**Industrial track: task description**

**Channel State Information Compression**

**All questions and solutions shall be addressed to Alexander Sherstobitov:** [**sherstobitov.alexander@huawei.com**](mailto:sherstobitov.alexander@huawei.com)

**Link for the simulation platform:** [**https://disk.yandex.ru/d/\_uz0b011Mw4vJA**](https://disk.yandex.ru/d/_uz0b011Mw4vJA)

Assume, we have measurements of system with multiple input and multiple output (Figure 1):

Let’s name as channel propagation matrix in which each element is a complex coefficient of transition between node to node (, where , see Figure 1b).

In general, input/output nodes ( from Figure 1b) can be organized in some structure: uniform linear antenna array (ULA), uniform rectangular antenna array (URA), uniform circular antenna array (UCA) or special design. On the figure 1a you can see example of URA structure of nodes on base station (BS) / user equipment (UE) sides (Appendix A).

Channel measurement describing instant conditions of wireless signal propagation in the narrow frequency band. In general assumption, variate in frequency domain as much as more clusters (Figure 1a) of reflection are present (Line of Site (LoS) vs Non Line of Site (NLoS) channel conditions).

Channel measurement defined in follow dimensions: input nodes (antennas of transmitter (TX) / antennas of receiver (RX) over frequency domain (subcarriers ))

,

where , is subbands / subcarriers number in the frequency domain.

|  |  |
| --- | --- |
|  |  |

1. b)

Figure 1. Example of wireless channel propagation modeling a) and representation as multipole system

Channel measurements is a three dimensional array (tensor), which needs to be forward from UE side to BS side. On the BS side, received feedback of channel measurement is require for beam forming design, MU-MIMO transmission preparation, multi-user interference calculation and so on.

System model of wireless signal transmission defined as follow

where is a received signal on RX antennas for fix frequency , - pilot signal to obtain channel estimation , is vector of noise , which refer to receiver sensitivity (not present on the figure 1b). Let assume that we have perfect channel estimation , and will deal with directly.

The equation (\*) is channel measurement equation. However, data transmission corresponds to following one

Pay attention that it is required to transmit single data stream in (\*\*). Thus it can be proof that optimal transmission scheme corresponds to

where is a first eigenvector of channel in the space domain ( TX antennas). Thus, it is enough to use just the first eigenvector to be sure in system efficiency.

**Goals of this task**: to propose algorithm of compression/decompression of channel measurements , which allows to achieve follow requirements:

* Compression/decompression algorithm must be “lossless”:

,

where:

, where andis a compression/decompression algorithm respectively.

* Compression rate must be more, then 3 times

,

where is operator of bits calculation (size of file on the HDD), is bits number per REAL/IMAGE component of complex number.

Initial baseline size of information bits , which require for storing channel measurement is defined as , where bits per each REAL/IMAGE components.

Defining baseline minimum requirements to algorithms of compression/decompression of channel measurements :

(1)

**Initial data and simulation platform.**

As initial data will provide channel measurements in follow format:

* Array of complex number ( of double format). Usual parameters variate and can be limit as following:

.

* Channel measurements represent by set of independent realizations . You can use it realization for testing and debugging of compression/decompression algorithms, which outperform baseline requirements (1).
* Optional may be using additional source of data (**Channel Generator**), which allows to extend realization size () for testing and verification algorithms of compression/decompression.

Simulation platform and requirements for algorithms compression/decompression implementation:

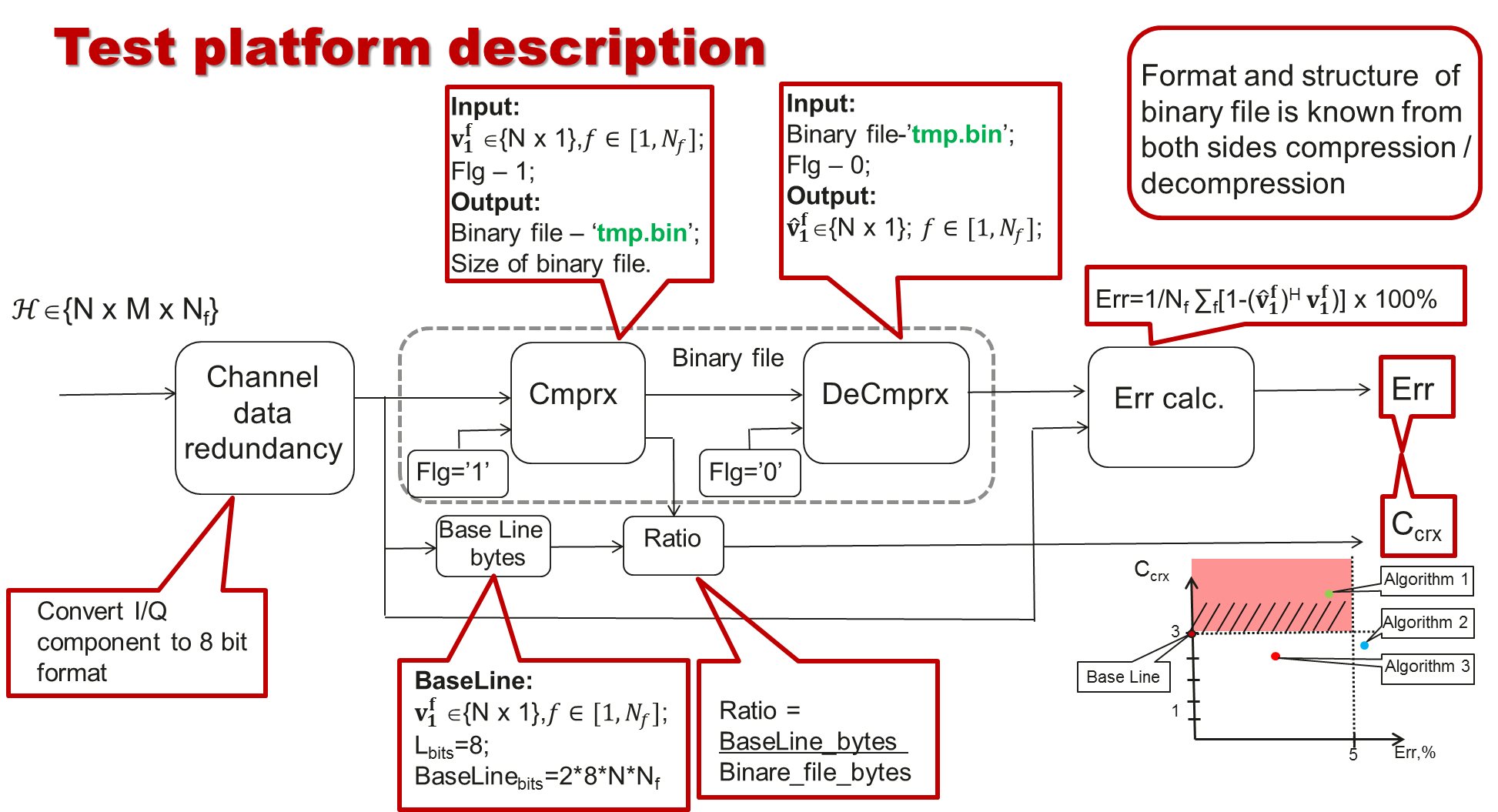
* All simulation and algorithm compression/decompression must be implement on **Octave/Matlab (without using any toolboxes and non standard functions)**
* Extended channel data may be generate based on the provided ‘Channel Generator’ (support **Octave/Matlab**)
* Simulation platform for comparison consist of several part (Figure 2):

1. Input channel data redundancy – convert samples of channel **:** REAL/IMAGE components convert to 8 bits format, extract from **,**  for compression
2. Test set of algorithms of channel compression/decompression:
3. Algorithms of channel compression/decompression is implemented in same file and switched by flag (‘1’ is compression, ‘0’ is decompression).
4. Structure/format of input and output data predefined by simulator (need to use example of code):

- **Input variables**: channel data in the 8 bits format of IQ components

- **Output**: and binary file ‘**tmp.bin**’ with – compressed version of

1. All parameters and initializations must be declare inside same file (as functions/structures)
2. Structure of binary file ‘**tmp.bin**’ can be defined by yourself.
3. Algorithms comparison based on the two metrics (1). First metric is a necessary delivery condition (Err<5%). Second metric is a sufficient condition () for verification

****

**Figure 2. Simulation test platform structure**

**Final results and comparison**

Winner selection is based on the objective comparison of different algorithms of compression / decompression in reference (1). All algorithms, which can achieve first part of criteria (1) to admit for comparison by second part of criteria (1).

All passed algorithms-candidate will ranking by second part of criteria (1) – compression ratio, and additionally tested on the extended test sequence of channel measurement (out of channel generation – other seeds and field test data).

Final ranking allows to choose winners of compression/decompression algorithm design.

**Remarks:** If we will find more, then one best of all algorithm with same parameters of , so will compare by additional parameters: comparison and Err[%] scenario by scenario.

**Appendixes**

* + 1. **Antenna array design**



Figure 3. Antenna array design from BS and UE side

* + 1. **Template of file for implementation compression/decompression algorithms**

Compression /decompression algorithm must be implemented in function with follow structure :

Example of file with algorithms compression/decompression

File: ‘AlgChannelCompxDecomprx.m’

----------------------------------------------------------------------------------------

function [out, cmprxData\_bytes]=AlgCompressDecompress(in, flg\_direction)

% ---------------------------- Description ------------------------------------------

Inputs:

in – input data of size [N x Nf x T] (for compression: flg\_direction=1)

in – full path to file ‘tmp.bin’ (for decompression: flg\_direction=0)

flg\_direction – flag of compression / decompression => respectevly (1/0)

Outputs:

out - compressed version of input data ‘in’ (for compression: flg\_direction=1)

out - decompressed version of input data ‘in’ (for decompression: flg\_direction=0)

cmprxData\_bytes - size of compressed channel in bytes

% -----------------------------------------------------------------------------------

% ------------- Initialization compression/decompression algorithm ------------------

cmp = Initialization(); % if necessary to initialize parameters or define class

% -----------------------------------------------------------------------------------

if flg\_direction

% --------------- Compressing procedures **(Example)** ------------------------

out = cmp.compress(in); % compressing

out.Parameter1 = Parameter1;

out.Parameter2 = Parameter2;

.

.

.

out.ParameterN = ParameterN;

cmprxData\_bytes = Byte\_calc(out,Param);

else

% --------------- DeCompressing procedures **(Example)**-------------------

out = cmp.restore(in); % decompressing

cmprxData\_bytes = 0;

end

end

classdef Initialization % **Example**

% ---------------------------- Description ------------------------------------------

% Here can be defined structure, variables, which can help

% to evaluation compressed and decompressed version of channel measurements

properties

…

end

methods

…

end

.

.

.

end

function bt = Byte\_calc(in,prm) % **Example**

% ---------------------------- Description ------------------------------------------

Input:

in – input compressed data for saving into file.

prm – input parameters for save assistance

Output:

bt - size of file ‘tmp.bin’ in bytes

% -----------------------------------------------------------------------------------

fid=fopen('tmp.bin','w');

fwrite(fid,in.field1,['ubit' X]);

fwrite(fid,in.field2,['ubit' X]);

.

.

fwrite(fid,in.fieldN,['ubit' X]);

fwrite(fid,in.Parameters1,['ubit' X]);

fwrite(fid,in.Parameters2,['ubit' X]) ;

.

.

.

fwrite(fid,in.ParametersN,['ubit' X]) ;

fclose(fid);

fl=dir('tmp.bin');

bt=fl.bytes;

delete('tmp.bin');

end