LDPC Error Floor Prediction using Trapping Set aware Code Shortening

D. Declercq, B. Vasic, B. Reynwar, V. Yella, S. Planjery

March 2019

This work was supported by the National Science Foundation under SBIR Phase II Grant 1534760.

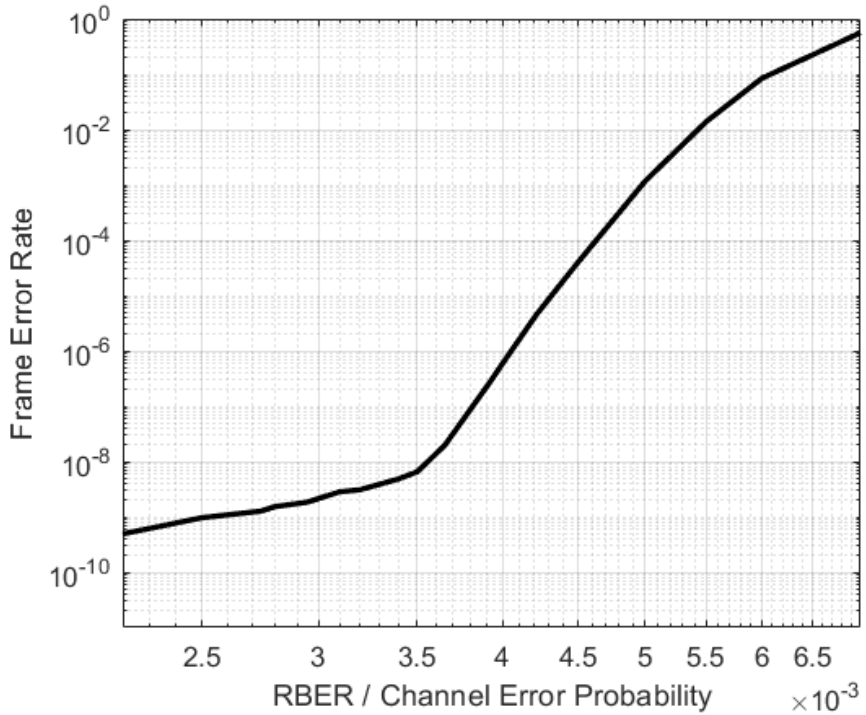
Outline

Outline

Error floor Problem in LDPC Decoders

Tom Richardson (Allerton 2003)

- An abrupt degradation of FER at low RBER caused by a failure of an iterative decoder to converge to a codeword
- Error floor is attributed to dense subgraphs present in the Tanner graph : Trapping Sets $\tau(a, b)$
- $\tau(a, b)$: a set of not eventually correct variabe nodes of size a, inducing a subgraph of b odd degree check nodes.



Error floor Problem in LDPC Decoders

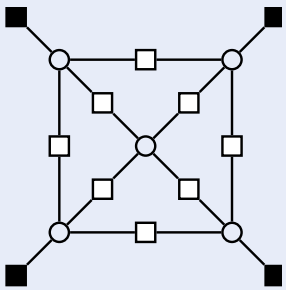
Vasic (Allerton 2005, ICC 2006)

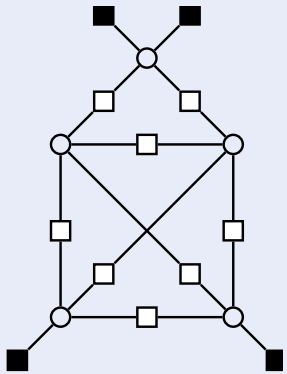
- The harmfulness of Trapping Sets is based on uncorrectable error patterns on isolated Trapping Sets.
- **Critical Number of a Trapping Set : c_τ**

  minimum number of errors in $\tau(a, b)$ (out of a) which causes failure, when $\tau(a, b)$ is **isolated from the rest of the Tanner graph**
- **Strength of a Trapping Set : s_τ**

  number of error patterns of weight c_τ bits, for which the decoder fails on the **isolated** $\tau(a, b)$

Two examples of $\tau(5, 4)$ Trapping Sets





Outline

Major Flaw of Existing Trapping Set Characterization

Subgraphs that are "believed" to be harmful

- Whether a Trapping Set is harmful depends on **a decoder** and its **neighborhood** in the Tanner graph
- For simple decoders (BF, Gal-B), Trapping Sets can be treated isolated from the rest of the graph
- For stronger decoders (min-sum, FAID), an isolated Trapping Set is not sufficient to predict its impact on error floor
- Trapping Sets with the same values of a and b can be **either** *harmful* **or** *not harmful*

We propose a new characterization of harmfulness to take into account

1- The **neighborhood** of the Trapping Set within a particular QC-LDPC code,

2- The **particular message passing decoding** rules $\phi = (\phi_V, \phi_C)$ that are used for decoding.

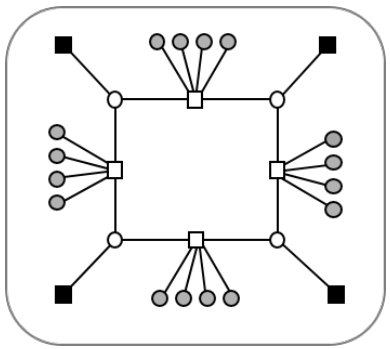
Advantages of our approach

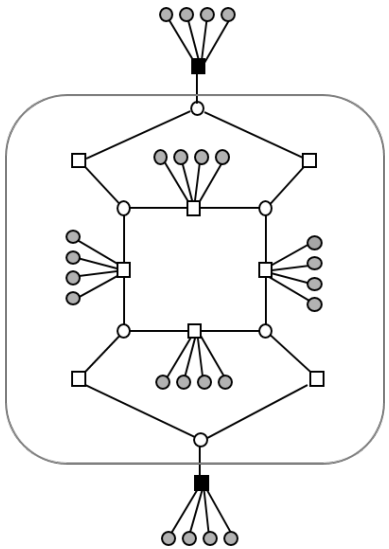
- The harmfulness of $\tau(a, b)$ will depend on the sparseness and the **topology of its neighborhood**.
- By considering a particular decoding rule, we will consider structures that are harmful *only* **for this particular decoder**
- We do not consider that a TS can be "universally" harmful, as it is often assumed in existing works.

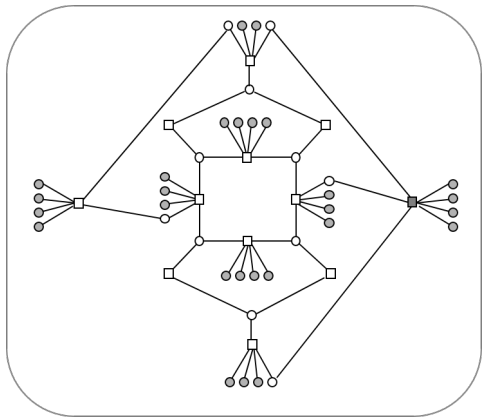
New Characterization of Harmfulness : expansion-contraction procedure

Step 1 : Expansion (neighborhood dependent)

- Let us consider a small Trapping Set $\tau(a, b)$ (not necessarily harmful) - or a single cycle - in a given LDPC code,
- The trapping set $\tau(a, b)$ is expanded by adding data node neighbors to the graph as long as they create **new cycles**,
- The expansion is recursively repeated until no new cycles can be created,
- After expansion, we obtain a larger structure $\mathcal{T}(A, B)$, with more cyles than the original TS,



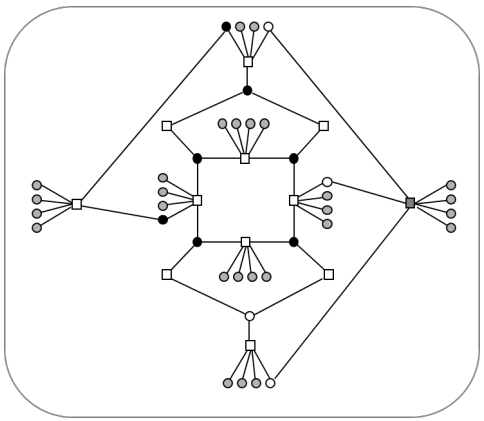


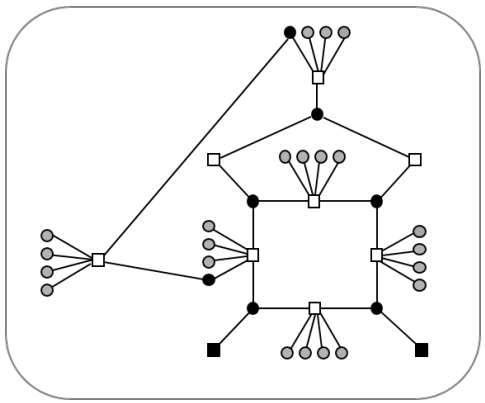


New Characterization of Harmfulness : expansion-contraction procedure

Step 2 : Contraction (decoder dependent)

- We consider the expanded Trapping Set $\mathcal{T}(A, B)$, and compute its critical number $\mathbf{c}_{\mathcal{T}}$ and strengh $\mathbf{s}_{\mathcal{T}}$,
- Let $\{\mathbf{e}_1, \ldots, \mathbf{e}_s\}$ be the set of error patterns of weight $c_{\mathcal{T}}$ (indicated in black circles),
- Define $\mathcal{S}$ as the union of the support of all error patterns : $\mathcal{S}\left(\mathbf{e}_1, \ldots, \mathbf{e}_s\right) = \bigcup_{k=1}^{s} \mathcal{S}(\mathbf{e}_k)$
- $\mathcal{S}$ is composed of all the bits in $\mathcal{T}(A, B)$ which participate in *at least* one decoding failure.





Definition

The subgraph corresponding to $\mathcal{S}$ is declared as **the harmful TS**, *denoted* $\tau_h\,(a_h, b_h)$

Harmfulness Spectrum of a LDPC Code

List of Harmful Trapping Sets

- Let $\mathcal{T} = \left\{ \tau_{h_1}, \tau_{h_2}, \ldots, \tau_{h_T} \right\}$ be the set of harmful Trapping Sets selected with expansion-contraction procedure
- $\mathcal{T}$ is identified from **large Trapping Sets** in an actual QC-LDPC code, *i.e.* not isolated from their neighborhood,
- $\mathcal{T}$ is identified by decoding error patterns using a particular message passing decoder, *i.e.* the set $\mathcal{T}$ differs from one decoder to another.

Harmfulness Characterization

- For each $\tau_{h_i} \in \mathcal{T}$, we consider its harmfulness as $\left(\mathbf{c}_{\tau_h}, \mathbf{s}_{\tau_h} \right)$
- Structures with the **smallest critical number** c_{τ_h} are the *most harmful*
- Within the structures with same $\mathbf{c}_{\tau_h}$, the ones with **largest strengh** $\mathbf{s}_{\tau_h}$ are the *most harmful*
- Relative harmfulness of two trapping sets with same critical numbers is the ratio of their strengths

Harmfulness Ranking

Using our expansion-contraction procedure, we obtain an **exact ranking** *of the harmful Trapping Sets, for a given LDPC code, and a given decoder.*

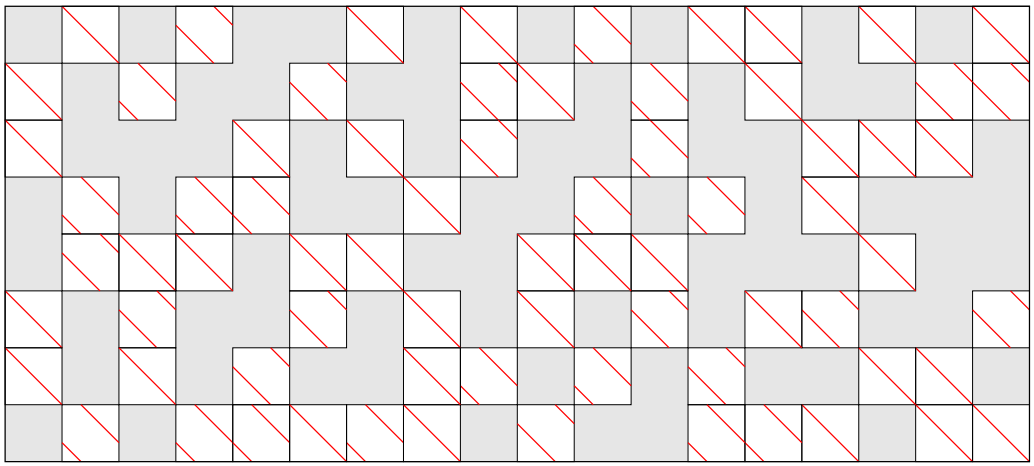
Outline

Shortening procedure

- Regular Quasi-cyclic LDPC (QC-LDPC) code defined by its PCM $\mathbf{H}$, organized in N_b *block-columns*,

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_{:,1} & \mathbf{H}_{:,2} & \dots & \mathbf{H}_{:,N_b} \end{bmatrix}$$

- $\mathbf{H}_{:,i}$ contains d_v circulant blocks, and $(M_b - d_v)$ all zero blocks,
- Select s block-column indices $\mathbf{i} = [i_1, \dots, i_s]$, with $M_b < s < N_b$
- Shortening = Extracting from $\mathbf{H}$ the corresponding block columns.
- The shortened code $\mathbf{H_{short}} = \begin{bmatrix} \mathbf{H}_{:,i_1} & \mathbf{H}_{:,i_2} & \dots & \mathbf{H}_{:,i_s} \end{bmatrix}$ has rate $R_{short} = 1 - M_b/s < 1 - M_b/N_b$.

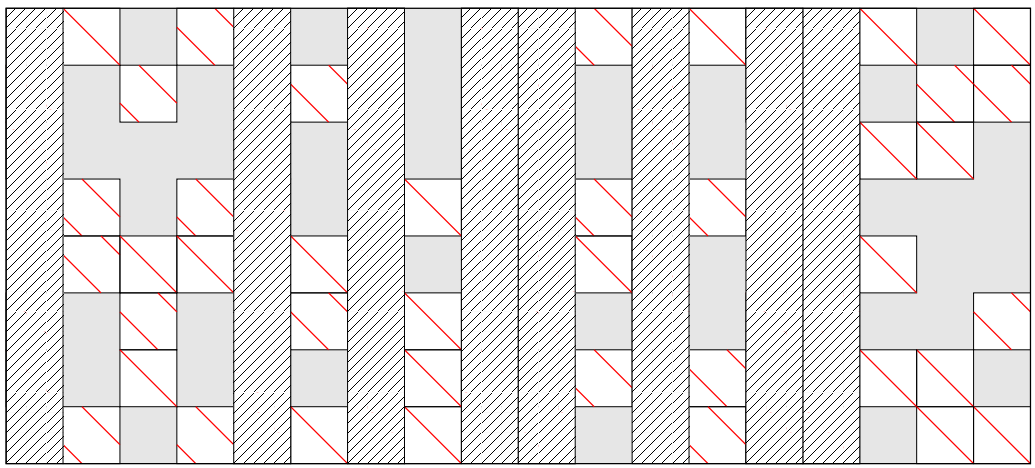


Shortening procedure

- Regular Quasi-cyclic LDPC (QC-LDPC) code defined by its PCM $\mathbf{H}$, organized in N_b *block-columns*,

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_{:,1} & \mathbf{H}_{:,2} & \dots & \mathbf{H}_{:,N_b} \end{bmatrix}$$

- $\mathbf{H}_{:,i}$ contains d_v circulant blocks, and $(M_b - d_v)$ all zero blocks,
- Select s block-column indices $\mathbf{i} = [i_1, \dots, i_s]$, with $M_b < s < N_b$
- Shortening = Extracting from $\mathbf{H}$ the corresponding block columns.
- The shortened code $\mathbf{H_{short}} = \begin{bmatrix} \mathbf{H}_{:,i_1} & \mathbf{H}_{:,i_2} & \dots & \mathbf{H}_{:,i_s} \end{bmatrix}$ has rate $R_{short} = 1 - M_b/s < 1 - M_b/N_b$.

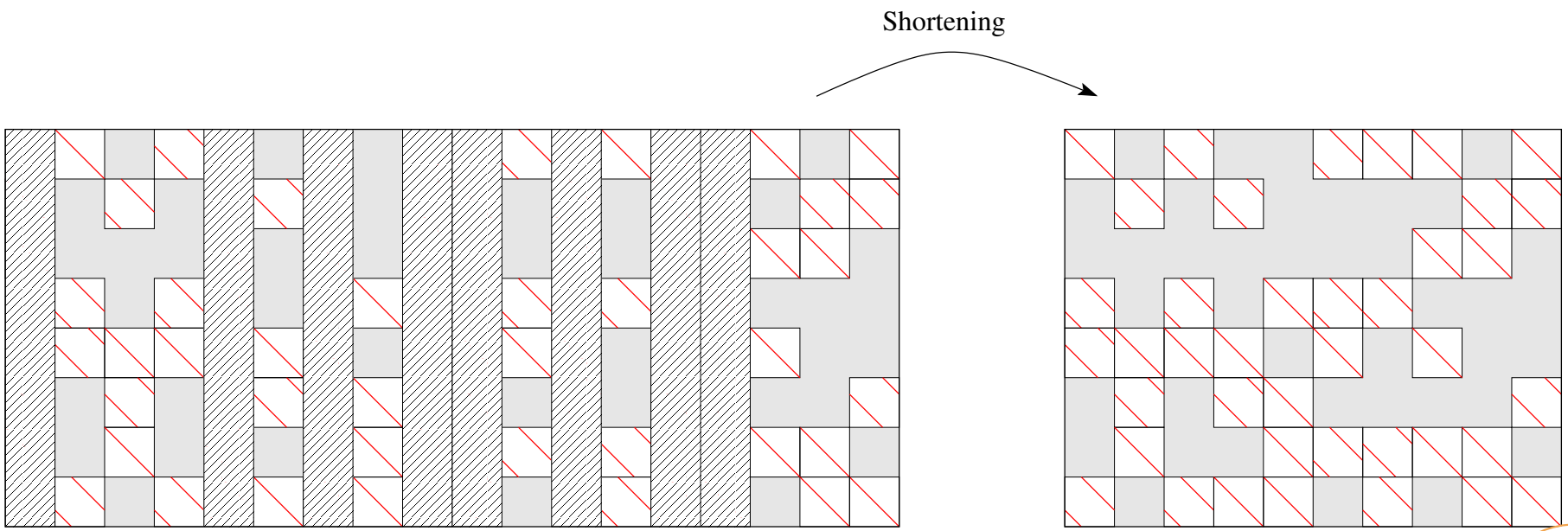


Shortening procedure

- Regular Quasi-cyclic LDPC (QC-LDPC) code defined by its PCM $\mathbf{H}$, organized in N_b *block-columns*,

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_{:,1} & \mathbf{H}_{:,2} & \dots & \mathbf{H}_{:,N_b} \end{bmatrix}$$

- $\mathbf{H}_{:,i}$ contains d_v circulant blocks, and $(M_b - d_v)$ all zero blocks,
- Select s block-column indices $\mathbf{i} = [i_1, \dots, i_s]$, with $M_b < s < N_b$
- Shortening = Extracting from $\mathbf{H}$ the corresponding block columns.
- The shortened code $\mathbf{H_{short}} = \begin{bmatrix} \mathbf{H}_{:,i_1} & \mathbf{H}_{:,i_2} & \dots & \mathbf{H}_{:,i_s} \end{bmatrix}$ has rate $R_{short} = 1 - M_b/s < 1 - M_b/N_b$.



Code Shortening to Estimate Error Floors

Design the **worst possible** shortened code

- Build a shortened code $\mathbf{H}_{\mathbf{short}}$ which contains the same most harmful Trapping Sets as the original code $\mathbf{H}$, and run Monte Carlo simulations on $\mathbf{H}_{\mathbf{short}}$ to detect its error floor
- If the decoder fails on the same structures for $\mathbf{H}_{\mathbf{short}}$ and $\mathbf{H}$, both curves will have an error floor **with the same slope**, but since $\mathbf{H}_{\mathbf{short}}$ has lower rate, the error floor will appear at a higher FER, resulting in computational savings
- Chosing properly the shortened columns is critical for the efficiency of our approach.
  - ◇ When s is large, close to N_b, it is easier to ensure that the harmful TSs of $\mathbf{H}$ will remain in $\mathbf{H}_{\mathbf{short}}$, but computational saving would be small
  - ◇ With a too small value of s, $\mathbf{H}_{\mathbf{short}}$ might not contain anymore the harmful TSs, resulting in an erroneous prediction of the error floor
- Our strategy is then to minimize the number of block-columns kept, while still ensuring that the harmful TSs are present in $\mathbf{H}_{\mathbf{short}}$
- This optimization problem is closely related to the well studied weapon-target assignment problem and the hypergraph demand matching problem

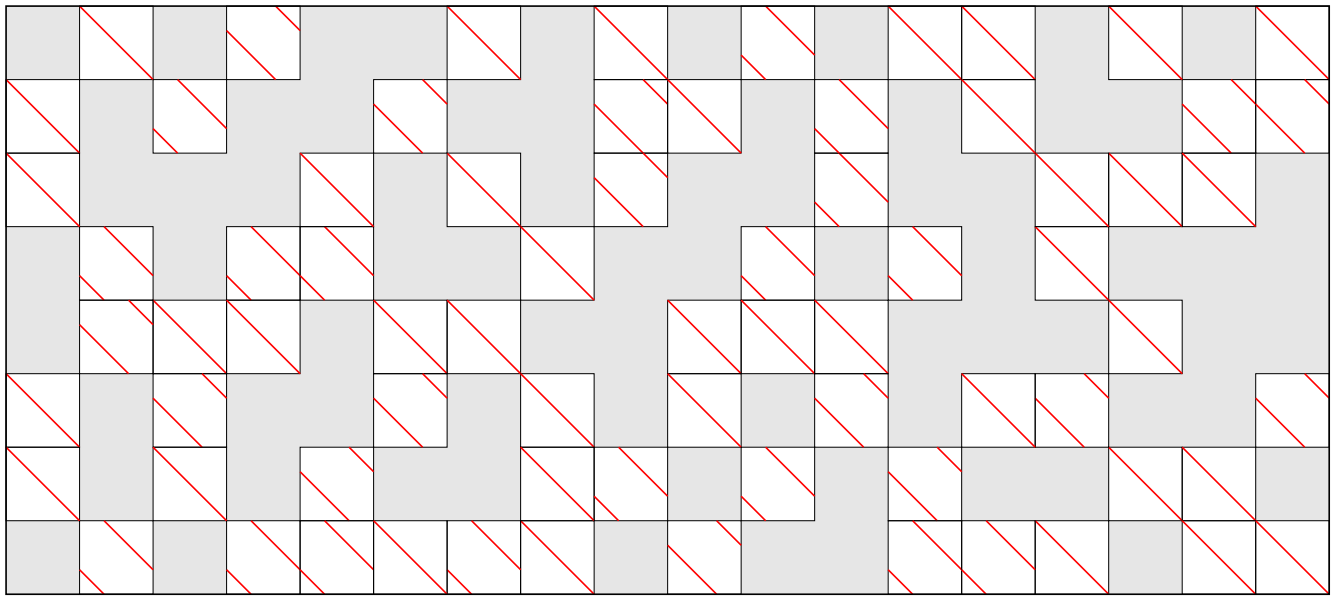
Design the **worst possible** shortened code

- We propose a pragmatic and simple approach to perform the shortening optimization

Target QC-LDPC Code

Example of LDPC code with Nb=18 columns and Mb=8 rows

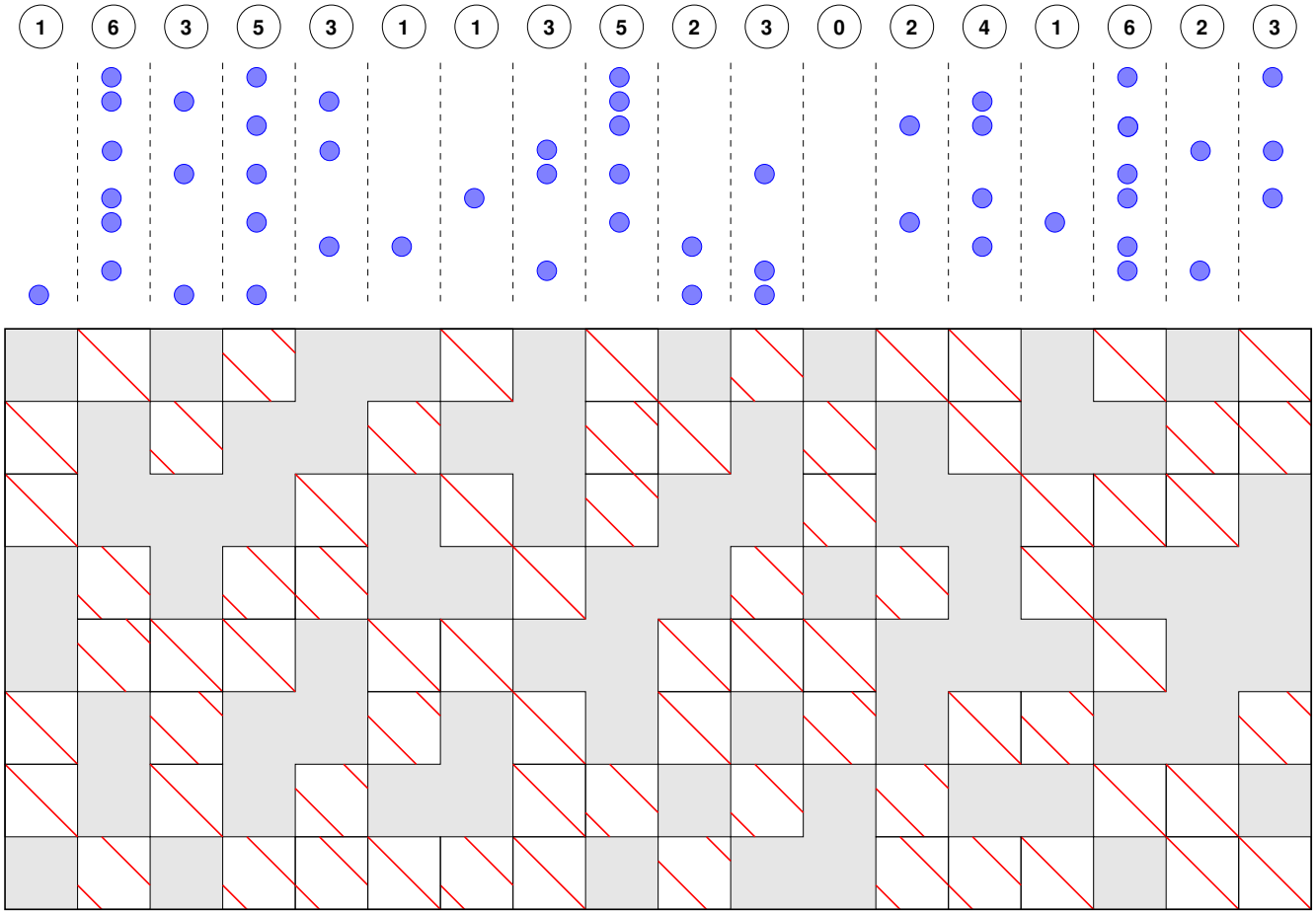
- Compute for this code a list of the $K = 10$ most harmful TS, with the expansion/compression method



Shortening STEP-1 : Accumulate Trapping Sets Indices

Shortening STEP-1

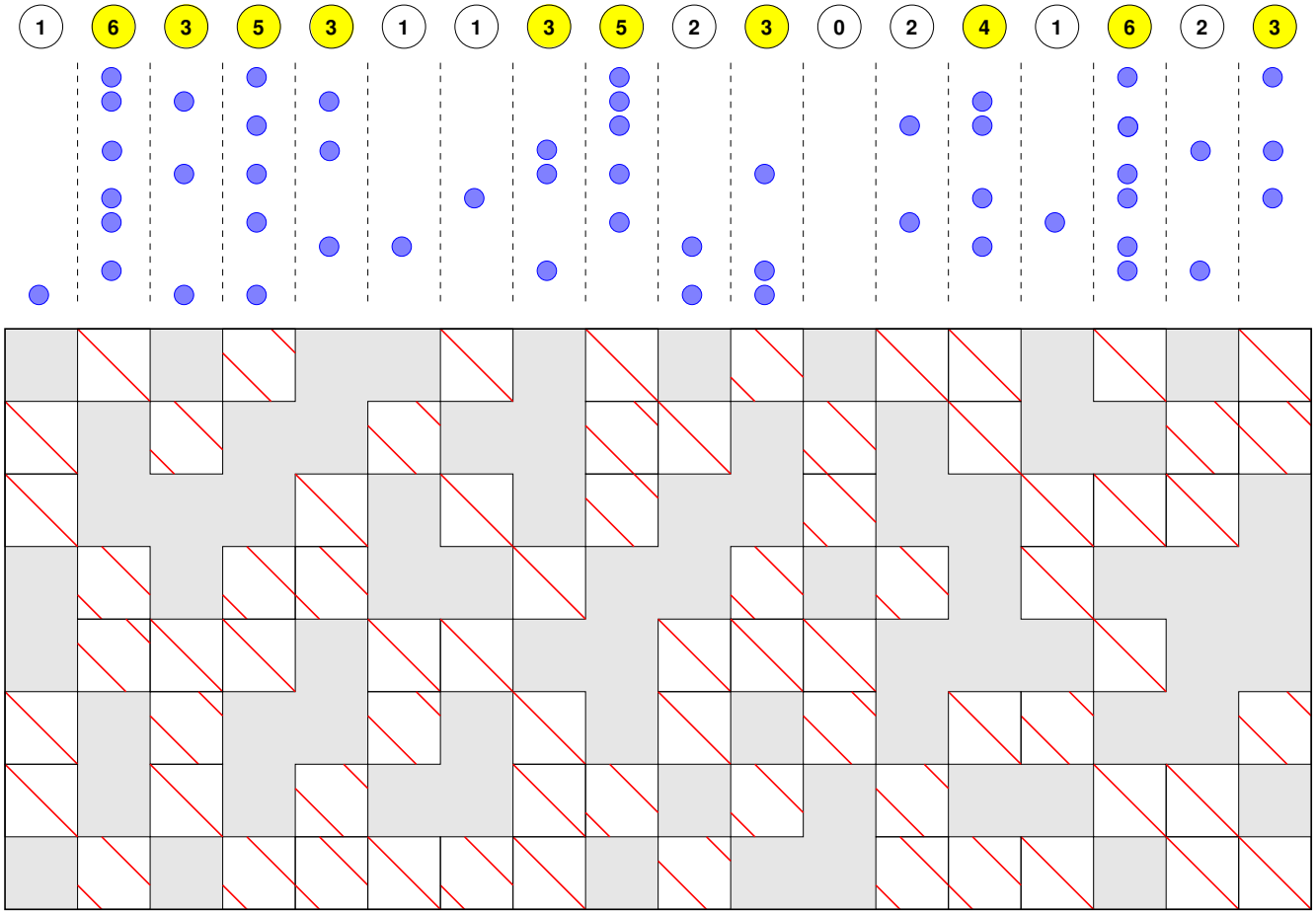
- The *i*-th block-column harmfulness = sum of harfmulnesses of all Trapping Sets having a variable node in it.



Shortening STEP-2 : Select the Worse Columns

Shortening STEP-2

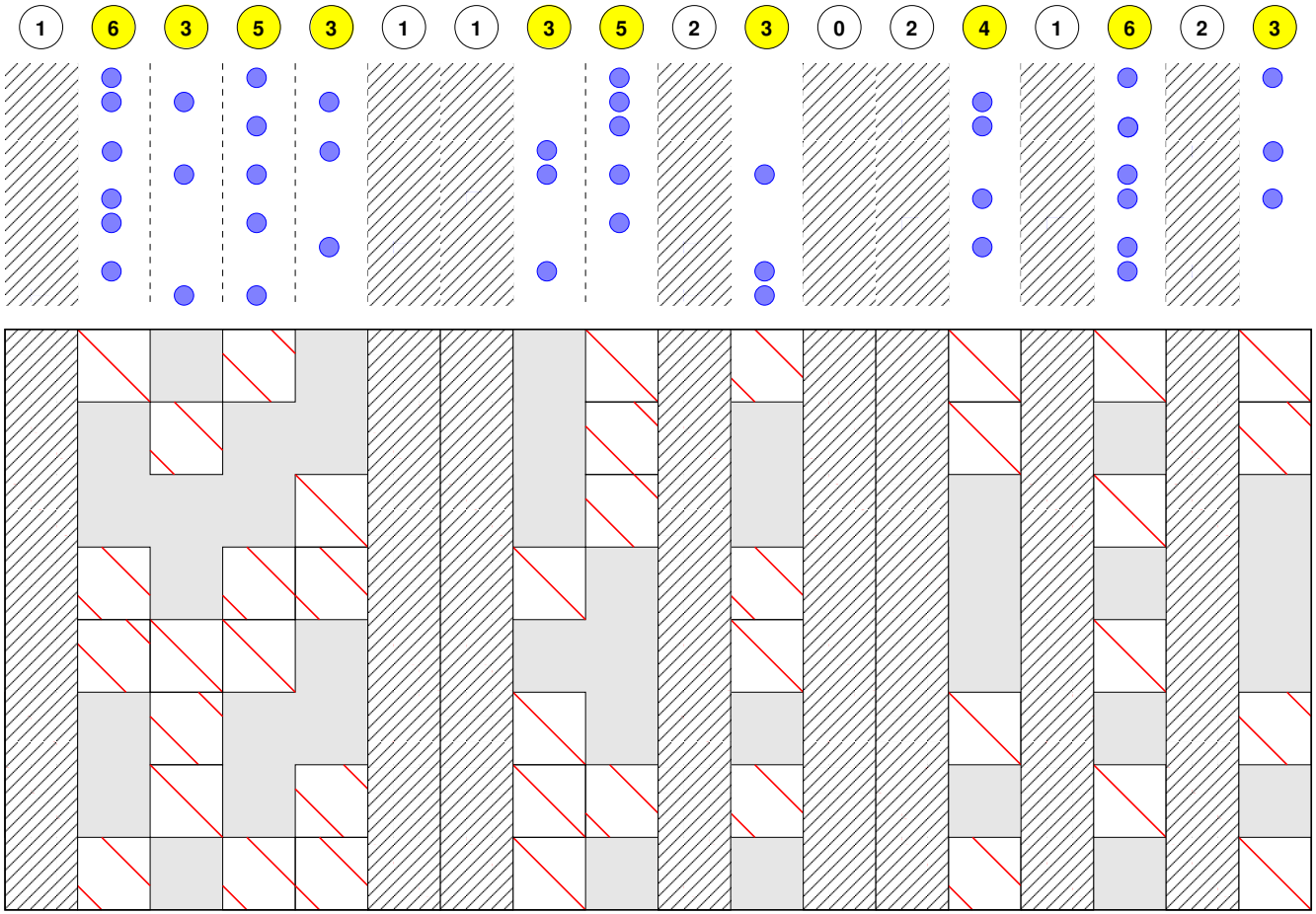
- Select the block-columns which have total harmfulness greater than a threshold t.



Shortening STEP-3 : Shorten Best Columns

Shortening STEP-3

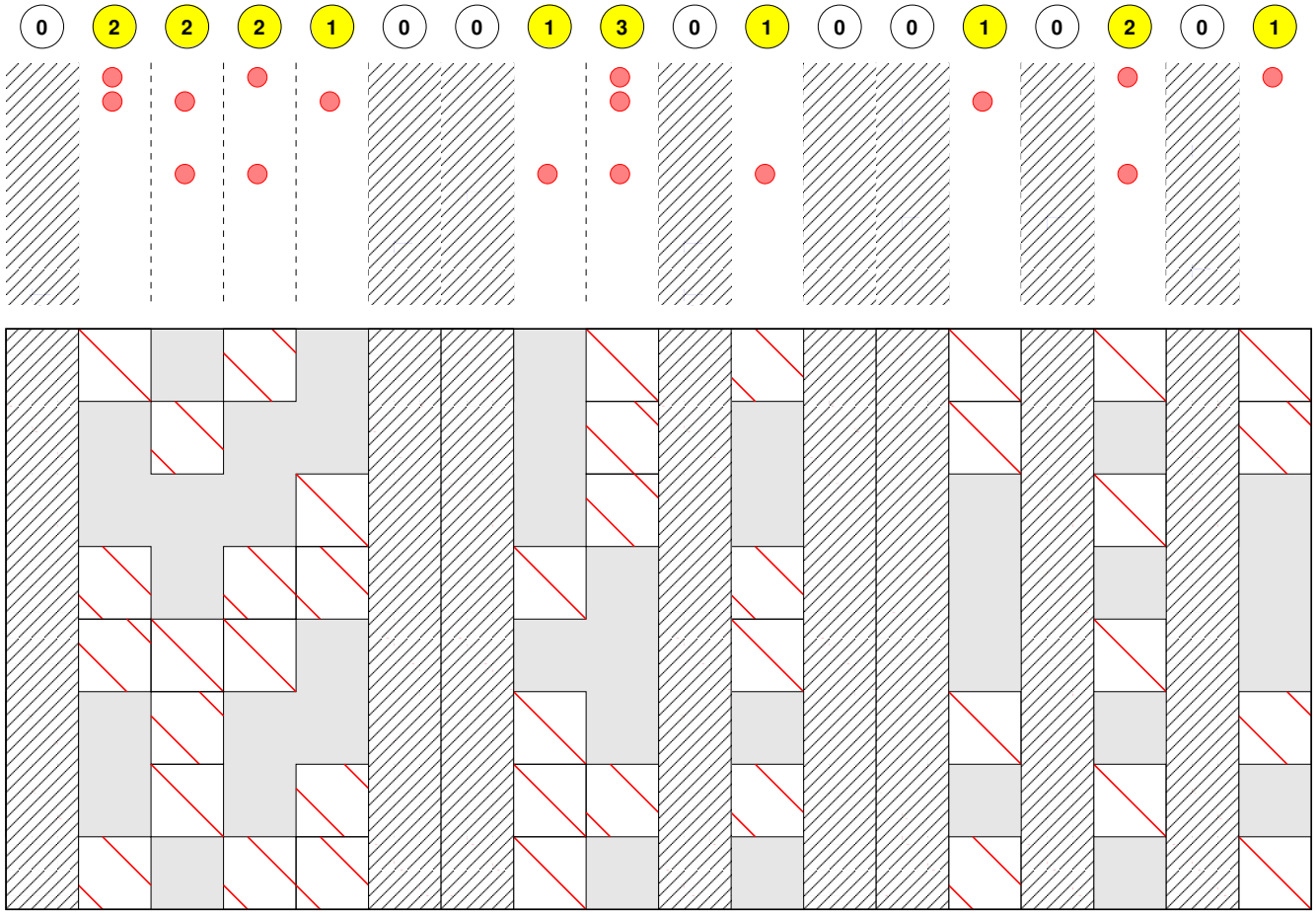
- Build a shortened version of the code, with Nbs=9 block-columns



Shortening STEP-4 : remaining harmfulness in H_{short}

Shortening STEP-4

- **Correction factor** - the ratio between remaining harmfulness weight and total harmfulness weight.

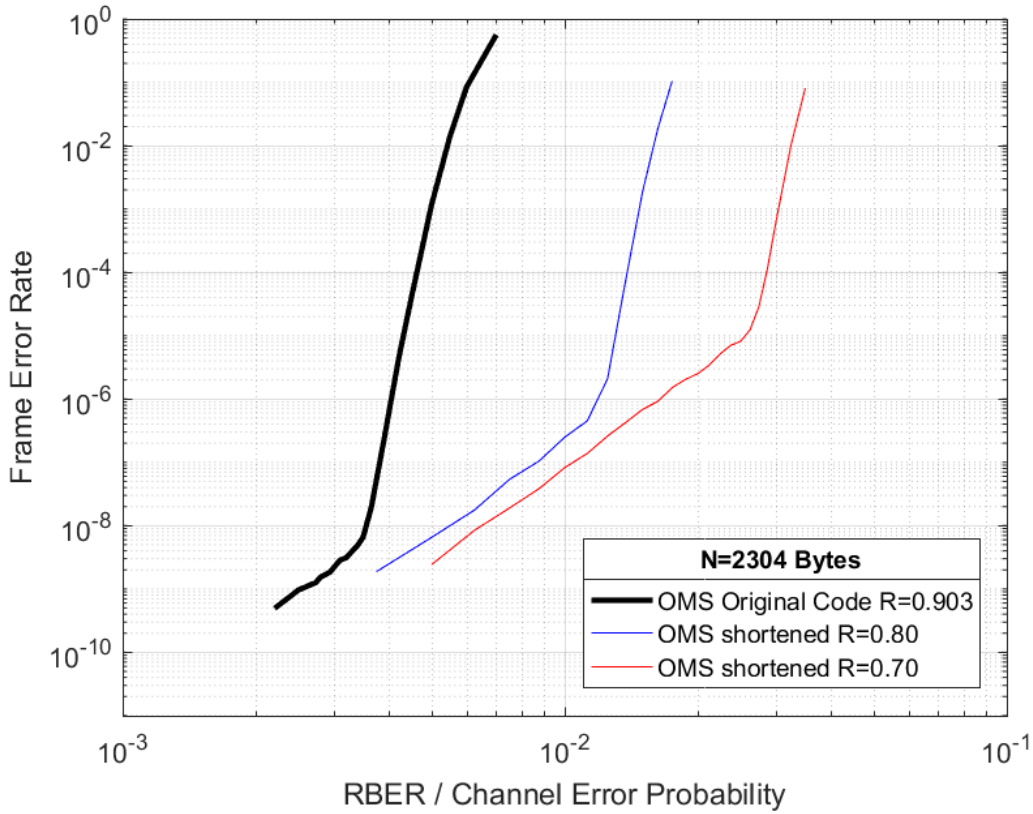


Outline

Results from Direct Simulation of the Shortened Codes

Target QC-LDPC code : N=2kBytes - $R = 0.903$ - $(M_b, N_b) = (56, 576)$ - $L = 32$

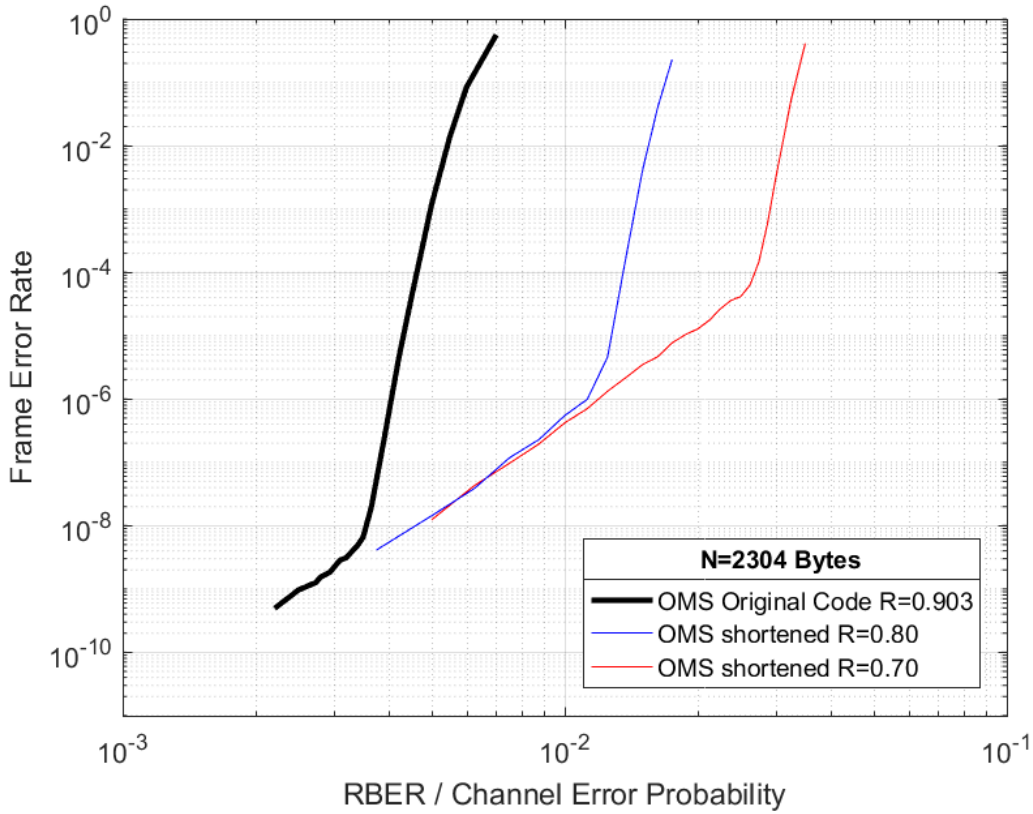
- Shortened Code C_1 : $(M_b, N_b) = (56, 283)$ - $R = 0.802$
- Shortened Code C_2 : $(M_b, N_b) = (56, 182)$ - $R = 0.692$
- **Decoder :** Offset corrected Min-Sum / 4 bits precision / 20 iterations



Results after Correction Factor

Target QC-LDPC code : N=2kBytes - $R = 0.903$ - $(M_b, N_b) = (56, 576)$ - $L = 32$

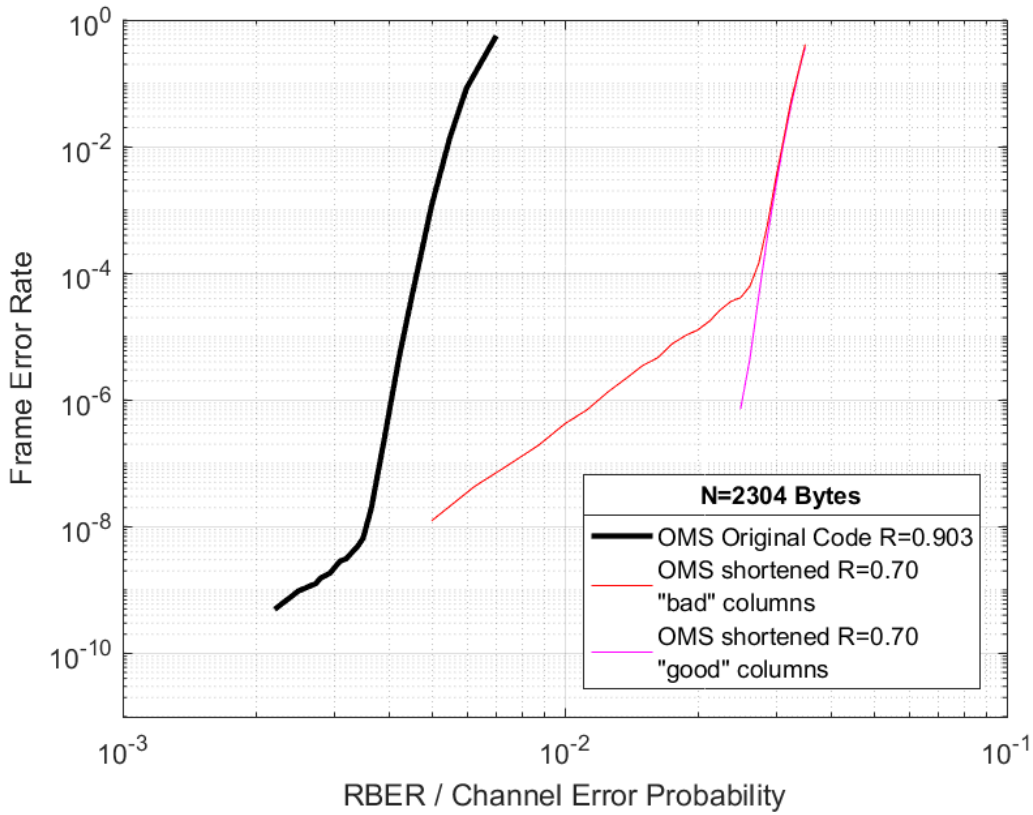
- Shortened Code C_1 : $(M_b, N_b) = (56, 283)$ - $R = 0.802$
- Shortened Code C_2 : $(M_b, N_b) = (56, 182)$ - $R = 0.692$
- **Decoder :** Offset corrected Min-Sum / 4 bits precision / 20 iterations



Selecting the correct block-columns is critical

Target QC-LDPC code : N=2kBytes - $R = 0.903$ - $(M_b, N_b) = (56, 576)$ - $L = 32$

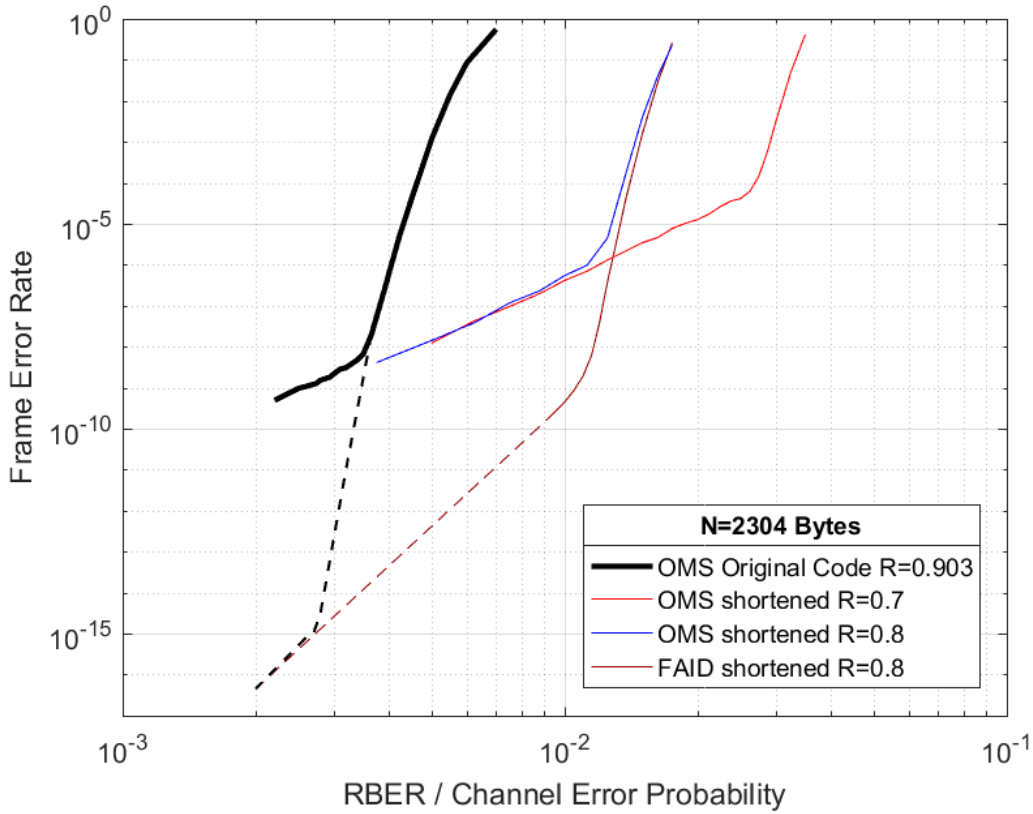
- "Bad" Shortened Code C_2 : $(M_b, N_b) = (56, 182)$ - $R = 0.692$
- "Good" Shortened Code C_2^{bis} : $(M_b, N_b) = (56, 182)$ - $R = 0.692$
- **Decoder :** Offset corrected Min-Sum / 4 bits precision / 20 iterations



Results with FAID decoders

Finite Alphabet Iterative Decoders (FAID) have an inherent error-floor mitigation property

- Our prediction method is valid for ANY decoder.
- The prediction of the Error Floor shows that FAID gains **5 orders of magnitude** compared to Offset-Min-Sum.



Outline

Conclusion

What we have shown

- A computationally efficient method for estimating error floor of QC-LDPC codes over the BSC channel for arbitrary message update rules,
- Applicable to regular and irregular codes, and for more general binary-input memoryless channels
- Graphs of small expansion in the Tanner graph are exhaustively expanded and contracted to obtain subgraphs that are true harmful trapping sets,
- Based on harmfulness of trapping sets, the code is shortened but in a way that it still contains the most harmful trapping sets,
- Allows to predict **very deep error floors**,

Way forward

- Validate our method for irregular codes, and for the BI-AWGN channel,
- Use the new harmfulness characterization to design LDPC codes with very low error-floor.