>BSaverageDelay</td>
<td>double</td>
<td>Average delay among all BS.</td>
</tr>
<tr>
<td>BSnrOfSuccessfulPackets</td>
<td>int</td>
<td>Total number of successfully delivered packets for all BS.</td>
</tr>
<tr>
<td>BSnrOfDelayedPackets</td>
<td>int</td>
<td>Total number of successfully delivered packets but delayed for all BS.</td>
</tr>
<tr>
<td>BSnrOfFailedPackets</td>
<td>int</td>
<td>Total number of failed packets for all BS.</td>
</tr>
<tr>
<td>BStotalPackets</td>
<td>int</td>
<td>Total number of sent packets (successfully delivered + failed).</td>
</tr>
<tr>
<td>BSpacketErrorRate</td>
<td>double</td>
<td>Average PER among all BS.</td>
</tr>
<tr>
<td>BStotalDeliveredBits</td>
<td>int</td>
<td>Total amount of delivered bits for all BS.</td>
</tr>
<tr>
<td>BStotRoundDeliveredBits</td>
<td>int</td>
<td>Total amount of bits delivered this turn for all BS.</td>
</tr>
<tr>
<td>BStotRoundDelay</td>
<td>int</td>
<td>Sum of delays from all packets delivered this turn for all BS.</td>
</tr>
<tr>
<td>BStotRoundPackets</td>
<td>int</td>
<td>Total amount of packets delivered this turn for all BS.</td>
</tr>
<tr>
<td>BSavgAvailableResources</td>
<td>double</td>
<td>Average of the sums of the used CQI numbers from UEs among all BS.</td>
</tr>
<tr>
<td>BSavgBestResources</td>
<td>double</td>
<td>Average of the sums of the best available CQI numbers from UEs among all BS.</td>
</tr>
</tbody>
</table>

Table A.1. Properties of the SimEnvironment class
<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>nrOfFAPUE</td>
<td>int</td>
<td>Number of UEs connected to all FAP.</td>
</tr>
<tr>
<td>nrOfSatisfiedFAPUE</td>
<td>int</td>
<td>Number of satisfied UEs that are connected to FAP.</td>
</tr>
<tr>
<td>FAPaverageRate</td>
<td>double</td>
<td>Average data throughput for among FAP.</td>
</tr>
<tr>
<td>FAPavgCellRate</td>
<td>double</td>
<td>Average data throughput for each macro cell.</td>
</tr>
<tr>
<td>FAPaverageDelay</td>
<td>double</td>
<td>Average delay among all FAP.</td>
</tr>
<tr>
<td>FAPnrOfSuccessfulPackets</td>
<td>int</td>
<td>Total number of successfully delivered packets for all FAP.</td>
</tr>
<tr>
<td>FAPnrOfDelayedPackets</td>
<td>int</td>
<td>Total number of successfully delivered packets but delayed for all FAP.</td>
</tr>
<tr>
<td>FAPnrOfFailedPackets</td>
<td>int</td>
<td>Total number of failed packets for all FAP.</td>
</tr>
<tr>
<td>FAPtotalPackets</td>
<td>int</td>
<td>Total number of sent packets (successfully delivered + failed).</td>
</tr>
<tr>
<td>FAPpacketErrorRate</td>
<td>double</td>
<td>Average PER among all FAP.</td>
</tr>
<tr>
<td>FAPtotalDeliveredBits</td>
<td>int</td>
<td>Total amount of delivered bits for all FAP.</td>
</tr>
<tr>
<td>FAProundDeliveredBits</td>
<td>int</td>
<td>Total amount of bits delivered this turn for all FAP.</td>
</tr>
<tr>
<td>FAPtotRoundDelay</td>
<td>int</td>
<td>Sum of delays from all packets delivered this turn for all FAP.</td>
</tr>
<tr>
<td>FAPtotRoundPackets</td>
<td>int</td>
<td>Total amount of packets delivered this turn for all FAP.</td>
</tr>
<tr>
<td>FAPavgAvailableResources</td>
<td>double</td>
<td>Average of the sums of the used CQI numbers from UEs among all FAP.</td>
</tr>
<tr>
<td>FAPavgBestResources</td>
<td>double</td>
<td>Average of the sums of the best available CQI numbers from UEs among all FAP.</td>
</tr>
</tbody>
</table>

Table A.1. Properties of the SimEnvironment class (continued)

A.1.2 Methods

void StartSimulate(void)

Input: none

Output: none

Description: Make preparations before simulation such as generating CQI reports etc.

void Simulate(int time, int FAPscheduler)

Input: Number of milliseconds/turns to simulate, Which FAP scheduler should be used.
Output: none

**Description:** Main loop which runs the simulation.

double FAPprobability(int nrOfUsers)

**Input:** Number of UEs per macro cell.

**Output:** The probability for a UE to belong to a FAP.

**Description:** Help function when dividing the UEs between the BS and the FAPs.

int BWtoNrOfRB(double bw)

**Input:** Bandwidth

**Output:** Number of RBs

**Description:** Translates bandwidth into number of RBs.

static Position GeneratePos(Position BSpos, double BWradius, int minDist)

**Input:** Position of owning node, radius of owning node, minimum distance allowed from owning node.

**Output:** A new position.

**Description:** Is used when deploying FAPs and UEs inside the coverage area of the BS respectively the FAPs.

void CalcEnodesChannels(double centerFrequency, double noise)

**Input:** Carrier frequency, Level of background noise

**Output:** none

**Description:** Sets the channels for each UE by running the used channel model on each link.

Vec<Node*>(VecToVecRef(vector<UE> * original))

**Input:** Reference to a std vector containing UEs.

**Output:** ITPP vector with references to the same UEs casted to Nodes.

**Description:** Help function.
void UpdateStatistics(void)

**Input:** none

**Output:** none

**Description:** Update the statistics for the SimEnvironment level and calls functions to update statistics on all levels.

void SetUpChannels(void)

**Input:** none

**Output:** none

**Description:** Defines which possible links interfere each UE.

### A.2 Node

#### A.2.1 Structs

<table>
<thead>
<tr>
<th>Name</th>
<th>Members</th>
<th>type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Position</td>
<td>x</td>
<td>double</td>
<td>x-coordinate</td>
</tr>
<tr>
<td></td>
<td>y</td>
<td>double</td>
<td>y-coordinate</td>
</tr>
<tr>
<td></td>
<td>z</td>
<td>double</td>
<td>z-coordinate</td>
</tr>
</tbody>
</table>

*Table A.2.* Structs defined in the Node class file

#### A.2.2 Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pos</td>
<td>Position</td>
<td>X, Y and Z coordinates for the Node.</td>
</tr>
<tr>
<td>indoor</td>
<td>bool</td>
<td>Whether the Node is indoor or not.</td>
</tr>
<tr>
<td>radius</td>
<td>double</td>
<td>Coverage area of the Node.</td>
</tr>
<tr>
<td>maxPower</td>
<td>double</td>
<td>Maximum allowed transmitting power.</td>
</tr>
<tr>
<td>NodeID</td>
<td>int</td>
<td>ID number of the Node.</td>
</tr>
</tbody>
</table>

*Table A.3.* Properties of the Node class
A.2.3 Methods

virtual int typeof()

Input: none

Output: Which type of derived

Description: Virtual method

static double CalculateAngles(Position firstPos, Position secondPos)

Input: position of first node, position of second node

Output: angle in radians from first node to second node

Description: Calculate angles between nodes in x,y-plane.

static double CalculateThetaAngles(Position firstPos, Position secondPos)

Input: position of first node, position of second node

Output: angle in radians from first node to second node

Description: Calculate angles between nodes in z-plane.

static double CalculateDistanceToNode(Position firstPos, Position secondPos)

Input: position of first node, position of second node

Output: distance from first node to second node

Description: Calculates distance between nodes.
### A.3 Enode

#### A.3.1 Structs

<table>
<thead>
<tr>
<th>Name</th>
<th>Members</th>
<th>type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HARQprocess</td>
<td>TBid</td>
<td>int</td>
<td>ID of the TB.</td>
</tr>
<tr>
<td>owner</td>
<td>UE*</td>
<td></td>
<td>Reference to owner of TB.</td>
</tr>
<tr>
<td>packetsInProcess</td>
<td>vector</td>
<td></td>
<td>Vector with all packets in TB.</td>
</tr>
<tr>
<td>roundsUntilValid</td>
<td>int</td>
<td></td>
<td>Rounds until the result of delivery is available.</td>
</tr>
<tr>
<td>successfully-Delivered</td>
<td>bool</td>
<td></td>
<td>If successfully delivered.</td>
</tr>
<tr>
<td>waitingFor-HARQretransmission</td>
<td>bool</td>
<td></td>
<td>If the TB failed and is waiting to be retransmitted.</td>
</tr>
<tr>
<td>size</td>
<td>int</td>
<td></td>
<td>Number of bits in TB.</td>
</tr>
<tr>
<td>redundancy-Version</td>
<td>int</td>
<td></td>
<td>Redundancy version of the TB.</td>
</tr>
<tr>
<td>firstUsedCQI</td>
<td>int</td>
<td></td>
<td>The CQI used at the first transmission.</td>
</tr>
<tr>
<td>K</td>
<td>Vec</td>
<td></td>
<td>Constants used in the rate matching.</td>
</tr>
<tr>
<td>addedDatabits</td>
<td>int</td>
<td></td>
<td>Number of extra bits added to the TB.</td>
</tr>
<tr>
<td>nrOfRow</td>
<td>Vec</td>
<td></td>
<td>Number of rows in the interleaving after turbo encoder.</td>
</tr>
<tr>
<td>rawData</td>
<td>bvec</td>
<td></td>
<td>The original bit vector.</td>
</tr>
<tr>
<td>CRCData</td>
<td>bvec</td>
<td></td>
<td>The original bit vector with added CRC bits.</td>
</tr>
<tr>
<td>codeBlocks</td>
<td>Vec</td>
<td></td>
<td>CBs with CRC added.</td>
</tr>
<tr>
<td>D0sys</td>
<td>Vec</td>
<td></td>
<td>Systematic bits from the turbo encoder.</td>
</tr>
<tr>
<td>D1par1</td>
<td>Vec</td>
<td></td>
<td>First parity stream from the turbo encoder.</td>
</tr>
<tr>
<td>D2par2</td>
<td>Vec</td>
<td></td>
<td>Second parity stream from the turbo encoder.</td>
</tr>
<tr>
<td>TurboCodeddata</td>
<td>Vec</td>
<td></td>
<td>Turbo coded data and interleaved.</td>
</tr>
<tr>
<td>fillbits</td>
<td>Vec</td>
<td></td>
<td>Added bits to enable interleaving.</td>
</tr>
<tr>
<td>received-ReundancyD0</td>
<td>Vec</td>
<td></td>
<td>The received D0 vector, for HARQ combining.</td>
</tr>
<tr>
<td>received-ReundancyD1</td>
<td>Vec</td>
<td></td>
<td>The received D1 vector, for HARQ combining.</td>
</tr>
<tr>
<td>received-ReundancyD2</td>
<td>Vec</td>
<td></td>
<td>The received D2 vector, for HARQ combining.</td>
</tr>
</tbody>
</table>
## A.3.2 Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CQIdelay</td>
<td>int</td>
<td>Number of turns the CQI reports are delayed.</td>
</tr>
<tr>
<td>ownedUE</td>
<td>vector</td>
<td>Vector with all UEs served by Enode.</td>
</tr>
<tr>
<td>RBGsize</td>
<td>int</td>
<td>Size of the RBGs.</td>
</tr>
<tr>
<td>lastRBGsize</td>
<td>int</td>
<td>Size of the last RBG.</td>
</tr>
<tr>
<td>nrOfRB</td>
<td>int</td>
<td>Number of RB.</td>
</tr>
<tr>
<td>nrOfRBG</td>
<td>int</td>
<td>Number of RBG.</td>
</tr>
<tr>
<td>environment</td>
<td>envType enum</td>
<td>Which environment is used.</td>
</tr>
<tr>
<td>HARQdata</td>
<td>HARQ-buffer</td>
<td>All the HARQ processes.</td>
</tr>
<tr>
<td>nrOfUE</td>
<td>int</td>
<td>Number of UEs.</td>
</tr>
<tr>
<td>nrOfSatisfiedUE</td>
<td>int</td>
<td>Number of satisfied UEs.</td>
</tr>
<tr>
<td>satisfactionRate</td>
<td>double</td>
<td>Rate of UE satisfaction.</td>
</tr>
<tr>
<td>averageRate</td>
<td>double</td>
<td>Average data throughput (bps).</td>
</tr>
<tr>
<td>averageDelay</td>
<td>double</td>
<td>Average packet delay.</td>
</tr>
<tr>
<td>nrOfSuccessfulPackets</td>
<td>int</td>
<td>Number of successfully delivered packets.</td>
</tr>
<tr>
<td>nrOfDelayedPackets</td>
<td>int</td>
<td>Number of delayed packets.</td>
</tr>
<tr>
<td>nrOfFailedPackets</td>
<td>int</td>
<td>Number of failed packets.</td>
</tr>
<tr>
<td>totalPackets</td>
<td>int</td>
<td>Total number of sent packets.</td>
</tr>
<tr>
<td>packetErrorRate</td>
<td>double</td>
<td>Packet error rate.</td>
</tr>
<tr>
<td>totalDeliveredBits</td>
<td>int</td>
<td>Total delivered bits.</td>
</tr>
<tr>
<td>roundDeliveredBits</td>
<td>int</td>
<td>Total delivered bits last turn.</td>
</tr>
<tr>
<td>totRoundDelay</td>
<td>int</td>
<td>Sum of the delays from all packets last turn.</td>
</tr>
<tr>
<td>totRoundPackets</td>
<td>int</td>
<td>Total delivered packets last turn.</td>
</tr>
<tr>
<td>avgAvailableResources</td>
<td>double</td>
<td>Average sum of used resources.</td>
</tr>
<tr>
<td>avgBestResources</td>
<td>double</td>
<td>Average sum of best resources.</td>
</tr>
<tr>
<td>numberOfDroppedUsers</td>
<td>int</td>
<td>Number of dropped UEs.</td>
</tr>
</tbody>
</table>

*Table A.5. Properties of the Enode class*
A.3.3 Methods

void CheckHARQbuffer(void)

Input: none

Output: none

Description: Checks status of the HARQ buffer, commits successful/failed TBs or put in queue to retransmit.

void UpdateStatistics(void)

Input: none

Output: none

Description: Updates statistics on each owned UE and then updates statistics for the Enode.

bool CheckIfOwner(int UEID)

Input: ID of UE

Output: true if UE is owned by Enode.

Description: Test if the UE ID is found among its UEs.

virtual void ManageStreams(void)

Input: none

Output: none

Description: Checks if EPS-bearers should be deleted or added.

void ManageUsers(void)

Input: none

Output: none

Description: Delete UEs if too many transmissions fail.

A.4 UE

Derived from the Node class.
### A.4.1 Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>satisfied</td>
<td>bool</td>
<td>If the UE is satisfied.</td>
</tr>
<tr>
<td>waitingDLpackets</td>
<td>vector</td>
<td>New packets waiting to be scheduled.</td>
</tr>
<tr>
<td>waitingDLpacketsSize</td>
<td>int</td>
<td>Total size of all waiting packets.</td>
</tr>
<tr>
<td>waitingDLpacketsMaxSize</td>
<td>int</td>
<td>Maximal allowed size of all waiting packets.</td>
</tr>
<tr>
<td>sentDLpackets</td>
<td>vector</td>
<td>Vector with sent packets where the result of the transmission has not yet returned.</td>
</tr>
<tr>
<td>sentDLpacketsSize</td>
<td>int</td>
<td>Size of all packets in sent packets buffer.</td>
</tr>
<tr>
<td>HARQDLbuffer</td>
<td>vector</td>
<td>Packets waiting to be retransmitted.</td>
</tr>
<tr>
<td>HARQDLbufferSize</td>
<td>int</td>
<td>Size of all packets in the retransmission buffer.</td>
</tr>
<tr>
<td>EPSstreams</td>
<td>vector</td>
<td>Vector with all EPS-bearers connected to the UE.</td>
</tr>
<tr>
<td>deletedEPSstreams</td>
<td>vector</td>
<td>Vector with all deleted EPS-bearers, used for statistics.</td>
</tr>
<tr>
<td>Channel</td>
<td>Mat</td>
<td>Matrix with all channels and their taps.</td>
</tr>
<tr>
<td>totalInterferingSignal</td>
<td>Vec</td>
<td>Vector with the interfering signals on each RB.</td>
</tr>
<tr>
<td>totalInterferencePower</td>
<td>Vec</td>
<td>Vector with interfering power on each RB.</td>
</tr>
<tr>
<td>totSendingChannel</td>
<td>Vec</td>
<td>Vector with the transmitting channel on each RB.</td>
</tr>
<tr>
<td>totSendingPower</td>
<td>Vec</td>
<td>Vector with the transmitting power on each RB.</td>
</tr>
<tr>
<td>ownCQIreport</td>
<td>Vec</td>
<td>Vector with the DELAY latest CQI reports.</td>
</tr>
<tr>
<td>SINR</td>
<td>Vec</td>
<td>Vector with the SINR on each RB.</td>
</tr>
<tr>
<td>downData</td>
<td>UE-scheduling</td>
<td>The scheduling info for this UE.</td>
</tr>
<tr>
<td>expectedRate</td>
<td>double</td>
<td>Expected rate for the UE.</td>
</tr>
<tr>
<td>averageRate</td>
<td>double</td>
<td>Average data rate for the UE.</td>
</tr>
<tr>
<td>averageDelay</td>
<td>double</td>
<td>Average packet delay for the UE.</td>
</tr>
<tr>
<td>packetErrorRate</td>
<td>double</td>
<td>Average packer error rate for the UE.</td>
</tr>
<tr>
<td>avgBufferSizeDer</td>
<td>double</td>
<td>Average buffer size derivation.</td>
</tr>
<tr>
<td>avgTBretransmissions</td>
<td>double</td>
<td>Average number of TB retransmissions.</td>
</tr>
<tr>
<td>longTermAvgCQI</td>
<td>double</td>
<td>Average over time of average CQI.</td>
</tr>
<tr>
<td>nrOfUsedCQI</td>
<td>int</td>
<td>Number of times there was transmissions (for the average above).</td>
</tr>
<tr>
<td>nrOfSuccessfulPackets</td>
<td>int</td>
<td>Number of successfully delivered packets.</td>
</tr>
<tr>
<td>nrOfDelayedPackets</td>
<td>int</td>
<td>Number of successfully but delayed packets.</td>
</tr>
<tr>
<td>totalPackets</td>
<td></td>
<td>Total number of sent packets.</td>
</tr>
<tr>
<td>totalDeliveredBits</td>
<td></td>
<td>Total number of delivered bits.</td>
</tr>
<tr>
<td>roundDeliveredBits</td>
<td></td>
<td>Total number of bits delivered last turn.</td>
</tr>
<tr>
<td>totRoundDelay</td>
<td></td>
<td>Sum of delays from packets delivered last turn.</td>
</tr>
<tr>
<td>totRoundPackets</td>
<td></td>
<td>Total number of packets delivered last turn.</td>
</tr>
<tr>
<td>lastRoundBufferSize</td>
<td></td>
<td>The size last turn of the buffer with new packets.</td>
</tr>
<tr>
<td>numberOfDroppedStreams</td>
<td></td>
<td>Number of dropped EPS-bearers.</td>
</tr>
<tr>
<td>numberOfCommittedTB</td>
<td></td>
<td>Number of committed TB transmissions, both failed and successful.</td>
</tr>
<tr>
<td>numberOfFailedTB</td>
<td></td>
<td>Number of failed TB transmissions.</td>
</tr>
</tbody>
</table>
A.4.2 Methods

int typeof()
Input: none
Output: 2
Description: Enable to check type of Node class derived instances.

void CalculateCQI()
Input: none
Output: none
Description: Generates CQI reports and put them in the delay queue.

int GetNrOfSoftBits()
Input: none
Output: Amount of soft bits that can be saved in the HARQ processes.
Description: Depending on UE type there is a number mapped to it.

void UpdateCQIdelay()
Input: none
Output: none
Description: Shift the CQI reports one step in the delay queue.

void RegisterSucessfullPackets(vector<Packet*> * receivedPacket)
Input: Vector with successfully delivered packets.
Output: none
Description: Register the packets in the statistics and deletes them.

void RegisterFailedPackets(vector<Packet*> * lostPackets)
Input: Vector with failed packets.
Output: none
Description: Register the packets in the statistics and deletes them.
void GenerateNewPackets()
Input: none
Output: none
Description: Go through all EPS-bearers of the UE and generates new packets.

bool SendContinously(EPSstream * stream)
Input: Reference to EPS-bearer.
Output: True if a packet should be added.
Description: Decides when a new packet should be added for streams with continuous sending.

bool SendHTTP(EPSstream * stream)
Input: Reference to EPS-bearer.
Output: True if a packet should be added.
Description: Decides when a new packet should be added for HTTP streams.

bool SendFTP(EPSstream * stream)
Input: Reference to EPS-bearer.
Output: True if a packet should be added.
Description: Decides when a new packet should be added for FTP streams.

void UpdateStatistics()
Input: none
Output: none
Description: Updates the statistics on each EPS-bearers and then the UE’s statistics.

void UpdateExpectedRate()
Input: none
Output: none
Description: Updates the bit arrival rate that is expected from the currently owned EPS-bearers.
A.5  FAP

Derived from the Enode class.

A.5.1  Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EnodeSchedulingData</td>
<td>Enode-DownData</td>
<td>All the scheduling data for the FAP this turn.</td>
</tr>
<tr>
<td>ChannelMod</td>
<td>Channel-Model</td>
<td>The channel model instance used by the FAP.</td>
</tr>
<tr>
<td>expectedRate</td>
<td>double</td>
<td>Expected rate of arriving bits to be transmitted.</td>
</tr>
<tr>
<td>cellEdge</td>
<td>bool</td>
<td>True if the FAP is situated in the area furthest away from the BS.</td>
</tr>
</tbody>
</table>

Table A.7. Properties of the FAP class

A.5.2  Methods

double CalculatePower(int redundancy, bool RS)

Input:  Redundancy version of transmission, if only RS

Output:  Power to allocate

Description:  Calculates the power to allocate to a certain RB.

int typeof()

Input:  none

Output:  1

Description:  Enable to check type of Node class derived instances.

void UpdateStatistics(void)

Input:  none

Output:  none

Description:  Updates the statistics on each served UE and then the FAP’s statistics.
void TransmitPackets(void)

Input: none

Output: none

Description: Creates a instance of the Transportation class and then transmits the scheduled TBs.

void ManageStreams()

Input: none

Output: none

Description: Deletes or adds EPS-bearers to the UEs of the FAP.

void UpdateExpectedRate(void)

Input: none

Output: none

Description: Updates the bit arrival rate that is expected from the served UEs.

A.6 BS

Derived from the Enode class.

A.6.1 Structs

<table>
<thead>
<tr>
<th>Name</th>
<th>Members</th>
<th>type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sector</td>
<td>sectorStart</td>
<td>double</td>
<td>Angle where the sector starts (in radians).</td>
</tr>
<tr>
<td></td>
<td>sectorEnd</td>
<td>double</td>
<td>Angle where sector ends (in radians).</td>
</tr>
<tr>
<td></td>
<td>users</td>
<td>Vec</td>
<td>Vector with ref. to all UEs in sector served by BS.</td>
</tr>
<tr>
<td>ChannelMod</td>
<td></td>
<td></td>
<td>Instance of the channel model used.</td>
</tr>
<tr>
<td></td>
<td>expectedRate</td>
<td>double</td>
<td>Expected rate of arriving bits to be transmitted.</td>
</tr>
<tr>
<td></td>
<td>avgRate</td>
<td>double</td>
<td>Average rate in the sector.</td>
</tr>
<tr>
<td></td>
<td>avgDelay</td>
<td>double</td>
<td>Average packet delay in the sector.</td>
</tr>
</tbody>
</table>

Table A.8. Structs defined in the BS class file
A.6.2 Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sectors</td>
<td>Sector[3]</td>
<td>The three sectors of the BS.</td>
</tr>
<tr>
<td>EnodeDownData</td>
<td>Vec</td>
<td>Scheduling info for the three sectors of the BS.</td>
</tr>
<tr>
<td>avgMacroThroughput</td>
<td>double</td>
<td>Average total throughput in macro cell, both BS and FAPs.</td>
</tr>
<tr>
<td>avgMacroDelay</td>
<td>double</td>
<td>Average packet delay in macro cell, both BS and FAPs.</td>
</tr>
<tr>
<td>avgMacroSatisfiedUE</td>
<td>double</td>
<td>Average UE satisfaction in macro cell, both BS and FAPs.</td>
</tr>
<tr>
<td>avgMacroPER</td>
<td>double</td>
<td>Average PER in macro cell, both BS and FAPs.</td>
</tr>
</tbody>
</table>

Table A.9. Properties of the BS class

A.6.3 Methods

void SetSectors(void)

Input: none

Output: none

Description: Defines the start and end of each sector.

int typeof()

Input: none

Output: 0

Description: Enable to check type of Node class derived instances.

void SetUESector(void)

Input: none

Output: none

Description: Defines which sector each served UE belongs to.
double CalculatePower(bool highPow, int sector, bool RS)

**Input:** If the UE is in the area furthest away, which sector to transmit in, if RS

**Output:** Power to allocate

**Description:** Calculates the power to allocate to a certain RB.

```cpp
void UpdateStatistics(vector<FAP*>)
```

**Input:** none

**Output:** none

**Description:** Updates the statistics on each served UE and then the BS’s statistics.

```cpp
void TransmitPackets(void)
```

**Input:** none

**Output:** none

**Description:** Creates an instance of the Transportation class and then transmits the scheduled TBs.

```cpp
void ManageStreams(int sec)
```

**Input:** none

**Output:** none

**Description:** Deletes or adds EPS-bearers to the UEs of the BS.

```cpp
void UpdateExpectedRate(int sec)
```

**Input:** sector to update

**Output:** none

**Description:** Updates the bit arrival rate that is expected from the served UEs.
A.7 ChannelModel

A.7.1 Structs

<table>
<thead>
<tr>
<th>Name</th>
<th>Members</th>
<th>type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correlation-Distance</td>
<td>DS</td>
<td>double</td>
<td>Delay spread correlation distance.</td>
</tr>
<tr>
<td>ASD</td>
<td>double</td>
<td></td>
<td>Rms azimuth spread of departure angles correlation distance.</td>
</tr>
<tr>
<td>ASA</td>
<td>double</td>
<td></td>
<td>Rms azimuth spread of arrival angles correlation distance.</td>
</tr>
<tr>
<td>SF</td>
<td>double</td>
<td></td>
<td>Correlation distance of shadow fading.</td>
</tr>
<tr>
<td>K</td>
<td>double</td>
<td></td>
<td>Correlation distance of Ricean K-factor.</td>
</tr>
<tr>
<td>receivingLink</td>
<td>receiver</td>
<td>UE*</td>
<td>Receiver in link.</td>
</tr>
<tr>
<td>channelGen</td>
<td>TDL_Channel*</td>
<td></td>
<td>Channel generator.</td>
</tr>
<tr>
<td>channel</td>
<td>Vec</td>
<td></td>
<td>Channel coefficients for the different taps.</td>
</tr>
<tr>
<td>pathloss</td>
<td>double</td>
<td></td>
<td>The pathloss between the sender and receiver.</td>
</tr>
<tr>
<td>LOS</td>
<td>bool</td>
<td></td>
<td>If there is LOS between sender and receiver.</td>
</tr>
</tbody>
</table>

Table A.10. Structs defined in the ChannelModel class file
A.7.2 Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sender</td>
<td>Enode*</td>
<td>Sending Enode.</td>
</tr>
<tr>
<td>receivers</td>
<td>Vec</td>
<td>Vector with all receiving links.</td>
</tr>
<tr>
<td>randGen</td>
<td>Uniform_RNG</td>
<td>Random generator used in 3GPP channel calculations.</td>
</tr>
<tr>
<td>simulationEnvironment</td>
<td>enum</td>
<td>Type of environment used in channel calculations.</td>
</tr>
<tr>
<td>wavelength</td>
<td>double</td>
<td>The wavelength of the carrier frequency.</td>
</tr>
<tr>
<td>centerFreq</td>
<td>double</td>
<td>The carrier frequency.</td>
</tr>
<tr>
<td>maxInterferenceDistance</td>
<td>double[3]</td>
<td>The maximum distance that the three node types are considered to interfere.</td>
</tr>
<tr>
<td>smallScaleMean</td>
<td>Mat</td>
<td>Matrix with the mean of the small scale parameters for the 3GPP channel.</td>
</tr>
<tr>
<td>smallScaleVar</td>
<td>Mat</td>
<td>Matrix with the variance of the small scale parameters for the 3GPP channel.</td>
</tr>
<tr>
<td>distCorr</td>
<td>Mat</td>
<td>Matrix containing calculated values for the 3GPP channel.</td>
</tr>
<tr>
<td>ExpDistCorr</td>
<td>Mat</td>
<td>Matrix containing calculated values for the 3GPP channel.</td>
</tr>
<tr>
<td>RealSQRTCrossCorr</td>
<td>Mat</td>
<td>Matrix containing calculated values for the 3GPP channel.</td>
</tr>
<tr>
<td>DS</td>
<td>Vec</td>
<td>Final delay spread.</td>
</tr>
<tr>
<td>ASD</td>
<td>Vec</td>
<td>Final rms azimuth spread of departure angles.</td>
</tr>
<tr>
<td>ASA</td>
<td>Vec</td>
<td>Final rms azimuth spread of arrival angles.</td>
</tr>
<tr>
<td>SF</td>
<td>Vec</td>
<td>Final shadow fading.</td>
</tr>
<tr>
<td>K</td>
<td>Vec</td>
<td>Final Ricean K-factor.</td>
</tr>
<tr>
<td>GMatrix</td>
<td>Mat</td>
<td>Matrix containing calculations with the previous five.</td>
</tr>
</tbody>
</table>

Table A.11. Properties of the ChannelModel class

A.7.3 Methods

bool IfWithinSector(double start, double end, double dir)

Input: start of sector, end of sector, direction
Output: True if direction is within sector

Description: Determines if dir is within sector.

static void BSSetReceivingUEs(BS * sender, vector<BS*> * allBS, vector<FAP*> * allFAP)

Input: BS, vector with all BS, vector with all FAPs

Output: none

Description: Sets which UEs the BS is interfering.

static void FAPSetReceivingUEs(FAP * sender, vector<BS*> * allBS, vector<FAP*> * allFAP)

Input: FAP, vector with all BS, vector with all FAPs

Output: none

Description: Sets which UEs the FAP is interfering.

void CalculatePathlossMatrix(vector<BS> * allBS, vector<FAP> * allFAP, double centerFrequency, ENode::envType simulationEnvironment)

Input: 

Output: 

Description:

bool GenerateLOS(double distance)

Input: distance from transmitter to receiver

Output: true if LOS

Description: Depending on environment determines if LOS.

double lnHLOS(double d)

Input: distance from transmitter to receiver

Output: probability for LOS

Description: Determines the probability for LOS in InH environment.
double UMiLOS(double d)
Input: distance from transmitter to receiver
Output: probability for LOS
Description: Determines the probability for LOS in UMi environment.

double UMaLOS(double d)
Input: distance from transmitter to receiver
Output: probability for LOS
Description: Determines the probability for LOS in UMa environment.

double RMaLOS(double d)
Input: distance from transmitter to receiver
Output: probability for LOS
Description: Determines the probability for LOS in RMa environment.

double CalculatePathloss(double distance, double cf, bool LOS, double TXheight, double RXheight, double indoorWalls)
Input: distance from transmitter to receiver, carrier frequency, LOS, height of transmitter, height of receiver, number of walls
Output: pathloss in dB
Description: Calculates the pathloss for the right environment with walls.

double lnHpathloss(double d, double cf, bool LOS)
Input: distance from transmitter to receiver, carrier frequency, LOS
Output: pathloss in dB
Description: Calculates the pathloss for the InH environment.

double UMipathloss(double d, double cf, bool LOS, double BSheight, double UEheight)
Input: distance from transmitter to receiver, carrier frequency, LOS, height of transmitter, height of receiver
Output: pathloss in dB
Description: Calculates the pathloss for the UMi environment.
double UMapathloss(double d, double cf, bool LOS, double BSheight, double UEheight)

**Input:** distance from transmitter to receiver, carrier frequency, LOS, height of transmitter, height of receiver

**Output:** pathloss in dB

**Description:** Calculates the pathloss for the UMa environment.

double RMapathloss(double d, double cf, bool LOS, double BSheight, double UEheight)

**Input:** distance from transmitter to receiver, carrier frequency, LOS, height of transmitter, height of receiver

**Output:** pathloss in dB

**Description:** Calculates the pathloss for the RMa environment.

void SetLOSvec(void)

**Input:** none

**Output:** none

**Description:** Set the LOS between transmitter and all receivers.

void CalculateSimplifiedChannel()

**Input:** transmitter, receivers, wavelength, environment, vector with LOS/NLOS to each receiver

**Output:** Vector with complex matrices containing the channel coefficients to each receiver.

**Description:** Calculates the 3GPP channel to each receiver and deduct pathloss.

void CalculateVerySimpleChannel()

**Input:** none

**Output:** none

**Description:** Calculates the ITU Pedestrian A channel to each receiver and deduct pathloss.

void SetSmallScaleParameters()

**Input:** none

**Output:** none

**Description:** Sets the small scale parameters depending on the environment.
void CalcExpCorrDist()
Input: none
Output: none
Description: Calculates the correlation between receivers.

void CalcCrossCorrMatrix(void)
Input: none
Output: none
Description: Calculates the cross-correlation of the small scale parameters.

void CalcSmallScaleVar(void)
Input: none
Output: none
Description: Sets the small scale parameters variance and calculates the final small scale parameters.

void SetBasicParameters(double cf, double wl, int maxInt0, int maxInt1, int maxInt2)
Input: carrier frequency, wavelength, maximum interference distance for UEs, maximum interference distance for FAPs, maximum interference distance for BSs
Output: none
Description: Sets basic values to the channel calculations.

void CalcStartup()
Input: none
Output: none
Description: Run all calculations for the 3GPP channel that can be run before simulation starts.

double GenerateClusterScaling(int clusters, bool InH)
Input: cluster index, if InH environment
Output: scaling
Description: Maps the cluster index to a scaling.
Vec<double> GenerateRayAngleSpread(Enode::envTypesimE, bool LOS, bool departure)

Input: environment, LOS, if departing ray
Output: Vector with angle spread for rays
Description: Calculates the angle spread for rays within clusters.

Mat<double> GenerateFieldPattern(double phi, double zeta, bool bs)

Input: angle from transmitter to receiver on z-axis, angle from receiver to transmitter on z-axis, if BS
Output: matrix with field pattern
Description: Calculates the field pattern to be used in the 3GPP channel.

double numberOfWalls(Enode * sender, UE * receiver)

Input: reference to transmitter, reference to receiver
Output: number of walls
Description: Calculates how many walls between transmitter and receiver for pathloss purposes.

A.8 CQI

A.8.1 Properties

<table>
<thead>
<tr>
<th>singleCQI</th>
<th>Vec</th>
<th>Vector with CQI value for each RB.</th>
</tr>
</thead>
<tbody>
<tr>
<td>averageCQI</td>
<td>int</td>
<td>Average CQI over all RBs.</td>
</tr>
</tbody>
</table>

Table A.12. Properties of the CQI class

A.8.2 Methods

static int CQItoBlockSizeOneLayer(int CQInr)

Input: CQI number
Output: Capacity of one RB
Description: Maps CQI number to a capacity in bits for one RB.
static int CQItoTransportBlockSize(int CQI, int nrOfRB, bool oneLayer=true)

**Input:** CQI number, number of used RB, if one layer is used (always in SISO)

**Output:** int

**Description:** Translates CQI and number of RBs into capacity in bits.

static int CQItoModulation(int CQI)

**Input:** CQI number

**Output:** number of bits in modulation

**Description:** Translates CQI number into which modulation should be used and how many bits maps to each symbol.

void CalculateAverageCQI(void)

**Input:** none

**Output:** none

**Description:** Calculates the average CQI among RBs.

### A.9 UEscheduling

#### A.9.1 Structs

<table>
<thead>
<tr>
<th>Name</th>
<th>Members</th>
<th>type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBScheduling</td>
<td>usedLayers</td>
<td>bitset&lt;4&gt;</td>
<td>Indicates which layers are used in RB.</td>
</tr>
<tr>
<td></td>
<td>allocatedPower</td>
<td>double</td>
<td>Allocated power to RB in mW.</td>
</tr>
</tbody>
</table>

*Table A.13.* Structs defined in the UEscheduling class file
A.9.2 Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>modulation</td>
<td>int</td>
<td>What modulation is scheduled, in bits per symbol.</td>
</tr>
<tr>
<td>CQInumber</td>
<td>int</td>
<td>What CQI number is scheduled.</td>
</tr>
<tr>
<td>scheduledPackets</td>
<td>vector</td>
<td>Vector with scheduled packets.</td>
</tr>
<tr>
<td>scheduledRBs</td>
<td>Vec</td>
<td>Indicates which RBs has been scheduled to the owner.</td>
</tr>
<tr>
<td>totalTXPower</td>
<td>double</td>
<td>The total amount of power allocated on RBs.</td>
</tr>
<tr>
<td>totalNrUsedRBs</td>
<td>int</td>
<td>Number of RBs scheduled to owner.</td>
</tr>
<tr>
<td>transportBlockSize</td>
<td>int</td>
<td>Capacity in bits of scheduled RBs and CQI.</td>
</tr>
<tr>
<td>totalUsedCapacity</td>
<td>int</td>
<td>Total size of all scheduled packets.</td>
</tr>
<tr>
<td>resourceAllocationMode</td>
<td>int</td>
<td>What resource allocation mode is used.</td>
</tr>
<tr>
<td>nrOfLayerMapped</td>
<td>int</td>
<td>Number of layers that will be mapped to.</td>
</tr>
<tr>
<td>HARQretransmission</td>
<td>bool</td>
<td>If the scheduled packets are a retransmission.</td>
</tr>
<tr>
<td>HARQTBID</td>
<td>int</td>
<td>ID of the retransmitted TB.</td>
</tr>
</tbody>
</table>

Table A.14. Properties of the UEscheduling class

A.10 EnodeScheduling

A.10.1 Structs

<table>
<thead>
<tr>
<th>Name</th>
<th>Members</th>
<th>type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBEnodeScheduling</td>
<td>allocatedUEs</td>
<td>Vec</td>
<td>Vector with UEs allocated to the RB, several UEs only possible in MIMO.</td>
</tr>
<tr>
<td></td>
<td>CQInumber</td>
<td>Vec</td>
<td>CQI number used to each allocated UE.</td>
</tr>
<tr>
<td></td>
<td>usedLayers</td>
<td>bitset&lt;4&gt;</td>
<td>Indicates which layers have been scheduled.</td>
</tr>
<tr>
<td></td>
<td>allocatedPower</td>
<td>double</td>
<td>Allocated power to RB in mW.</td>
</tr>
</tbody>
</table>

Table A.15. Structs defined in the EnodeScheduling class file
A.10.2 Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBallocation</td>
<td>Vec</td>
<td>Vector with a RBEnodeScheduling struct for each RB.</td>
</tr>
<tr>
<td>scheduledPackets</td>
<td>vector</td>
<td>Scheduled packets to be sent from Enode.</td>
</tr>
<tr>
<td>highSpectrum</td>
<td>bvec</td>
<td>Indicates which RBs should be used for high power transmissions.</td>
</tr>
<tr>
<td>lowSpectrum</td>
<td>bvec</td>
<td>Indicates which RBs should be user for low power transmissions.</td>
</tr>
<tr>
<td>totalTXPower</td>
<td>double</td>
<td>Total scheduled power in mW.</td>
</tr>
<tr>
<td>totalUsedRBs</td>
<td>int</td>
<td>Total number of, by the Enode, scheduled RBs.</td>
</tr>
</tbody>
</table>

Table A.16. Properties of the EnodeScheduling class

A.11 Scheduler

A.11.1 Methods

static void Schedule(int scheduler, Enode * baseStation)

Input: which scheduler to run, which Enode should we schedule for

Output: none

Description: Runs the correct scheduler depending on the input.

static void QOSscheduler(FAP * baseStation)

Input: FAP to schedule

Output: none

Description: Schedule the FAP using the proposed QoS scheduler and sets the UEscheduling and EnodeScheduling instances.

static void ProportionalFairBSScheduler(BS * baseStation)

Input: BS to schedule

Output: none

Description: Schedule the BS using the reference PF scheduler and sets the UEscheduling and EnodeScheduling instances.
static void ProportionalFairFAPScheduler(FAP * baseStation)

**Input:** FAP to schedule

**Output:** none

**Description:** Schedule the FAP using the reference PF scheduler and sets the UE scheduling and EnodeScheduling instances.
### A.12 Transportation

#### A.12.1 Structs

<table>
<thead>
<tr>
<th>Name</th>
<th>Members</th>
<th>type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UEinfo</td>
<td>owner</td>
<td>UE*</td>
<td>Reference to TB owner.</td>
</tr>
<tr>
<td>ID</td>
<td></td>
<td>unsigned</td>
<td>ID of the TB.</td>
</tr>
<tr>
<td>HARQretransmission</td>
<td></td>
<td>bool</td>
<td>If TB is a retransmission.</td>
</tr>
<tr>
<td>redundancyVer</td>
<td></td>
<td>int</td>
<td>Redundancy version of the TB.</td>
</tr>
<tr>
<td>nrOfLayersMapped</td>
<td></td>
<td>int</td>
<td>Number of layers mapped.</td>
</tr>
<tr>
<td>usedRBs</td>
<td></td>
<td>bvec</td>
<td>Which RBs are used by the TB.</td>
</tr>
<tr>
<td>CQInr</td>
<td></td>
<td>int</td>
<td>CQI number used in TB transmission.</td>
</tr>
<tr>
<td>packetsInProcess</td>
<td></td>
<td>Vec</td>
<td>Vector with packets in TB.</td>
</tr>
<tr>
<td>rawData</td>
<td></td>
<td>bvec</td>
<td>Original transmitted data bits.</td>
</tr>
<tr>
<td>CRCData</td>
<td></td>
<td>bvec</td>
<td>Data bits after adding CRC.</td>
</tr>
<tr>
<td>codeBlocks</td>
<td></td>
<td>Vec</td>
<td>Code Blocks with added CRC on each.</td>
</tr>
<tr>
<td>D0sys</td>
<td></td>
<td>Vec</td>
<td>Vector with systematic bits out from turbo encoder.</td>
</tr>
<tr>
<td>D1par1</td>
<td></td>
<td>Vec</td>
<td>Vector with parity 1 bits out from turbo encoder.</td>
</tr>
<tr>
<td>D2par2</td>
<td></td>
<td>Vec</td>
<td>Vector with parity 2 bits out from turbo encoder.</td>
</tr>
<tr>
<td>TurboCodedData</td>
<td></td>
<td>Vec</td>
<td>Vector with concatenated and interleaved streams from the turbo encoder.</td>
</tr>
<tr>
<td>FillbitIndication</td>
<td></td>
<td>Vec</td>
<td>Vector with bit indication of where there were added bits to enable interleaving of the turbo encoded data.</td>
</tr>
<tr>
<td>rateMatchedData</td>
<td></td>
<td>Vec</td>
<td>Vector with data bit stream after rate matching.</td>
</tr>
<tr>
<td>RateMatchedData-AndConcat</td>
<td></td>
<td>bvec</td>
<td>Concatenated rate matched bit streams.</td>
</tr>
<tr>
<td>modulatedData</td>
<td></td>
<td>cvec</td>
<td>Vector with complex symbols to be transmitted.</td>
</tr>
<tr>
<td>HMatrix</td>
<td></td>
<td>cmat</td>
<td>Matrix with the channel coefficients for each time and RE.</td>
</tr>
<tr>
<td>HVector</td>
<td></td>
<td>cvec</td>
<td>Vector with channel coefficients for each RE.</td>
</tr>
<tr>
<td>recSymbols</td>
<td></td>
<td>cvec</td>
<td>Vector with the received symbols, with channel and interference added.</td>
</tr>
<tr>
<td>demodulatedData</td>
<td></td>
<td>vec</td>
<td>Vector with softbits after demodulation.</td>
</tr>
</tbody>
</table>
A.12.2 Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>UEallocation</td>
<td>Vec</td>
<td>Vector with all transmissions (UEinfo) from Enode.</td>
</tr>
</tbody>
</table>

Table A.18. Properties of the Transportation class

A.12.3 Methods

void InitializeTransport(Enode * sender, Vec<RBEnodeScheduling> * scheduledRBs, HARQbuffer * HARQs)

**Input:** transmitter, reference to scheduling information from Enode, reference to HARQ processes

**Output:** none

**Description:** Set-up the basic parameters for each transmission.

void TransportBS(BS * owner, HARQbuffer * HARQs)

**Input:** reference to transmitting BS, reference to HARQ processes

**Output:** none

**Description:** Handling transmissions for BS.

void TransportFAP(FAP * owner, HARQbuffer * HARQs)

**Input:** reference to transmitting FAP, reference to HARQ processes

**Output:** none

**Description:** Handling transmissions for FAP.

void GenerateData(UEinfo * UEsending, int size)

**Input:** reference to current UEinfo, size of data in TB

**Output:** none

**Description:** Generates the bits to be transmitted.

void AddCRCtoTB(UEinfo * UEsending, bvec * data)

**Input:** reference to current UEinfo, data bits to add CRC to

**Output:** none

**Description:** Add CRC to original data bits.
void DivideIntoCBAddCRC(UEinfo * UEsending, bvec * data)

**Input:** reference to current UEinfo, data bits to divide

**Output:** none

**Description:** Divides the data bits into code blocks and add CRC to each code block.

Vec<int> ReturnK(int nrOfCB, int dataSize, int maxSize, int CRCsize)

**Input:** number of code blocks, number of data bits, maximum data bits, size of CRC addition

**Output:** vector with the K values

**Description:** Calculates the K values for each code block needed in the rate matching.

void TurboEncode(UEinfo * UEsending, Vec<bvec> * blocks)

**Input:** reference to current UEinfo, code blocks to encode

**Output:** none

**Description:** Turbo encodes code blocks.

void SubBlockInterleaving(UEinfo* UEsending, Vec<bvec> * D0blocks, Vec<bvec> * D1blocks, Vec<bvec> * D2blocks)

**Input:** reference to current UEinfo, vector with D0 streams from turbo encoder, vector with D1 streams from turbo encoder, vector with D2 streams from turbo encoder

**Output:** none

**Description:** Interleaves the streams out from the turbo encoder for each code block.

ivec GenerateTurboInterleaverSequence(int blockSize)

**Input:** size of code block

**Output:** interleaver sequence

**Description:** Determines the interleaver sequence out from the size of the code blocks.
int ReturnSubblockInterleavingIndex(int index)

Input: index in target

Output: index in source

Description: Gives the source index when interleaving

void RateMatch(UEinfo * UEsending, Vec<bvec> * turboData)

Input: reference to current UEinfo, reference to turbo encoded data

Output: none

Description: Performs rate matching.

void Modulate(UEinfo * UEsending, bvec * data)

Input: reference to current UEinfo, data to modulate

Output: none

Description: Modulates data.

cmat TBsToSendBlock()

Input: none

Output: matrix with symbols in RE in time slots and subcarriers.

Description:

cmat ApplyChannel(cmat transmission, Enode * basestation)

Input: matrix with transmitted symbols, reference to transmitter

Output: matrix with received symbols in RE, in time and subcarrier

Description: Applies the channel and interference to the transmitted symbols.

void SendBlockToTBs(cmat freqDom)

Input: matrix with received symbols

Output: none

Description: Reorganizes received matrix into TB.
void Demodulate(Vec<complex<double> * input, UEinfo * UEsending, Enode * owner)

**Input:** vector with symbols to demodulate, reference to current UEinfo, reference to transmitter

**Output:** none

**Description:** Demodulates complex symbols into soft bits.

void InvRateMatch(UEinfo * UEsending, vec * demodData)

**Input:** reference to current UEinfo, demodulated data vector

**Output:** none

**Description:** Inverts the rate matching, fill the unknown positions with soft bits with value 0.

void TurboDecoding(UEinfo * UEsending, Vec<vec> * D0, Vec<vec> * D1, Vec<vec> * D2)

**Input:** reference to current UEinfo, Vector with the D0 vectors, Vector with the D1 vectors, Vector with the D2 vectors

**Output:** none

**Description:** Performs turbo decoding of the received soft bits.

bool CodeBlockCRCcheckAndConcat(UEinfo*UEsending, HARQbuffer * HARQs, Vec<bvec> * decBits)

**Input:** reference to current UEinfo, reference to HARQ processes, vector with decoded data bit streams

**Output:** true if the CRC check passed

**Description:** Checks the CRC code on the decoded code blocks and then concatenates them.

bool TransportBlockCRCCheck(UEinfo * UEsending, HARQbuffer * HARQs, bvec * data)

**Input:** reference to current UEinfo, reference to HARQ processes, bit vector with TB with CRC

**Output:** true if TB passed CRC check

**Description:** Check the CRC code on the TB.
void FailedTB(UEinfo * UEsending, HARQbuffer * HARQs)
Input: reference to current UEinfo, reference to HARQ processes
Output: none
Description: Saves the necessary info about the transmission in HARQ processes into a UEinfo struct.

void SuccessfullTB(UEinfo * UEsending, HARQbuffer * HARQs)
Input: reference to current UEinfo, reference to HARQ processes
Output: none
Description: Saves the necessary info about the transmission in HARQ processes into a UEinfo struct.

void TestSoftBits(bvec original, vec softbits)
Input: bit vector, soft bit vector with doubles
Output: none
Description: Prints the number of errors among soft bits (according to IT++’s definition of soft bits).

void TestBits(bvec original, bvec currBits)
Input: original bit vector, bit vector to compare
Output: none
Description: Prints the number of differences in the two vectors.

void PrintModulated(cvec modulated)
Input: vector with complex values
Output: none
Description: Prints a complex vector in a nice way.

void PrintSoftBits(vec softbits)
Input: vector with doubles
Output: none
Description: Prints a double vector in a nice way.
void PrintBits(bvec bits, string msg)

**Input:** bit vector

**Output:** none

**Description:** Prints a bit vector in a nice way.
### A.13 EPSstream

#### A.13.1 Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>enum</td>
<td>Type of EPS-bearer.</td>
</tr>
<tr>
<td>packetSize</td>
<td>int</td>
<td>Size of packets.</td>
</tr>
<tr>
<td>maxDelay</td>
<td>int</td>
<td>Requirement on maximum allowed packet delay.</td>
</tr>
<tr>
<td>maxPacketErrorRate</td>
<td>double</td>
<td>Requirement on maximum allowed PER.</td>
</tr>
<tr>
<td>minRate</td>
<td>double</td>
<td>Requirement on minimum data rate (bps).</td>
</tr>
<tr>
<td>roundCreated</td>
<td>int</td>
<td>The round the EPS-bearer was created.</td>
</tr>
<tr>
<td>meanRate</td>
<td>double</td>
<td>Expected average data rate (bps).</td>
</tr>
<tr>
<td>clusterDistance</td>
<td>double</td>
<td>If cluster, expected time between clusters in seconds.</td>
</tr>
<tr>
<td>packetSpace</td>
<td>double</td>
<td>Time between packets in ms.</td>
</tr>
<tr>
<td>clusterNr</td>
<td>int</td>
<td>If cluster, packet number in cluster.</td>
</tr>
<tr>
<td>satisfying</td>
<td>bool</td>
<td>true if the requirements are fulfilled.</td>
</tr>
<tr>
<td>averageRate</td>
<td>double</td>
<td>Average data rate for the EPS-bearer.</td>
</tr>
<tr>
<td>averageDelay</td>
<td>double</td>
<td>Average packet delay for the EPS-bearer.</td>
</tr>
<tr>
<td>packetErrorRate</td>
<td>double</td>
<td>Average packer error rate for the UE.</td>
</tr>
<tr>
<td>nrOfSuccessfulPackets</td>
<td>int</td>
<td>Number of successfully delivered packets.</td>
</tr>
<tr>
<td>nrOfDelayedPackets</td>
<td>int</td>
<td>Number of successfully but delayed packets.</td>
</tr>
<tr>
<td>nrOfFailedPackets</td>
<td>int</td>
<td>Number of failed packets.</td>
</tr>
<tr>
<td>totalPackets</td>
<td>int</td>
<td>Total number of sent packets.</td>
</tr>
<tr>
<td>totalDeliveredBits</td>
<td>int</td>
<td>Total number of delivered bits.</td>
</tr>
<tr>
<td>roundDeliveredBits</td>
<td>int</td>
<td>Total number of bits delivered last turn.</td>
</tr>
<tr>
<td>totRoundDelay</td>
<td>int</td>
<td>Sum of delays from packets delivered last turn.</td>
</tr>
<tr>
<td>totRoundPackets</td>
<td>int</td>
<td>Total number of packets delivered last turn.</td>
</tr>
</tbody>
</table>

*Table A.19. Properties of the EPSstream class*
A.13.2 Methods

void UpdateStatistics(void)

Input: none

Output: none

Description: Updates the statistics of the EPS-bearer.

void ClearRoundStatistics()

Input: none

Output: none

Description: Clear round specific statistics.

void UpdateUserSatisfaction()

Input: none

Output: none

Description: Checks if the EPS-bearer fulfills the requirements to be satisfying.

A.14 Packet

A.14.1 Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>packetID</td>
<td>int</td>
<td>ID of the packet.</td>
</tr>
<tr>
<td>owningStream</td>
<td>EPSstream*</td>
<td>Reference to the EPS-bearer sending the packet.</td>
</tr>
<tr>
<td>packetSize</td>
<td>int</td>
<td>Size of the packet.</td>
</tr>
<tr>
<td>packetDelay</td>
<td>int</td>
<td>How much time (ms) from arriving to send buffer until delivered.</td>
</tr>
<tr>
<td>packetArrivalTime</td>
<td>unsigned int</td>
<td>Round the packet arrived in send buffer.</td>
</tr>
<tr>
<td>packetDeliveryTime</td>
<td>unsigned int</td>
<td>Round the packet was delivered.</td>
</tr>
</tbody>
</table>

Table A.20. Properties of the Packet class
A.15 Logger

A.15.1 Properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>simE</td>
<td>SimEnvironment</td>
<td>Reference to the instance of the simulation.</td>
</tr>
</tbody>
</table>

Table A.21. Properties of the Logger class

A.15.2 Methods

void PrintRound()

**Input:** none

**Output:** none

**Description:** Prints a round update.

static void PrintCQI(Enode * owner)

**Input:** transmitter

**Output:** none

**Description:** Prints CQI reports for all UE served by the transmitting Enode.

void PrintAllCQI()

**Input:** none

**Output:** none

**Description:** Prints CQI reports for ALL UEs in simulation.

void PrintPathlossMatrix()

**Input:** none

**Output:** none

**Description:** Prints the pathloss between all Enodes and all UEs.

void PrintDLPowerScheduling()

**Input:** none

**Output:** none

**Description:** Prints the scheduled transmitting power to each UE.
static void PrintUserScheduling(Enode * owner)
Input: Enode to print data from
Output: none
Description: Prints the scheduling for Enode.

static void PrintPacketScheduling(Enode * owner)
Input: Enode to print data from
Output: none
Description: Prints which packets have been scheduled by the Enode.

void PrintOwnedWaitingDLPackets()
Input: none
Output: none
Description: Prints the ID and size of all new waiting packets at each Enode.

void PrintOwnedSentDLPackets()
Input: none
Output: none
Description: Prints the ID and size of all sent packets at each Enode.

void PrintChannels()
Input: none
Output: none
Description: Prints the channel for each UE.

void PrintPositions()
Input: none
Output: none
Description: Prints the positions of all nodes in simulation.

void PrintBufferSizes()
Input: none
Output: none
Description: Prints buffer sizes for each UE.
void PrintAllHARQProcesses()
Input: none
Output: none
Description: Prints all HARQ processes for all Enodes.

void PrintOwnedWaitingHARQPackets()
Input: none
Output: none
Description: Prints all packets in HARQ process for each Enode.

void PrintEnvStatistics()
Input: none
Output: none
Description: Prints statistics for the whole environment.

void PrintUEStatistics(UE * user)
Input: reference to UE
Output: none
Description: Prints statistics for the UE.

void PrintFAPStatistics(FAP * base)
Input: reference to FAP
Output: none
Description: Prints statistics for the FAP.

void PrintBSStatistics(BS * base)
Input: reference to BS
Output: none
Description: Prints statistics for the BS.

void PrintAllStatistics()
Input: none
Output: none
Description: Runs the printing of all statistics in the environment.
bool SaveSimulationInfo(string folder)
Input: string with dir to save in
Output: true if successful
Description: Saves some information about the simulation to a txt-file.

bool SaveEnvironmentStats(string folder)
Input: string with dir to save in
Output: true if successful
Description: Saves the statistics regarding the whole environment into a m-file.

bool SavePositions(string folder)
Input: string with dir to save in
Output: true if successful
Description: Saves the positions of all nodes in the simulation into a m-file.

bool SaveFAPStats(string folder, FAP * femto)
Input: string with dir to save in, reference to FAP
Output: true if successful
Description: Saves the statistics regarding the FAP into a m-file.

bool SaveBSStats(string folder, BS * baseStation)
Input: string with dir to save in, reference to BS
Output: true if successful
Description: Saves the statistics regarding the BS into a m-file.

bool SaveUEStats(string folder, UE * user)
Input: string with dir to save in, reference to UE
Output: true if successful
Description: Saves the statistics regarding the UE into a m-file.

bool SaveEPSStats(string folder, EPSstream * stream)
Input: string with dir to save in, reference to EPS-bearer
Output: true if successful
Description: Saves the statistics regarding the EPSstream into a m-file.
void SaveAllStat(void)

**Input:** none

**Output:** none

**Description:** Runs the saving of all statistics.