Scheduling and HARQ

Seminar LTE: Der Mobilfunk der Zukunft

Tobias Schrage

University Erlangen-Nürnberg
Chair of Mobile Communications

January 13, 2010
Outline

1 Scheduling
2 HARQ
3 Literature
Why Scheduling?

Situation
In a LTE cell several UEs are logged in and each one wants to access the channel for data transfer. How can the eNodeB determine which one should be served?
Scheduling Basics

What can be scheduled?
- Space
- Bandwidth
- Code
- Transmit Power

What has to be considered?
- Maximum Delay
- Fairness
- Problem of “User Starvation”
- Efficiency (overall throughput)
- Overhead

Some of these aspects are contradictory
Scheduling Basics

What can be scheduled?
- Space
- Bandwidth
- Code
- Transmit Power

What has to be considered?
- Maximum Delay
- Fairness
- Problem of “User Starvation”
- Efficiency (overall throughput)
- Overhead

Some of these aspects are contradictory
The following scheduling algorithms will be discussed:

- Round Robin
- Maximum Rate
- Proportional Fair Scheduling
Round Robin is the simplest known scheduling algorithm. It allocates each UE equally spaced timeslots, completely ignoring channel quality or data rate.
As we can see in the diagram, the Round Robin Algorithm has some serious drawbacks:

- It is not efficient
- It is not fair in the sense of service quality

Therefore we need better scheduling algorithms.
The Maximum Rate Scheduler always selects the UE with the best data rate for transfer. Therefore it needs information about the expected data rate (which can be derived from the SNR) of each UE.
Maximum Rate II

Advantages

- Very efficient in the sense of overall datarate
- Easy implementation, only information about signal quality is needed.

Disadvantages

- User starvation is very probable
- Impossible to calculate maximum delay, very high delay time possible

→ Practical application is pointless
To avoid “User Starvation” it is possible to add a minimal data rate $r_{min}$ each user needs to receive. This data rate can be assigned to the users in a manner similar to the Round Robin algorithm and thus we obtain a combination of both algorithms.
Basic Idea

Measure the link quality of each UE over a defined time interval and select the UE with the best quality with respect to its average

\[ \hat{i} = \arg \max_i \left\{ \frac{r_{i,n+1}}{\theta_{i,n}} \right\} \]
**Advantages**

- “User Starvation” not very likely
- Better overall throughput than the Round Robin Scheduler

**Disadvantages**

- Information about the channel quality needed
- Higher complexity
- Lower throughput than Max Rate Scheduler
Comparison

![Graphs comparing different scheduling algorithms]

- **CDF of user throughput**
  - Full buffer
  - Web browsing (bursty traffic)

- **Algorithms**
  - Round Robin
  - Proportional Fair
  - Max Rate
Remarks I

Multiple Carriers

In LTE, these algorithms have to be extended to the amount of available resource blocks (OFDM-Subcarriers).

Not part of the specification

Scheduling algorithms are not part of any specification. The necessary interfaces have to be specified (e.g. how to obtain information about channel quality) but not the scheduler itself.
**Quality of Service**

Another important point the scheduler has to consider is QoS: Different services should have different priorities and maybe some users should be served preferential.

**HARQ**

Requested repetition of a packet should be able to prioritise other scheduled packets.
In principle, the same algorithms can be used for the uplink channel. However, some problems have to be considered:

- Limited power of the UEs, hence it is unlikely that a single terminal can use the whole link capacity
Outline

1. Scheduling
2. HARQ
3. Literature
Like every communication channel, the LTE channel is also subject to errors. These might occur due to:

- Variations in channel quality (relatively slow varying)
- Intracell interference and receiver noise (fast varying and unpredictable)
There are two different approaches to error correction:

**FEC - Forward Error Correction**

Introduces redundancy in the transmitted signal by adding parity bits.

*Can only correct a certain number of errors*

**ARQ - Automated Repeat reQuest**

Receiver performs Cyclic Redundancy Check check of the received data. If the received packet is error-free, the reception will be acknowledged and the packet accepted, otherwise a retransmission will be requested.

*Adds delay for every erroneously received bit*
Combination of FEC & ARQ

The best of both worlds

1. FEC: Corrects a certain maximum of errors in the packet
2. ARQ: Performs CRC check on the possibly already corrected packet and requests retransmission on erroneous reception.

→ HARQ (Hybrid-ARQ)
HARQ with Chase Combining

Input to Decoder

Accumulated Energy

- E
- 2E
- 3E
- 4E

Resulting Code Rate

- R = 3/4
- R = 3/4
- R = 3/4
- R = 3/4
**HARQ with Chase Combining II**

**Same Symbols**

If the reception fails, the same symbols will be send again

→ Accumulated Energy increases (and thus SNR improves)

**Code Rate**

Code Rate is untouched, only determined by the FEC Code
HARQ with Soft Combining I

Input to Decoder

Accumulated Energy

Resulting Code Rate

E

2E

3E

4E

R = 3/4

R = 3/8

R = 3/12 = 1/4

R = 1/4
1. The information bits are encoded with a *Mother Code*, typically a low-rate code (R=1/4 in the example).

2. The encoded bits are either punctured or repeated to fit the desired code rate.

3. The receiver decodes the bits and upon error-free decoding accepts the block and notifies the sender, otherwise the next version is requested from the sender.
The information bits are encoded with a *Mother Code*, typically a low-rate code (R=1/4 in the example).

The encoded bits are either punctured or repeated to fit the desired code rate.

The receiver decodes the bits and upon error-free decoding accepts the block and notifies the sender, otherwise the next version is requested from the sender.
The information bits are encoded with a *Mother Code*, typically a low-rate code (R=1/4 in the example).

The encoded bits are either punctured or repeated to fit the desired code rate.

The receiver decodes the bits and upon error-free decoding accepts the block and notifies the sender, otherwise the next version is requested from the sender.
The sender will transmit the next version with redundancy bits and the receiver will try to decode the block, using the information from the old and the new block.

If the decoding fails again, in the example it would be possible to request a third block.

If the decoding is still impossible, repetition will start. The first block will be send again, thus the accumulated energy will be increased and hopefully allow decoding.
The sender will transmit the next version with redundancy bits and the receiver will try to decode the block, using the information from the old and the new block.

If the decoding fails again, in the example it would be possible to request a third block.

If the decoding is still impossible, repetition will start. The first block will be send again, thus the accumulated energy will be increased and hopefully allow decoding.
The sender will transmit the next version with redundancy bits and the receiver will try to decode the block, using the information from the old and the new block.

If the decoding fails again, in the example it would be possible to request a third block.

If the decoding is still impossible, repetition will start. The first block will be send again, thus the accumulated energy will be increased and hopefully allow decoding.
Benefits of HARQ

- FEC can correct a small amount of errors
- ARQ can rely on FEC for packets with few errors and only has to treat packets with many errors
Comparison of ARQ and Link Adaptation

**ARQ**
- Can automatically adapt code rate
- Reacts immediately upon occurrence of an error
- Increases delay time while correcting

**Link Adaptation**
- Adapts code rate explicitly
- Reacts relatively slow
- Does not affect delay
Multiple parallel HARQ processes run in the UE and the eNodeB.

How do receiver and sender know when to perform Soft Combining and when to clear their buffers for a new block?

→ new-data indicator has to be send
Asynchronous, adaptive HARQ

- **Asynchronous**: No strict timing, retransmission packet is scheduled like any other packet, normally with higher priority.

- **Adaptive**: The retransmission can use other resources (frequency, modulation) than the original packet.

For each block, the HARQ process-id and the redundancy version number has to be send via the **PDCCH** (Physical Downlink Control Channel). The ACK/NAK message is sent through the uplink channel.
**Synchronous, non-adaptive HARQ**

- **Synchronous**: Retransmission after a fixed interval (8 packets)
- **non-adaptive**: Same resources have to be used again.

- The only necessary feedback to the sender is a single bit, with information whether to resend the packet or not.
- There is also a possibility to explicitly define the retransmission for another frequency block, but this is **not** the usual behaviour.
How to support both modes?

There are two ways to request a retransmission:

- **PHICH (Physical Hybrid-ARQ Indicator Channel):** The receiver can request the retransmission using the same resources.

- **PDCCH:** The eNodeB can use the PDCCH to allocate another frequency for the retransmission or to request a certain redundancy version of the packet.


Scheduling and HARQ

Seminar LTE: Der Mobilfunk der Zukunft

Tobias Schrage
University Erlangen-Nürnberg
Chair of Mobile Communications

January 13, 2010