NOMA simulation based on OAI

Xinghe Chu
Joint Open5G Lab
1. Functions in Downlink Simulation

2. Data in Downlink Simulation

3. Process of NOMA Simulation

4. Result of Simulation

5. Shortcomings and Expectation
1. Functions in Downlink Simulation

begin
configure parameter in LTE
DCI initialization
eNB transmission phy_procedures_eNB_TX
OFDM modulation do_OFDM_mod_l

simulation
DL_channel

end
BLER statistic
UE receive phy_procedures_UE_RX
slot_fep
eNB transmission-> phy_procedures_eNB_

• The details of the function:

- generate data of users
  - get_dlsch_sdu

- DCI
  - gerenate_dci_top

- pdsch process
  - pdsch_procedures

- generate DCI
  - gerenate_dci0

- pdcch scrambling
  - pdcch_scrambling

- pdcch interleaving
  - pdcch_interleaving

- pdcch mapping

- pdcch encoding
  - dlsch_encoding

- scrambling
  - dlsch_scrambling

- modulation
  - dlsch_modulation

- generate pilots information
  - generate_pilots
The data of users are random numbers, which is saved in `input_buffer0` or `input_buffer1`.

The random numbers are generated by the function `unsigned int tau(void)` and the operator `and(&)`.

And the function `get_dlsch_sdu` will return the array `input_buffer0` or `input_buffer1` based on the parameter `Tbindex` is 0 or 1 (when `Tbindex` is 0, `input_buffer0` will be returned, or `input_buffer1` will be returned).
The file dlsim.c can simulate many kinds of channels, including SCM_C channel, SCM_D channel, EPA channel, EVA channel, ETU channel, MBSFN channel, Rayleigh channel, Rice channel and AWGN channel.

And the model of the simulation is just like \( y = x \otimes h + n \).

In the model, \( x \) is the data to be transmitted, \( h \) is channel impulse response, \( n \) is white Gaussian noise and \( y \) is the receiving data.

In the simulation of NOMA, I set the channel as AWGN channel, so the channel impulse response \( h_{\text{awgn}} \) is 1, and the model is \( y = x + n \).
• The details of the function:

- **pdcch procedure**
  - receive pdcch signals `rx_pdcch`
  - decode dci `dci_decoding`

- **pdsch procedure**
  - receive pdsch signals `rx_pdsch`
  - extract subcarrier `dlsch_extract_rbs_single`
  - channel compensation `dlsch_channel_compensation`
  - `dlsch_qpsk_llr`

- **dlsch procedure**
  - unscramble `dlsch_unscrambling`
  - decode `dlsch_decoding`
2. Data in Downlink Simulation

- **input_buffer**
  - dlsch_encoding
  - eNB->dlsch>
    - harq_processes[h
      - ar_pid]->e
    - dlsch_scrambling
    - eNB->dlsch>
      - harq_processes[h
        - ar_pid]->e
      - dlsch_modulation

- **UE->common_vars.r**
  - DL_channel
    - do_OFDM_mod_l
    - generate_pilots
    - eNB->common_vars.t
    - xdataF
    - dl_ch_estimates
    - llr
    - rx_pdsch
dlsch_unscrambling
    - llr
dlsch_decoding
    - UE->dlsch_eNB[
      - eNB_id]->
      - harq_processes->b[i]
3. Process of NOMA Simulation

- Step 1: There are 2 UEs in my simulation, the channel of one is good, and the other is bad.
- Step 2: After encoding, scrambling, modulating and IDFT of the data from different users, we can reach two txdata array. Txdata1 is the data from UE1 with good channel, and txdata2 is the data from UE2 with bad channel.
- Step 3: So the new array $txdata = (\alpha * txdata1 + \beta * txdata2) / (\alpha + \beta)$, $\alpha < \beta$.
- Step 4: Send txdata to DLchannel, we can reach rxdata.
◆ Step 5: For UE2, we can directly use the array rxdata and send it to the function phy_procedures_UE_RX to get the result.

◆ Step 6: For UE1, firstly, we need send rxdata to phy_procedures_UE_RX to get the data of UE2. And then we send the data we got to phy_procedures_eNB_TX. So we can reach a new txdata array, we call it txdata'. Then we can reach a new rxdata called rxdata' by the formula.

\[
rxdata' = \frac{rxdata \times (\alpha + \beta) - txdata' \times \beta}{\alpha}
\]

At last, we send the array rxdata' to phy_procedures_UE_RX, and we can get the data of UE1.
4. Result of Simulation

In my simulation, the parameters are set as follows:

<table>
<thead>
<tr>
<th>parameter</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>duplex mode</td>
<td>FDD</td>
</tr>
<tr>
<td>transmission mode</td>
<td>TM1(SISO)</td>
</tr>
<tr>
<td>DL carrier frequency</td>
<td>2.59GHz</td>
</tr>
<tr>
<td>system bandwidth</td>
<td>5MHz</td>
</tr>
<tr>
<td>channel</td>
<td>AWGN</td>
</tr>
</tbody>
</table>
- picture1 and picture2 are the constellations of UE1 and UE2 (the array txdataF after dlsch_modulation).
- picture3 is the constellation of UE1 again (in step 6, we need to use the data of UE1 to reach txdata', and this is the constellation of txdataF after dlsch_modulation).
This is the constellation of rxdataF just after going through AWGN Channel.
This is the constellation of rxdataF'.

constellation
In this simulation, we don't have the picture of error rate because when the result is wrong, the result is 0 all the time and when the simulation system can reach the result, the result is always totally the same as the user's data (input_buffer). So when the simulation system cannot reach the result, the error rate is always around 50% and 0 otherwise.

But I did another work---the relation between power allocation and SNR. We can see, when the power allocation is settled, we can always find a minimum SNR of UE1 with good channel that makes the result right. So this SNR is called mSNR.
The x axis is the rate of alpha/beta, and y axis is mSNR(dB).

We can see from the first picture, as the increasing of alpha/beta, the mSNR decreases. But it stops at 0.8--0.9.

So I increase the precision of alpha/beta just like picture2. And we can see it goes irregular in this range.

And after 0.92, the result goes wrong.

So we can make the conclusion that in that simulation, the best power allocation rate is from 0.8 to 0.9, which allows channels with SNR around 10dB.
5. Shortcomings and Expectation

- **Shortcomings:**
  This simulation can only simulate the most ideal scene in noma system, just like AWGN channel, SISO and so on. So it can only adapt to the ideal scene and some of the result may not be realistic in practice.

- **Expectation:**
  On the one hand, I will do more work on the simulation, so I will conclude more useful things, on the other hand, I will try to build NOMA system in real world.
THANK YOU