A Free-Ride Linear Block Codes: Puncturing Optimization and matrix Construction

GiGi Chou

Advisor: Prof. Jiun Hung Yu
Prof. Tofar Chih-Yuan Chang

Transmission and Networking Technologies Laboratory,
Institute of Communications Engineering
National Chiao Tung University, Hsinchu, Taiwan

Outline

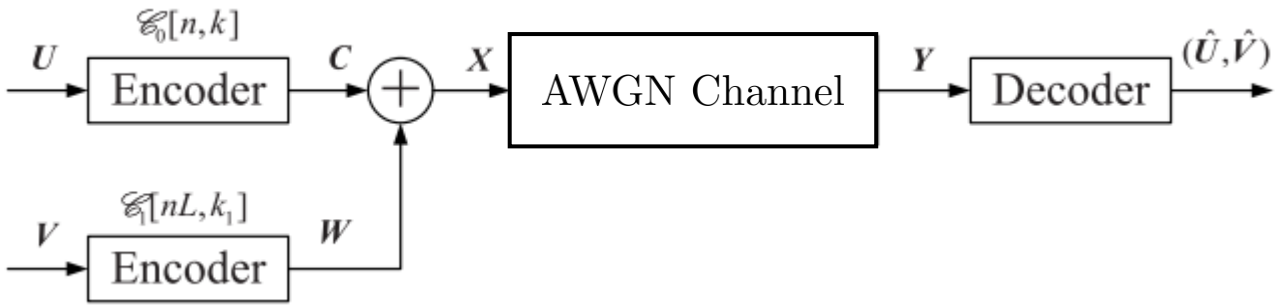
- **Introduction**
- **Free-Ride Concept**
- **Tanner Graph Combination**
- **Puncture Concept**
  - One–Step Recoverable Node Scheme
  - Rate–Compatible Puncturing Scheme
- **Origin Structure**
- **Partial-Extra Structure**
- **Enhanced Structure**
- **Partial-Extra & Enhanced Structure**
- **Reference**

Introduction

- In many communication systems, alongside the primary payload data, a small number of supplementary bits (e.g., device-ID fields or HARQ ACK/NACK flags) must also be transmitted. Traditionally, these extra bits are encoded and sent separately to meet their unique reliability and timing requirements, but at the cost of additional bandwidth and power. The recently proposed Partial-Superposition Incremental-Redundancy HARQ (PS-IR-HARQ) scheme[6] overcomes this overhead by partially superimposing previous code-block information onto the first HARQ retransmission—embedding both payload and control bits without any extra spectral or power resources, thereby boosting throughput and reliability simultaneously.

- This separation is either because the receiver is only interested in the additional bits or because the reliability requirement of the additional bits is higher. One main drawback of the separated transmission paradigm is that it will require additional bandwidth and power.

Free-Ride – Encoding Concept



Encoding Scheme

U : Payload Information Bits
V : Extra Information Bits
C : Payload CodeWord
W : Extra CodeWord
X : Transmit CodeWord
Y : Receive signal
$\hat{\mathbf{U}}$: Estimated Payload Information bits
$\hat{\mathbf{V}}$: Estimated Extra Information bits

Algorithm 1 Encoding scheme

1: Encode the payload information sequence $\mathbf{U} = (u_0, u_1, \ldots, u_{k-1})$ with the encoder of the LDPC code $\xi_0[n,k]$ to obtain the codeword $\mathbf{C} = (c_0, c_1, \ldots, c_{n-1})$.
2: Encode the extra information sequence $\mathbf{V} = (v_0, v_1, \ldots, v_{l-1})$ with the encoder of the LDPC code $\xi_1[n,k]$ to obtain the codeword $\mathbf{W} = (w_0, w_1, \ldots, w_{n-1})$.
3: Superimpose $\mathbf{W}$ onto $\mathbf{C}$ to obtain the transmitted sequence $\mathbf{X} = (\mathbf{C} + \mathbf{W}) \bmod 2$.

Encoding Algorithm

Free-Ride – Pros & Cons

- **Pros**
  - With Free-Ride Decoding, the Extra performance even surpasses simply transmitting the extra bits.
  - Payload performance is almost unchanged.
- **Cons**
  - The Extra decoding employs a maximum likelihood–based decoding approach, so it isn't practical to include too many extra bits in the process.
  - The decoding process follows a strictly serial approach: it must fully complete the extra-bit decoding before it can begin payload decoding.

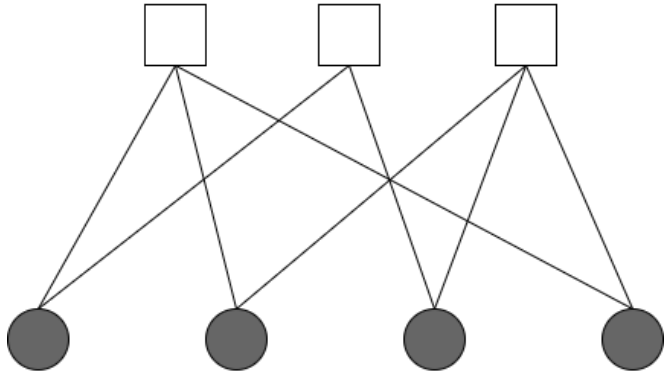
Outline

- **Introduction**
- **Free-Ride Concept**
- **Tanner Graph Combination**
- **Puncture Concept**
  - One–Step Recoverable Node Scheme
  - Rate–Compatible Puncturing Scheme
- **Origin Structure**
- **Partial-Extra Structure**
- **Enhanced Structure**
- **Partial-Extra & Enhanced Structure**
- **Reference**

Tanner Graph Combination

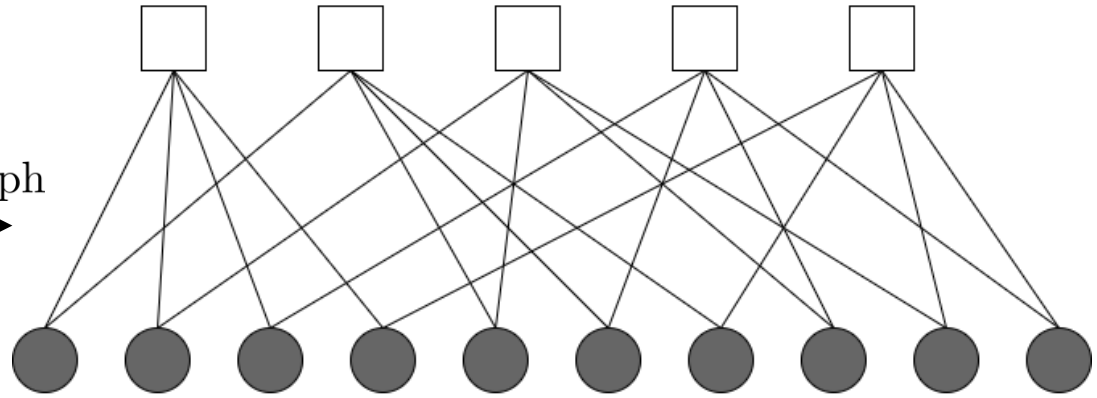
$$H_{extra} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Tanner Graph →

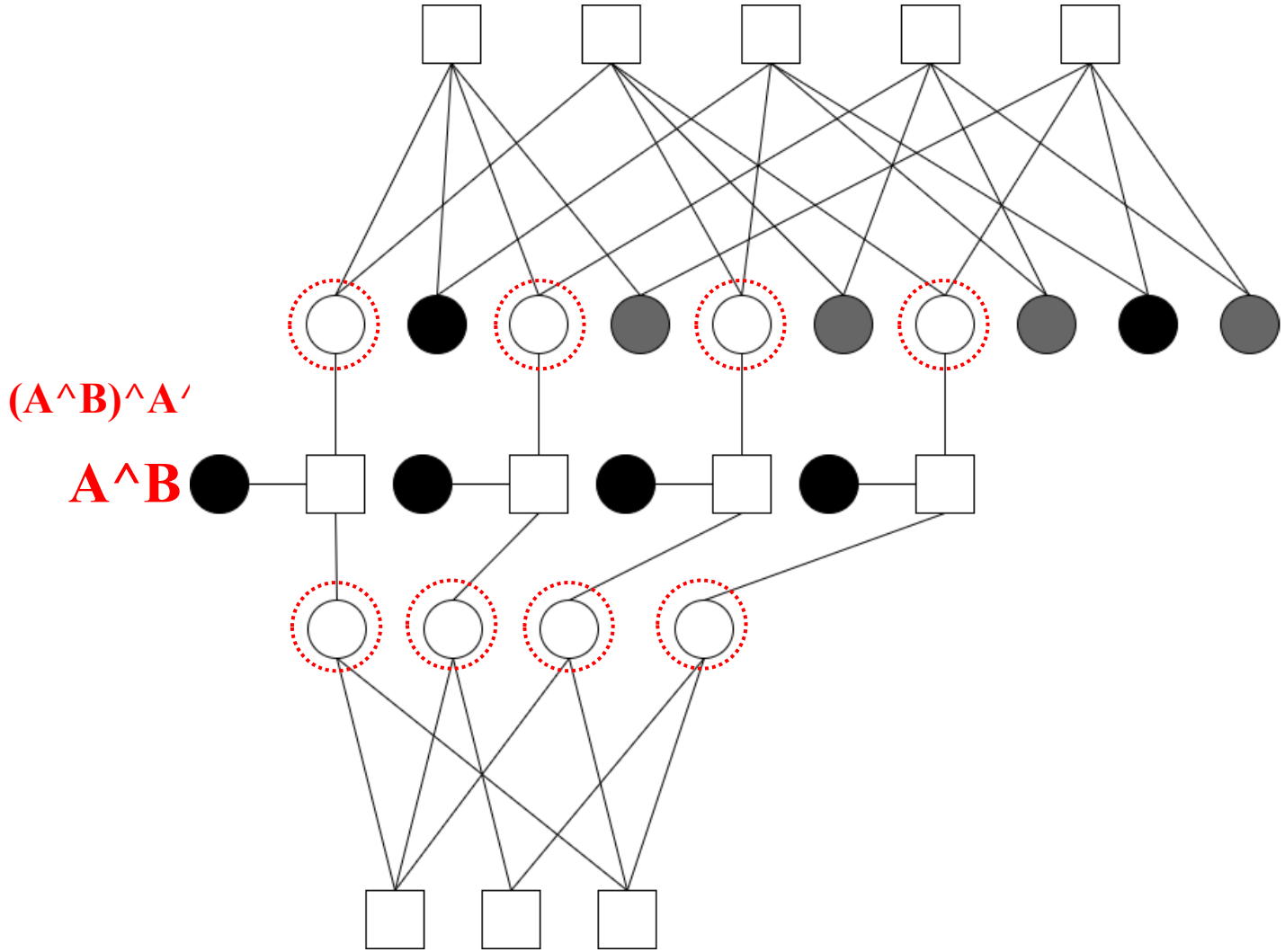


$$H_{payload} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

Tanner Graph →



Tanner Graph Combination



Outline

- **Introduction**
- **Tanner Graph Combination**
- **Puncture Concept**
  - One–Step Recoverable Node Scheme
  - Rate–Compatible Puncturing Scheme
- **Origin Structure**
- **Partial-Extra Structure**
- **Enhanced Structure**
- **Partial-Extra & Enhanced Structure**
- **Reference**

Puncture – Concept

- Puncturing in LDPC (Low-Density Parity-Check) codes refers to the deliberate omission of certain coded bits before transmission to increase the effective code rate. After generating a full LDPC codeword according to a base code rate, the encoder simply removes (punctures) a predefined subset of bits and does not send them over the channel. At the decoder, these punctured positions are treated as erasures; the remaining received bits and the code's sparse parity-check structure are used to infer the missing values during iterative decoding.

- Puncturing lets our adapt the code rate without changing the parity-check matrix but reduces redundancy and degrades error-correction performance. It is typically combined with rate-matching in standards like LTE and 5G NR to optimize throughput and reliability.

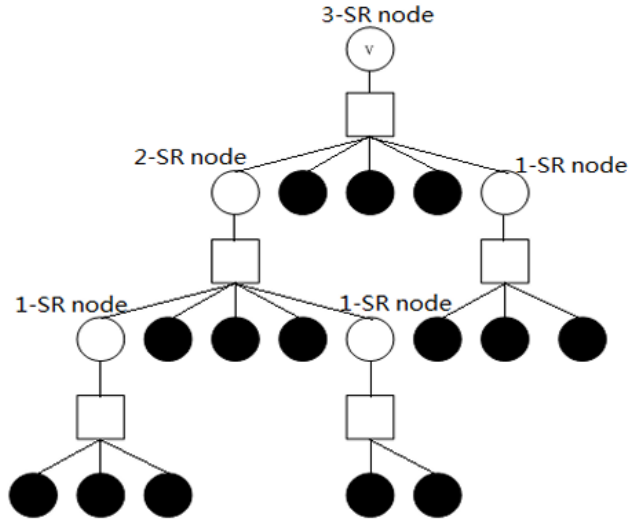
Puncture – k-SR

- A punctured variable node can be reliably recovered if it is connected to a sufficient number of check nodes, each of which is itself connected to reliable variable nodes.
- Generally, a punctured variable node V is called k-step recoverable (k-SR) if V has at least one neighbor C, such that

$$N(C) \setminus \{V\}$$

contains at least one $(k-1)$-SR node and all other nodes are m-SR for some $0 \leq m \leq k-1$. The k-SR node is recovered after the k-th iteration.

Recovery Tree



- filled circles: unpunctured nodes
- unfilled circles: punctured nodes

Recovery Error Probability

Let $G_{k>0}$ be the set of variable nodes at depth $\leq k$ in the recovery tree of v. Over a BEC with erasure probability ζ, the recovery error probability of a variable node v is

$$P_e(v) = \tfrac{1}{2}\big(1 - \Phi(v, \zeta)\big),$$

where

$$\Phi(v, \zeta) = (1 - \zeta)^{S(v)},$$

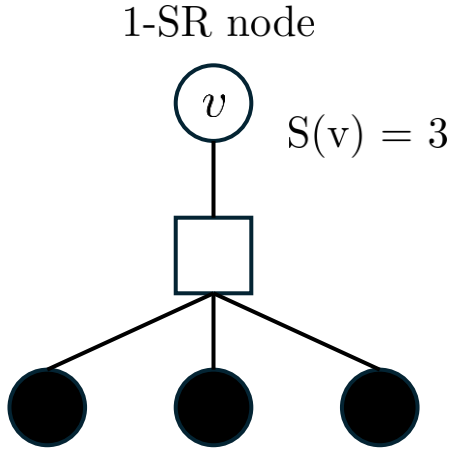
and $S(v)$ is the number of surviving neighbors of v in its recovery tree.

Base Case ($k = 1$)

For $k = 1$, all variable nodes in the recovery tree of v lie in G_0. Each surviving check node has degree d_c, so

$$P_e(v) = \tfrac{1}{2}\big(1 - (1 - \zeta)^{d_c - 1}\big) = \tfrac{1}{2}\big(1 - (1 - \zeta)^{S(v)}\big) = \tfrac{1}{2}\big(1 - \Phi(v, \zeta)\big).$$

Here d_c is the degree of the surviving check node of v.



- filled circles: unpunctured nodes
- unfilled circles: punctured nodes

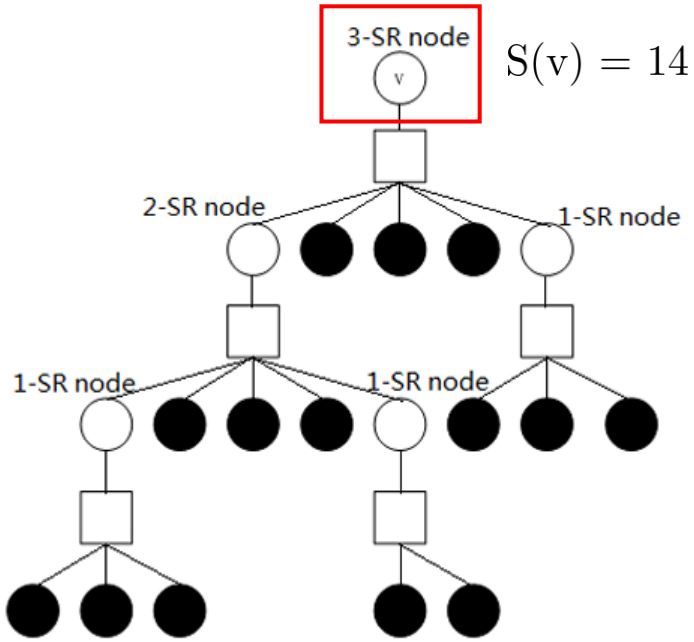
Induction Step $(k > 1)$

For any $v \in G_{k+1}$ with surviving check-node degree d_c and neighbors $\gamma_1, \ldots, \gamma_{d_c-1} \in G_h$ $(1 \le h \le k)$, define

$$S(v) = \sum_{j=1}^{d_c-1} S(\gamma_j), \quad \Phi(x, \zeta) = (1-\zeta)^{S(x)}.$$

Then

$$\begin{aligned} P_e(v) &= \tfrac{1}{2}\big(1 - \prod_{j=1}^{d_c-1} \Phi(\gamma_j, \zeta)\big) \\ &= \tfrac{1}{2}\big(1 - (1-\zeta)^{\sum_{j=1}^{d_c-1} S(\gamma_j)}\big) \\ &= \tfrac{1}{2}\big(1 - (1-\zeta)^{S(v)}\big) \\ &= \tfrac{1}{2}\big(1 - \Phi(v, \zeta)\big). \end{aligned}$$



- filled circles: unpunctured nodes
- unfilled circles: punctured nodes

Puncture – k-SR Algorithm[4]

Algorithm 4 Puncturing Node Grouping

1: **Input:** Parity-check matrix $H \in \{0,1\}^{m\times n}$

2: **[Initialize]**

$$k \leftarrow 1, \quad R_0 \leftarrow \emptyset,\ R_1 \leftarrow \emptyset,\ R_\infty \leftarrow \{1,2,\ldots,m\},$$

$$\Gamma^\rho \leftarrow \{\gamma : H_{\rho,\gamma} = 1, and\ 1 \le \gamma \le n\}, \quad \Lambda^\gamma \leftarrow \{\rho : H_{\rho,\gamma} = 1, and\ 1 \le \rho \le m\},$$

$$G_0 \leftarrow \emptyset,\ G_1 \leftarrow \emptyset,\ G_\infty \leftarrow \{1,2,\ldots,n\},\ S(j) \leftarrow 0\ for\ all\ 1 \le j \le n$$

3: **[Group Column Indices]** For each $\rho \in R_\infty$, set $\vartheta_\infty^\rho = \Gamma^\rho \cap G_\infty$

4: **[Find Candidate Rows]** Make a subset of R_∞ (call it Ω) such that

$$\forall \omega \in \Omega, \quad \left|\vartheta_\infty^\omega\right| = rw_{\text{eff}}^{\min} \le \left|\vartheta_\infty^\rho\right| = rw_{\text{eff}}(\rho, G_\infty) \quad \forall \rho \in R_\infty.$$

5: **[Group Row Indices]** Make a set C_∞^ω such that

$$C_\infty^\gamma = \Lambda^\gamma \cap R_\infty, \quad \forall \gamma \in \vartheta_\infty^\omega,\ \omega \in \Omega.$$

6: **[Find Best Rows]** Find a subset of Ω (call it Ω^*) such that

$$\forall \omega^* \in \Omega^*, \ \exists c \in \vartheta_\infty^{\omega^*} \ \ such\ that \ \ |C_\infty^c| = cw_{\text{eff}}^{\min} \le |C_\infty^\gamma| = cw_{\text{eff}}(\gamma, R_\infty)$$

$$for\ any\ \omega \in \Omega\ and\ \gamma \in \vartheta_\infty^\omega.$$

7: **[Make a Set of Ordered Pairs]** Pick a column index

$$c^* \in G_\infty^{\omega^*} \quad with \quad cw_{\text{eff}}^{\min}$$

randomly , when there are more than one column index with $cw_{\text{eff}}^{\min}$ .Then, we will have an ordered pair

$$\mathcal{O} = \{(\omega_1^*, c_1^*), (\omega_2^*, c_2^*), \ldots, (\omega_p^*, c_p^*)\},$$

where each ω_j^* and c_j^* is a row- and column-index pair satisfying

$$rw_{\text{eff}}^{\min} \quad and \quad cw_{\text{eff}}^{\min}, \quad 1 \le j \le |\mathcal{O}| = p.$$

8: **[Find The Best Pair]** Pick a pair (ω^*, c^*) from $\mathcal{O}$ such that

$$\mathcal{W}(\omega^*) \le \mathcal{W}(\omega_j^*), \quad 1 \le j \le p, \quad \mathcal{W}(\omega_j^*) = \sum_{\gamma \in \Gamma^{\omega_j^*}} S(\gamma).$$

If there are more than one pair satisfying the inequality, pick one randomly.

9: **[Update]**

$$G_k \leftarrow G_k \cup \{c^*\}, \quad G_0 \leftarrow G_0 \cup \left(\vartheta_\infty^{\omega^*} \setminus \{c^*\}\right),$$

$$G_\infty \leftarrow G_\infty \setminus \vartheta_\infty^{\omega^*}, \quad R_k \leftarrow R_k \cup \{\omega^*\},$$

$$R_0 \leftarrow R_0 \cup \left(C_\infty^{c^*} \setminus \{\omega^*\}\right), \quad R_\infty \leftarrow R_\infty \setminus C_\infty^{c^*},$$

$$S(\gamma) = 1 \quad \forall \gamma \in \vartheta_\infty^{\omega^*} \setminus \{c^*\}, \quad S(c^*) = \sum_{\gamma \in \Gamma^{\omega^*} \setminus \{c^*\}} S(\gamma).$$

10: **[Check Stop Condition]** If G_∞ is an empty set, then **STOP**.

11: **[Decision]** If $R_\infty \neq \emptyset$, go to 3.

12: **[No More Undetermined Rows]**

$$R_\infty = \{\rho : \rho \in R_0\ and\ rw_{\text{eff}}(\rho, G_\infty) > 0\}.$$

13: $k \leftarrow k+1$, and go to 3.

Find most 1-SR → 2-SR → ... → k-SR

Puncture – RC Scheme Algorithm[3]

Algorithm 5 Rate-Compatible Puncturing Scheme

1: **Inputs:** Parity-check matrix (Tanner graph) of $C(r_0)$; sequence of rates $r_1, \ldots, r_m$; $\eta_{\max}$; $\ell_{\max}$. Set k =1.

2: Find the set UPset_k, and calculate P_k(node number). Also, for all $c \in V_c$, calculate $F(c)$ and $U(c)$, and set counter $\leftarrow 0$.

3: Let

$$\Psi = \{ c^* \in V_c : F(c^*) \leq F(c) \; \forall \, c \in V_c \},$$

then set

$$\text{CandidateCheck} = \{ c^* \in \Psi : U(c^*) \leq U(c) \; \forall \, c \in \Psi \}.$$

4: Let

$$\text{CandidateVar} = \{ v : v \in N(c) \; \forall \, c \in \text{CandidateCheck} \} \; \cap \; \text{UPset}_k.$$

5: Let

$$\Gamma = \{ v^* \in \text{CandidateVar} : H(v^*) \leq H(v) \; \forall \, v \in \text{CandidateVar} \}.$$

If $|\Gamma| = 1$, $v_p = \Gamma$ and go to Step 10.

6: Let

$$\Delta = \{ v^* \in \Gamma : d(v^*) \leq d(v) \; \forall \, v \in \Gamma \}.$$

If $|\Delta| = 1$, $v_p = \Gamma$ and go to Step 10.

7: Let

$$\Theta = \{ v^* \in \Delta : K(v^*) \leq K(v) \; \forall \, v \in \Delta \}.$$

If $|\Theta| = 1$, $v_p = \Gamma$ and go to Step 10.

8: Set $\ell \leftarrow 4$ and $\Lambda \leftarrow \Theta$.

(a) $\Lambda \leftarrow \{ v^* \in \Lambda : S^{v^*}_{\eta_{\max}}(\ell) \leq S^{v}_{\eta_{\max}}(\ell) \; \forall \, v \in \Lambda \}$.

(b) **If** $|\Lambda| = 1$ **then** $v_p = \Gamma$ and go to Step 10.

(c) **If** $\ell < \ell_{\max}$ **then** $\ell \leftarrow \ell + 2$ and **go to** Step 8(a).

9: Choose v_p at random from Λ.

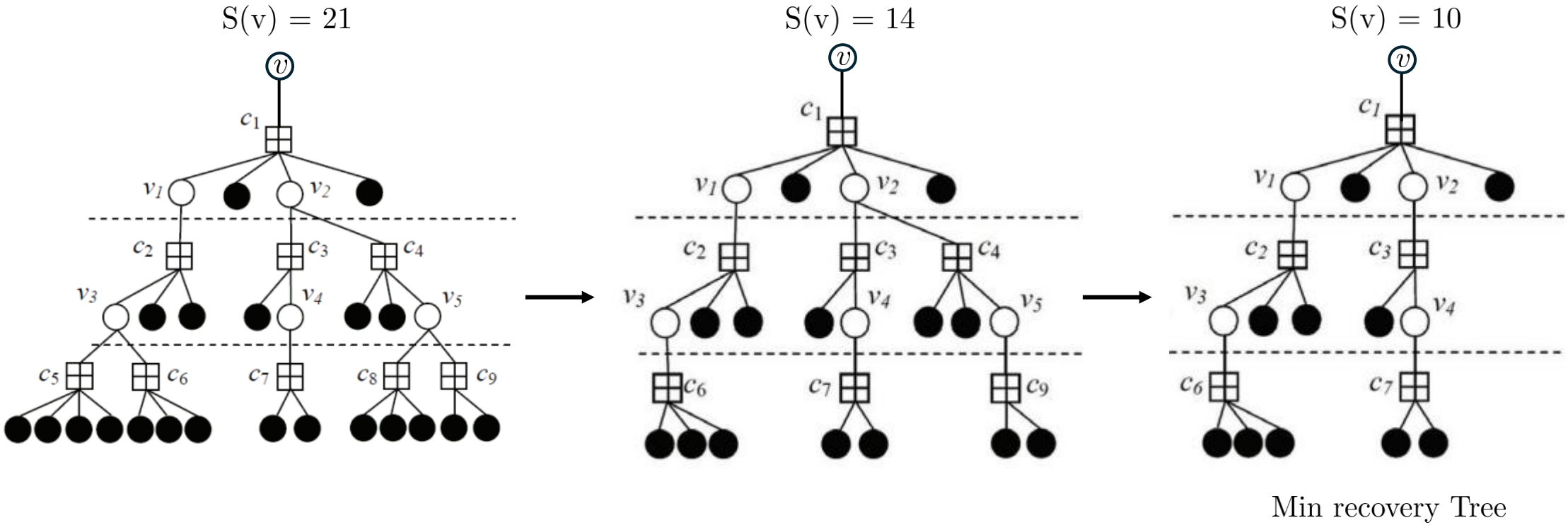
10: Puncture v_p, set

$$\text{counter} \leftarrow \text{counter} + 1, \quad \text{UPset}_k \leftarrow \text{UPset}_k \setminus \{v_p\}, \quad \text{Pset}_k \leftarrow \text{Pset}_k \cup \{v_p\}.$$

11: **if** counter $< P_k$, $\forall \, c \in N(v_p)$, set $F(c) \leftarrow F(c)+1$, and $\forall \, c \in \{ c : v_p \in T(c) \}$, update $U(c)$. Also, $\forall \, v \in B(v_p)$, set $H(v) \leftarrow H(v) + 1$, and Go to Step 3. Otherwise, if $k < m$, then $k \leftarrow k + 1$, and go to Step 2.

12: Stop

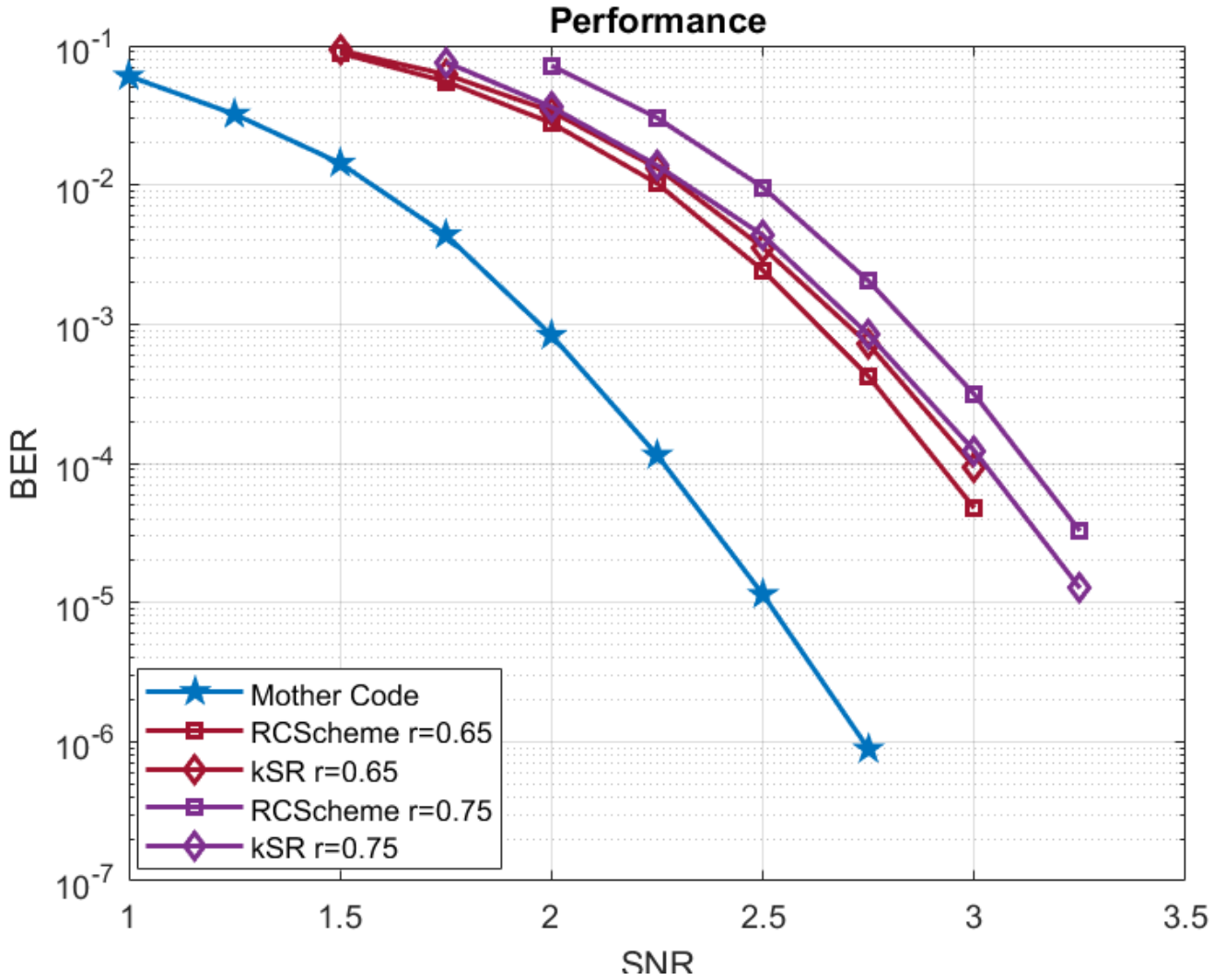
Puncture – RC Scheme Algorithm



Min recovery Tree

Puncture – simulation

Compare different puncturing algorithm :

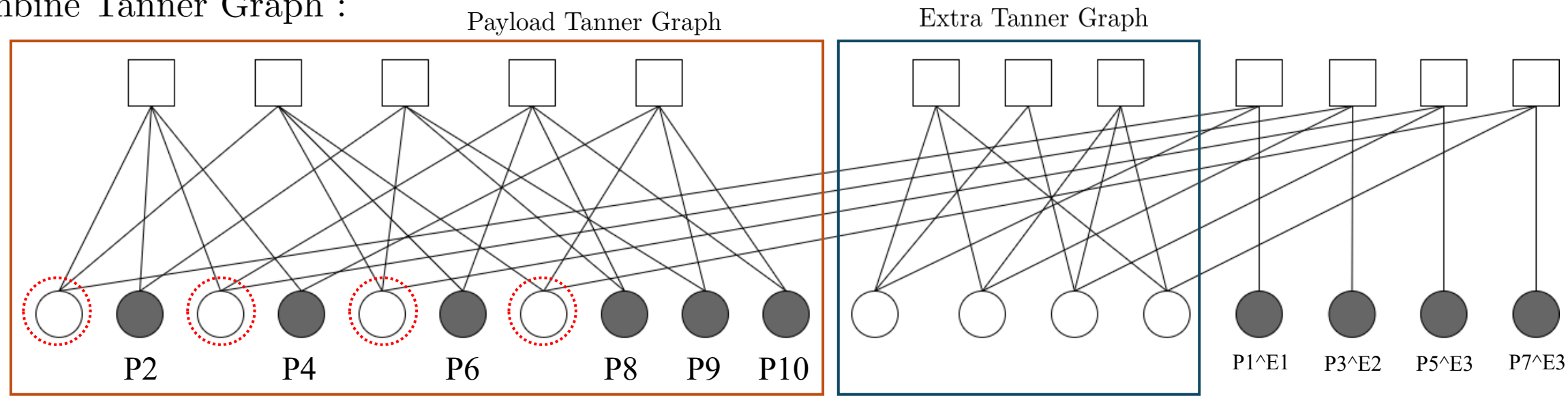


Outline

- **Introduction**
- **Free-Ride Concept**
- **Tanner Graph Combination**
- **Puncture Concept**
  - One–Step Recoverable Node Scheme
  - Rate–Compatible Puncturing Scheme
- **Origin Structure**
- **Partial-Extra Structure**
- **Enhanced Structure**
- **Partial-Extra & Enhanced Structure**
- **Reference**

Origin Structure

Combine Tanner Graph :



- filled circles: unpunctured variable nodes
- unfilled circles: punctured variable nodes

$$\boldsymbol{H}_{Combine}: \begin{bmatrix} \boldsymbol{H}_{\text{payload}} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{H}_{\text{extra}} & \boldsymbol{0} \\ \boldsymbol{M}_{\text{Puncpos}} & \boldsymbol{I} & \boldsymbol{I} \end{bmatrix}$$

$M_{puncpos}$ each row degree is 1.

Origin Structure - Algorithm

Algorithm 1 Original Combined Tanner Graph Construction

Require: Parity-check matrix $\mathbf{H}$, index set $\mathcal{I}$ of payload VNs

Ensure: Updated parity-check matrix $\mathbf{H}'$ representing the combined Tanner graph

$\mathbf{H}' \leftarrow \mathbf{H}$

for each $i \in \mathcal{I}$ **do**

  Let p_i be the payload VN and e_i the corresponding extra VN

  Create a new variable node z_i that carries $p_i \oplus e_i$

  Create a new check node c_i enforcing

$$p_i \oplus e_i \oplus z_i = 0$$

  Insert one row for c_i and one column for z_i into H'

  Connect c_i to p_i, e_i, and z_i in the Tanner graph

end for

return $\mathbf{H}'$

Origin Structure – Code-Rate Fix

- In our structured Tanner graph, we fix the code-rate r as

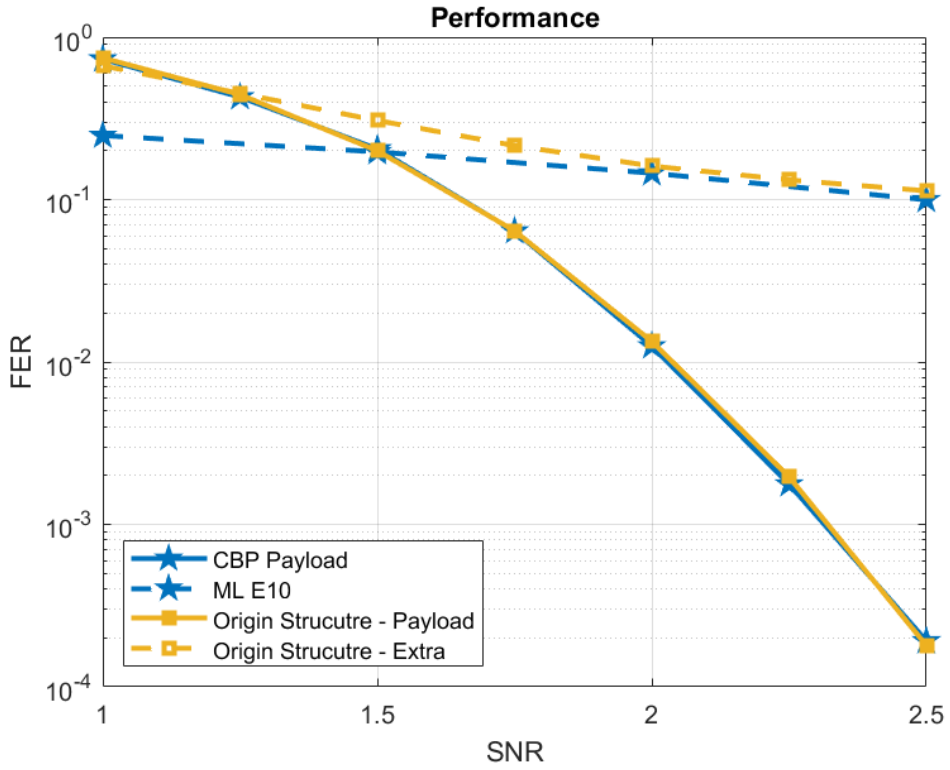
$$r = \frac{\text{length(payload info)} + \text{length(extra info)}}{\text{length(payload encode)}}$$

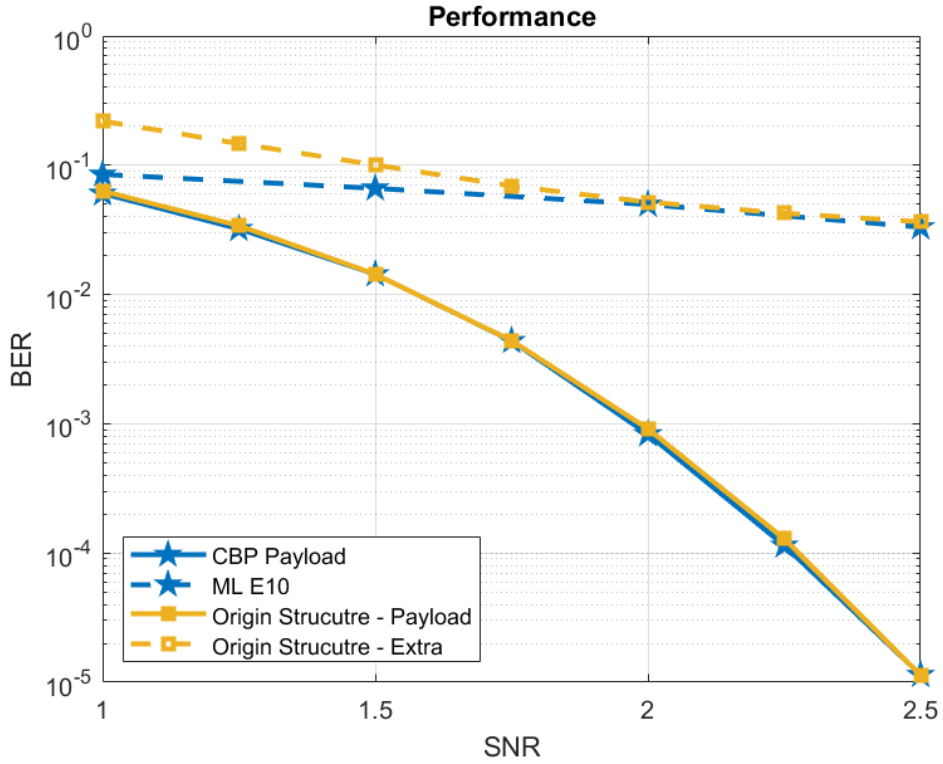
Code Information :

Code	Codeword Length	Information Length	Code Rate	Max Col Degree	Max Row Degree	Girth
PEG-P1	**1008**	**504**	**0.5**	**3**	**8**	**8**
LDPC-E1	**10**	**5**	**0.5**	**2**	**4**	**6**
BCH-E2	**15**	**7**	**0.47**	**4**	**4**	**6**

Origin Structure – Simulation

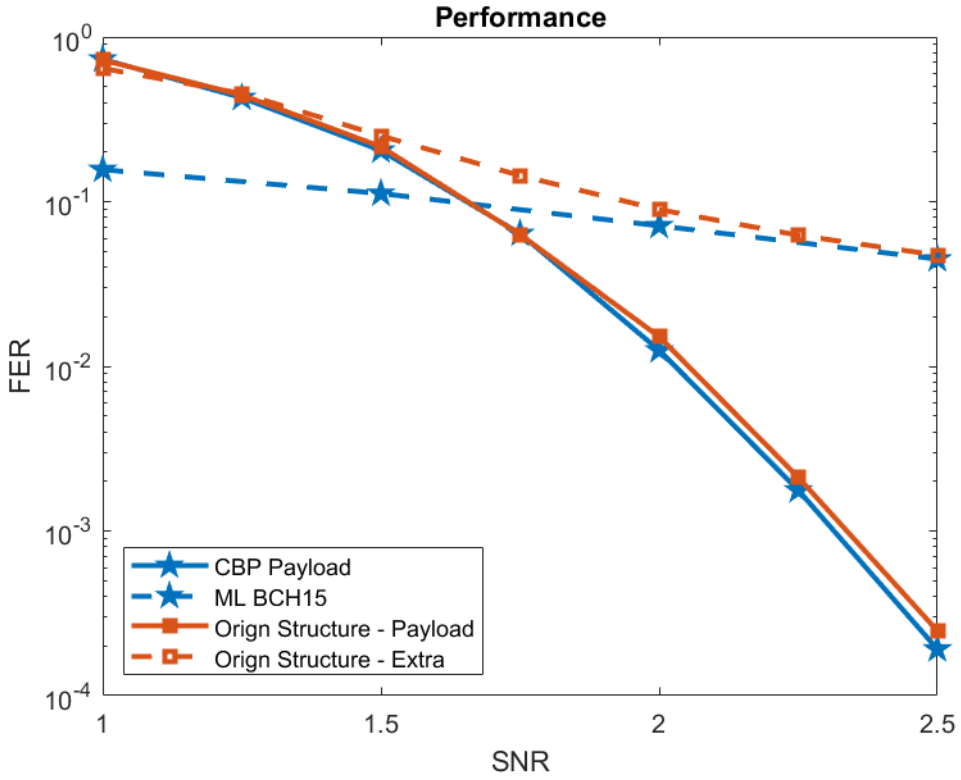
- Payload : PEG-P1
- Extra : LDPC-E1

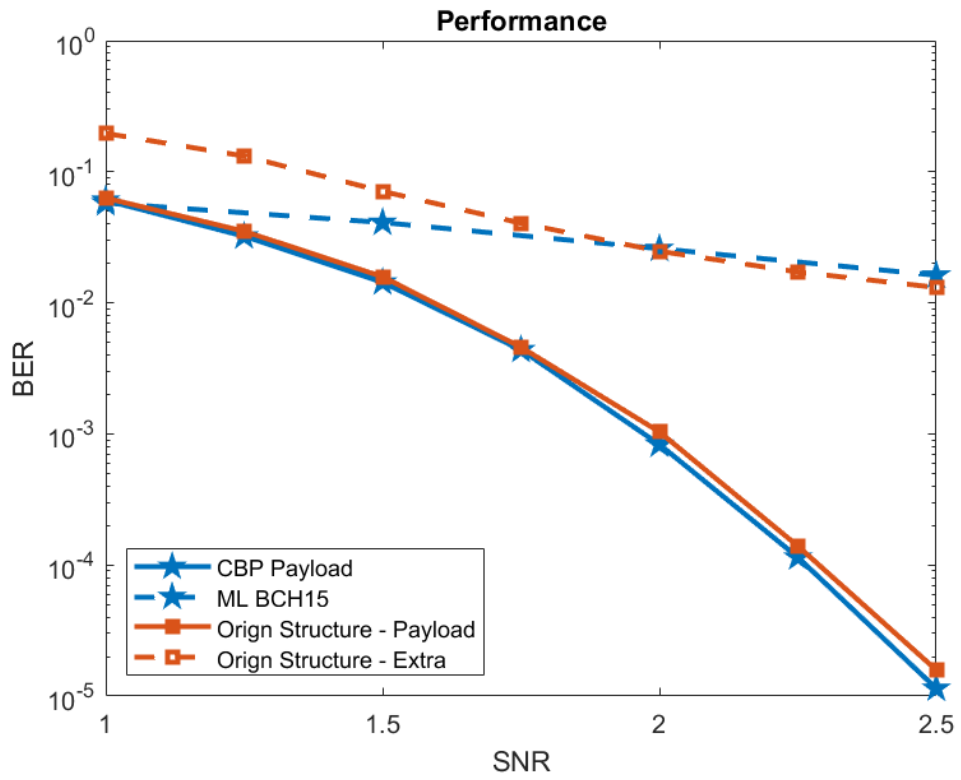




Origin Structure – Simulation

- Payload : PEG-P1
- Extra : BCH-E2





Origin Structure – minimum distance

Review:

[**Minimum Distance**] The minimum weight (or minimum distance) of an (n, k) linear block code $\mathcal{C}$ with parity check matrix $\mathbf{H}$ is equal to the smallest number of columns of $\mathbf{H}$ whose vector sum is the zero vector.

Assume $\mathbf{H}_{\text{payload}} \in \mathbb{F}_2^{r_1 \times c_1}$ as the payload parity-check matrix, and $\mathbf{H}_{\text{extra}} \in \mathbb{F}_2^{r_2 \times c_2}$ as the extra parity-check matrix.

Let

$$d_{\min}(\mathbf{H}_{\text{payload}}) = M, \quad d_{\min}(\mathbf{H}_{\text{payload}}) = N,$$

To determine the minimum distance of $\mathbf{H}_{\text{origin}}$ using the definition that it corresponds to the smallest number of columns whose vector sum is zero. There are two distinct possibilities:

- First, the zero-sum condition may be satisfied by a set of M columns from the $\mathbf{H}_{\text{payload}}$ portion.
- Second, the zero-sum condition may also arise from a set of columns originating from the $\mathbf{H}_{\text{extra}}$ portion.

Origin Structure – minimum distance

Case 1: Zero-sum Condition from the $\mathbf{H}_{\text{payload}}$ Block

Define the set of weight-M codewords of $\mathbf{H}_{\text{payload}}$ by

$$\mathcal{C}_{\text{payload}} = \left\{ \vec{c} \in \{0,1\}^{c_1} \;\middle|\; \vec{c}\mathbf{H}_{\text{payload}}^{\mathbf{T}} = \vec{0},\ \text{wt}(\vec{c}) = M \right\},$$

where $\text{wt}(\vec{c})$ is the number of ones in binary vector $\vec{c}$. Next, let

$$S = \left\{ j \in \{1, 2, \ldots, c_1\} \;\middle|\; \exists\, \vec{c} \in \mathcal{C}_{\text{payload}} :\ c_j = 1 \right\},$$

i.e. the set of column indices in which some weight-M payload codeword has a 1.

Define

$$\mathcal{P} = \left\{ j : \text{column } j \text{ of } \mathbf{M}_{\text{puncpos}} \text{ has degree } 1 \right\},$$

and set $k = |S \cap \mathcal{P}|$.

Origin Structure – minimum distance

Observe that in $\mathbf{H}_{\text{origin}}$ these columns appear in the sub-block

$$\begin{pmatrix} \mathbf{H}_{\text{payload}} & \mathbf{O}_{\mathbf{r_1} \times \mathbf{c_2}} \\ \mathbf{M}_{\text{puncpos}} & \mathbf{I}_{\mathbf{c_2}} \end{pmatrix},$$

where $\mathbf{M}_{\text{puncpos}} \in \mathbb{F}_2^{c_2 \times c_1}$ has every row of degree 1 and every column of degree at most 1, and $\mathbf{I}_{\mathbf{c_2}}$ is the $c_2 \times c_2$ identity matrix.

Selecting any $j \in S$ from the payload block flips the j-th row of the lower identity block from 0 to 1. If $j \in \mathcal{P}$, then to restore that row's sum to zero we must additionally select column j of the identity block.

Since each flipped row can only be canceled by its matching puncture column, any weight-M payload configuration requires

$$M + k$$

total columns to achieve a zero-sum in this subblock. Hence

$$d_{\min}^{(1)} = M + k.$$

Origin Structure – minimum distance

Case 2: Zero-sum Condition from the $\mathbf{H}_{\text{extra}}$ Block

Define the set of weight-N codewords of $\mathbf{H}_{\text{extra}}$ by

$$\mathcal{C}_{\text{extra}} = \left\{ \vec{c} \in \{0,1\}^{c_2} \;\middle|\; \vec{c}\,\mathbf{H}_{\text{extra}}^{\mathbf{T}} = \vec{0}, \quad \text{wt}(\vec{c}) = N \right\},$$

where $\text{wt}(\vec{c})$ is the number of ones in the binary vector $\vec{c}$. Next, let

$$T = \left\{ j \in \{1, 2, \ldots, c_2\} \;\middle|\; \exists\, \vec{c} \in \mathcal{C}_{\text{extra}} :\; c_j = 1 \right\},$$

i.e. the set of column indices in which some weight-N codeword has a 1.

Observe that in $\mathbf{H}_{\text{origin}}$ these columns appear in the block

$$\begin{pmatrix} \mathbf{H}_{\text{extra}} & \mathbf{O}_{\mathbf{r_2} \times \mathbf{c_2}} \\ \mathbf{I}_{\mathbf{c_2}} & \mathbf{I}_{\mathbf{c_2}} \end{pmatrix},$$

where $\mathbf{I}_{\mathbf{c_2}}$ is the $c_2 \times c_2$ identity matrix.

Origin Structure – minimum distance

Selecting any column $j \in T$ from the $\mathbf{H}_{\text{extra}}$ portion flips the j-th row of the lower identity block from 0 to 1. To restore that col-sum to zero, we must also select the corresponding column j from the right-hand identity block. Since each identity row has degree 1 and they are independent, no combination of other columns can cancel it that each flipped row demands its own matching identity column.

Therefore, for each weight-N codeword one must pick its N columns plus N additional identity columns, for a total of $2N$. Hence in this case

$$d_{\min}^{(2)} = 2N.$$

Since the minimum distance may arise from either of the two cases discussed above, we conclude that the overall minimum distance must be at least the smaller of the two. Hence,

$$d_{\min}(\mathbf{H}_{\text{origin}}) \geq \min(M + k,\ 2N) = \min(d_{min}^{(1)}, d_{min}^{(2)}).$$

Origin Structure – Observation

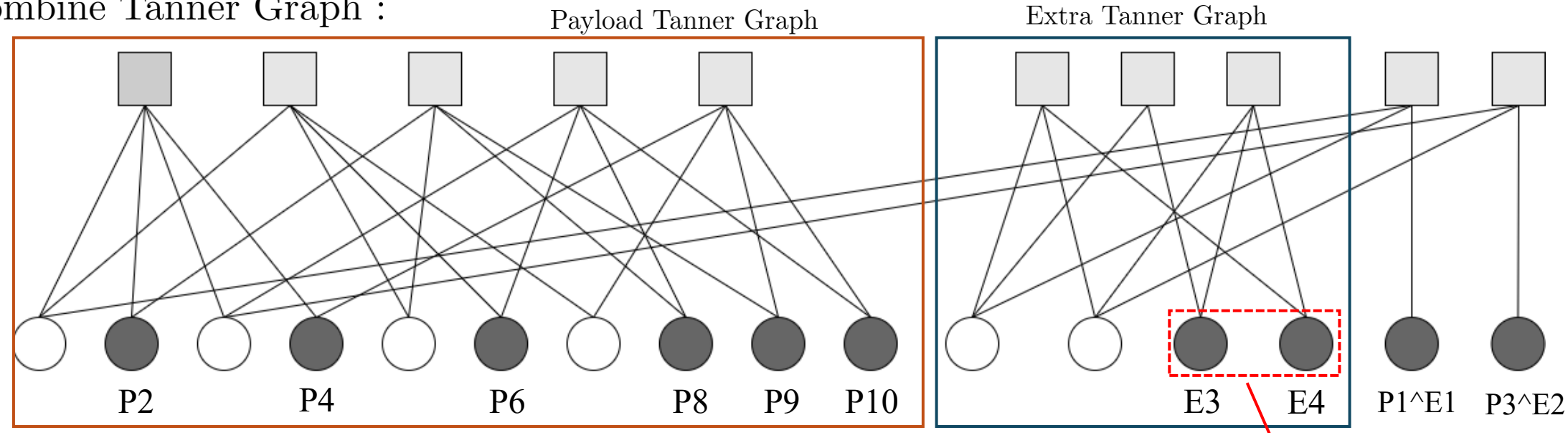
- Merging the Tanner graph allows Extra and Payload to be decoded at the same time, and the payload's performance is barely affected.
- This method of combining Tanner graphs allows Payload and Extra to be decoded in parallel.

Outline

- **Introduction**
- **Free-Ride Concept**
- **Tanner Graph Combination**
- **Puncture Concept**
  - One–Step Recoverable Node Scheme
  - Rate–Compatible Puncturing Scheme
- **Origin Structure**
- **Partial-Extra Structure**
- **Enhanced Structure**
- **Partial-Extra & Enhanced Structure**
- **Reference**

Partial-Extra Structure

Combine Tanner Graph :



- filled circles: unpunctured variable nodes
- unfilled circles: punctured variable nodes

Transmit parital extra bits(random choose)

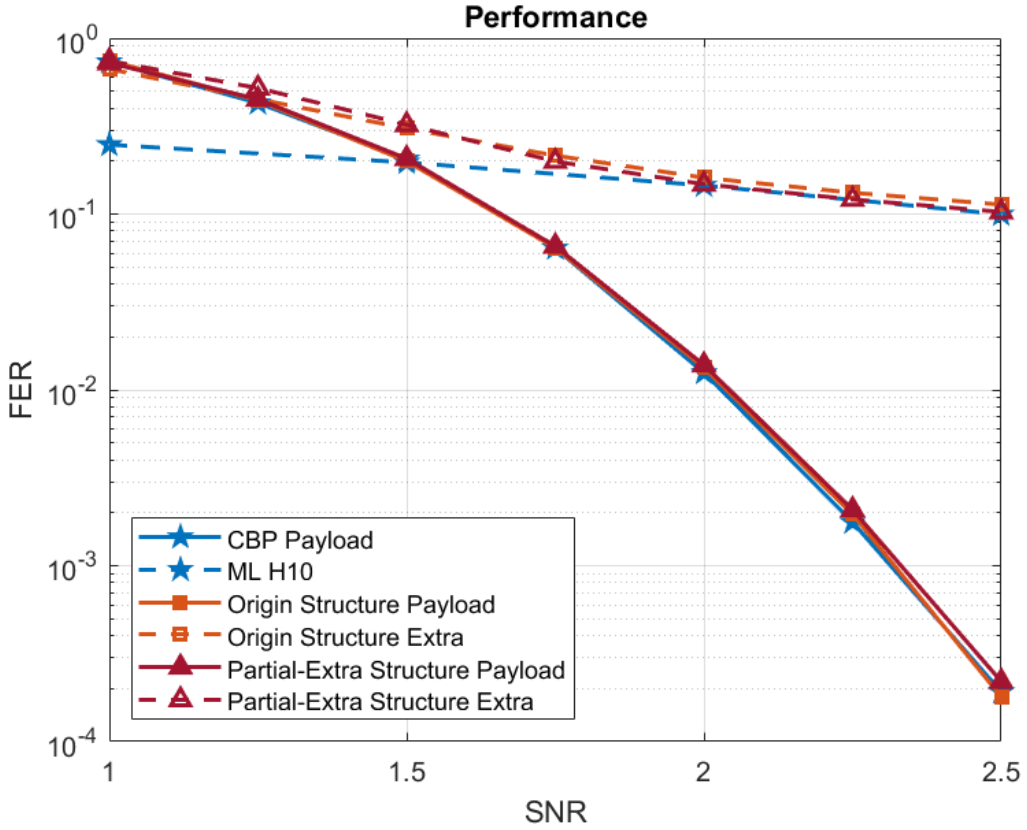
$$\boldsymbol{H}_{Combine}: \begin{bmatrix} \boldsymbol{H}_{\text{payload}} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{H}_{\text{extra}} & \mathbf{0} \\ \boldsymbol{M}_{\text{Puncpos}} & \boldsymbol{M}_{\text{extrapunc}} & \boldsymbol{I} \end{bmatrix}$$

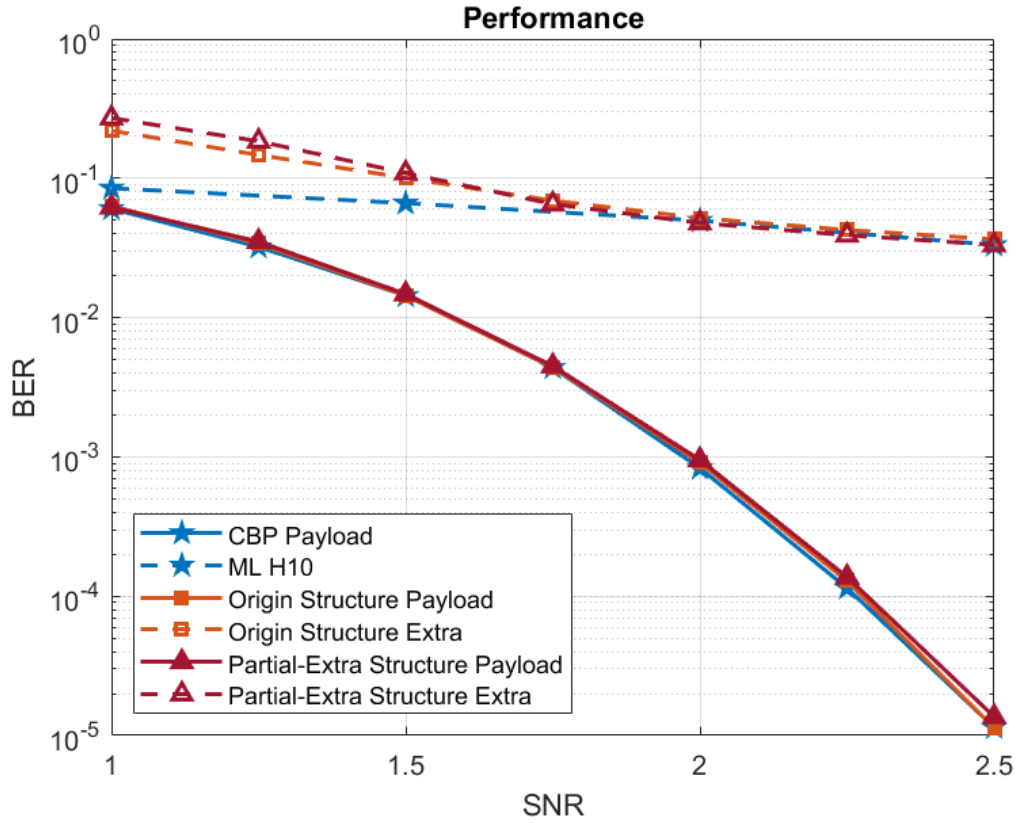
$M_{Puncpos}$ each row degree is 1.

$M_{extrapunc}$ each row degree is 1.

Partial-Extra Structure – Simulation

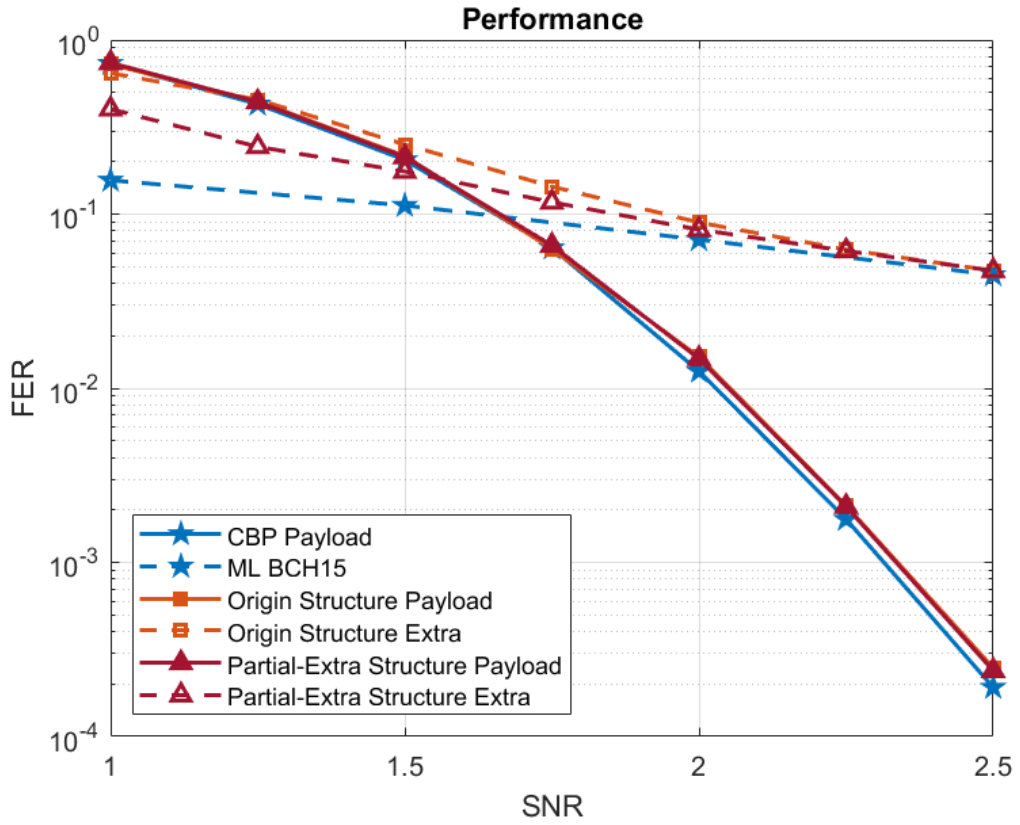
- Payload : PEG-P1
- Extra : LDPC-E1

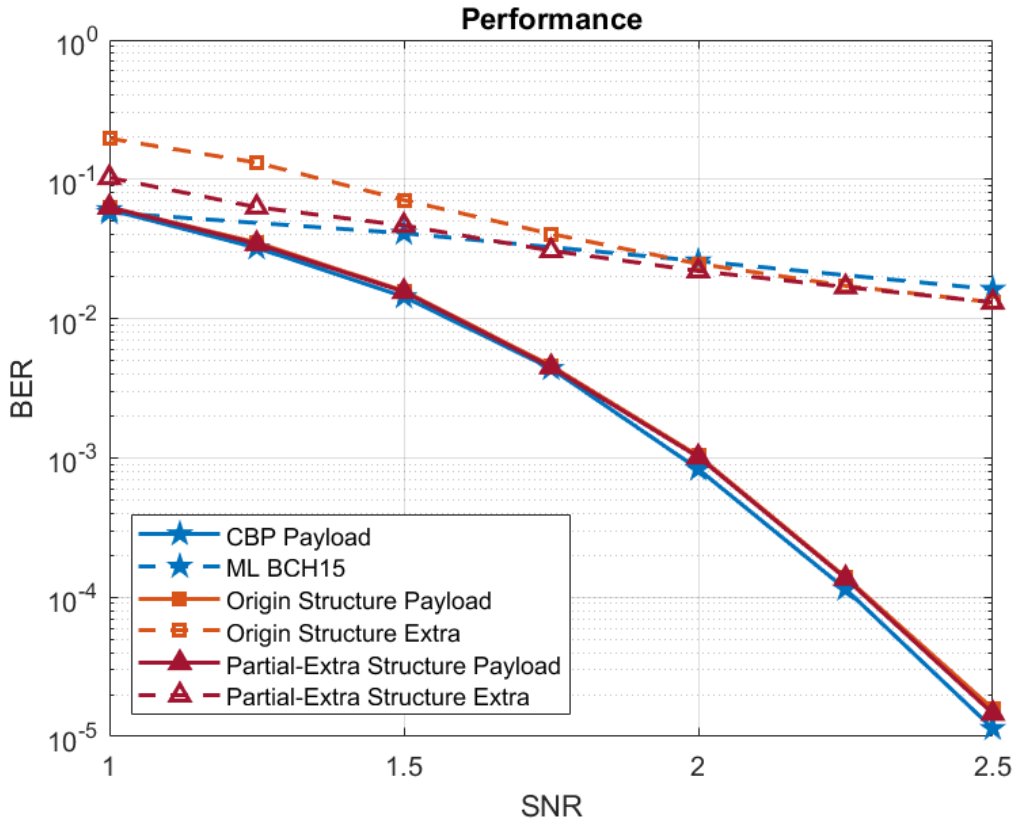




Partial-Extra Structure – Simulation

- Payload : PEG-P1
- Extra : BCH-E2





Partial-Extra Structure – Observation

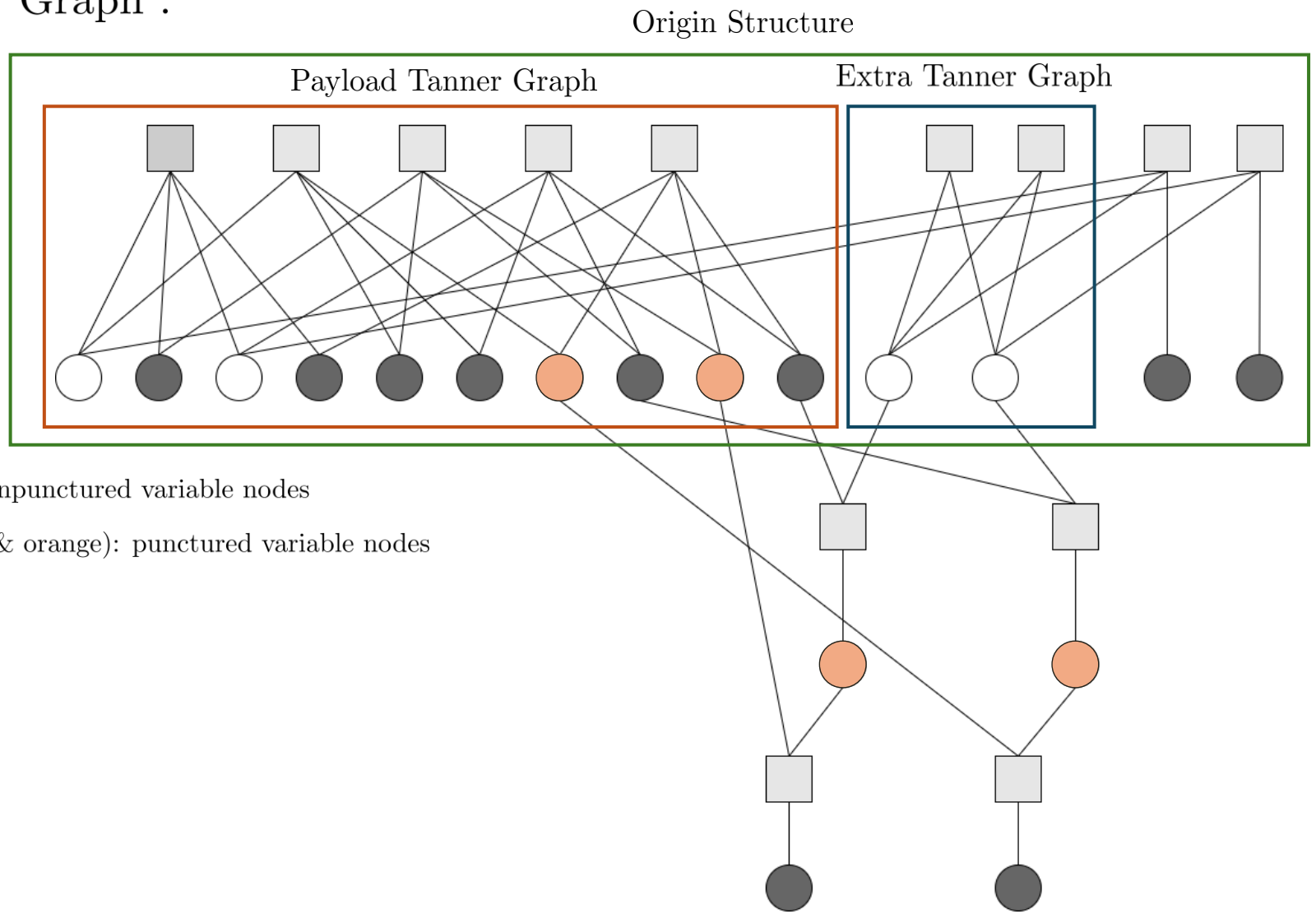
- Merging the Tanner graph allows Extra and Payload to be decoded at the same time, and the payload's performance is barely affected.
- This method of combining Tanner graphs allows Payload and Extra to be decoded in parallel.
- Sending some extra bits can improve Extra's performance at low SNR.

Outline

- **Introduction**
- **Free-Ride Concept**
- **Tanner Graph Combination**
- **Puncture Concept**
  - One–Step Recoverable Node Scheme
  - Rate–Compatible Puncturing Scheme
- **Origin Structure**
- **Partial-Extra Structure**
- **Enhanced Structure**
- **Partial-Extra & Enhanced Structure**
- **Reference**

Enhanced Structure

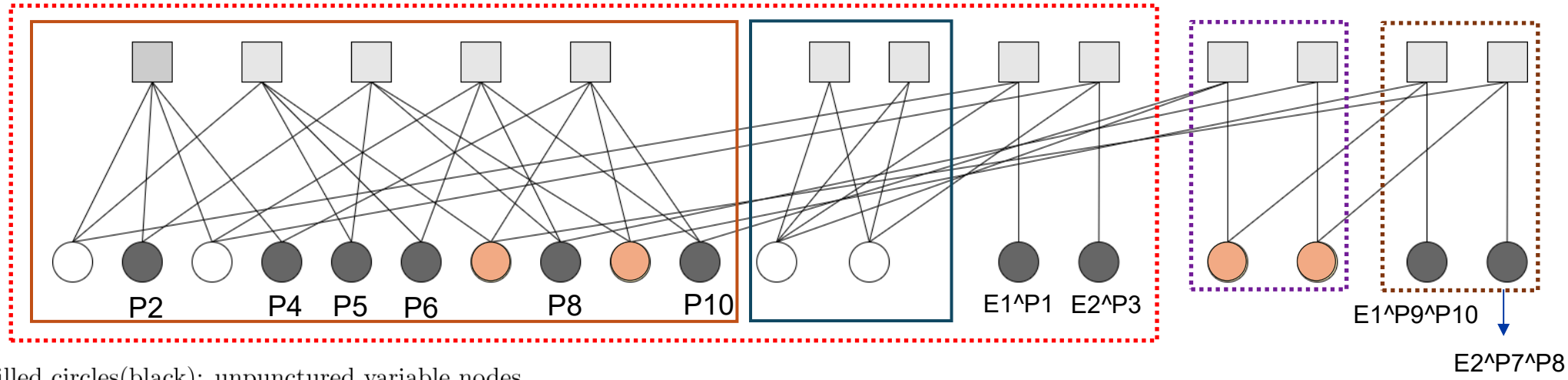
Combine Tanner Graph :



- filled circles(black): unpunctured variable nodes
- unfilled circles(white & orange): punctured variable nodes

Enhanced Structure

Combine Tanner Graph :



- filled circles(black): unpunctured variable nodes
- unfilled circles(white & orange): punctured variable nodes

$$\boldsymbol{H}_{Combine}: \begin{bmatrix} \boldsymbol{H}_{\text{payload}} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{H}_{\text{extra}} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{M}_{\text{Puncpos}} & \boldsymbol{I} & \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{M}_{\text{Enhanced1}} & \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{M}_{\text{Enhanced2}} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{I} \end{bmatrix}$$

$M_{puncpos}$ each row degree is 1.

$M_{Enhanced1}$ each row degree is 1.

$M_{Enhanced2}$ each row degree is 1.

Enhanced Structure - Algorithm

Algorithm 2 Enhanced Combined Tanner Graph Construction

Require: Parity-check matrix $\mathbf{H}$, disjoint payload index sets $\mathcal{I}_1, \mathcal{I}_2$ and stage-2 index set $\mathcal{J}$

Ensure: Enhanced parity-check matrix $\mathbf{H}_{\text{enh}}$

$\mathbf{H}_{\text{enh}} \leftarrow \mathbf{H}$

for each $i \in \mathcal{I}_1$ **do** ▷ Stage 1: original merge (payload bits in $\mathcal{I}_1$ will be punctured)

  Create variable node z_i and check node c_i

  Enforce $p_i \oplus e_i \oplus z_i = 0$

  Insert row for c_i and column for z_i into $\mathbf{H}_{\text{enh}}$

  Connect c_i to p_i, e_i, and z_i

  Puncture payload VN p_i

end for

for each $j \in \mathcal{J}$ **do** ▷ Stage 2: intermediate merge (payload bits in $\mathcal{J}$ are not punctured)

  Create variable node y_j and check node d_j

  Enforce $z_j \oplus p_j \oplus y_j = 0$

  Insert row for d_j and column for y_j into $\mathbf{H}_{\text{enh}}$

  Connect d_j to z_j, p_j, and y_j

end for

for each $i \in \mathcal{I}_2$ **do** ▷ Stage 3: final merge (payload bits in $\mathcal{I}_2$ will be punctured)

  Create variable node x_i and check node f_i

  Enforce $y_i \oplus p_i \oplus x_i = 0$

  Insert row for f_i and column for x_i into $\mathbf{H}_{\text{enh}}$

  Connect f_i to y_i, p_i, and x_i

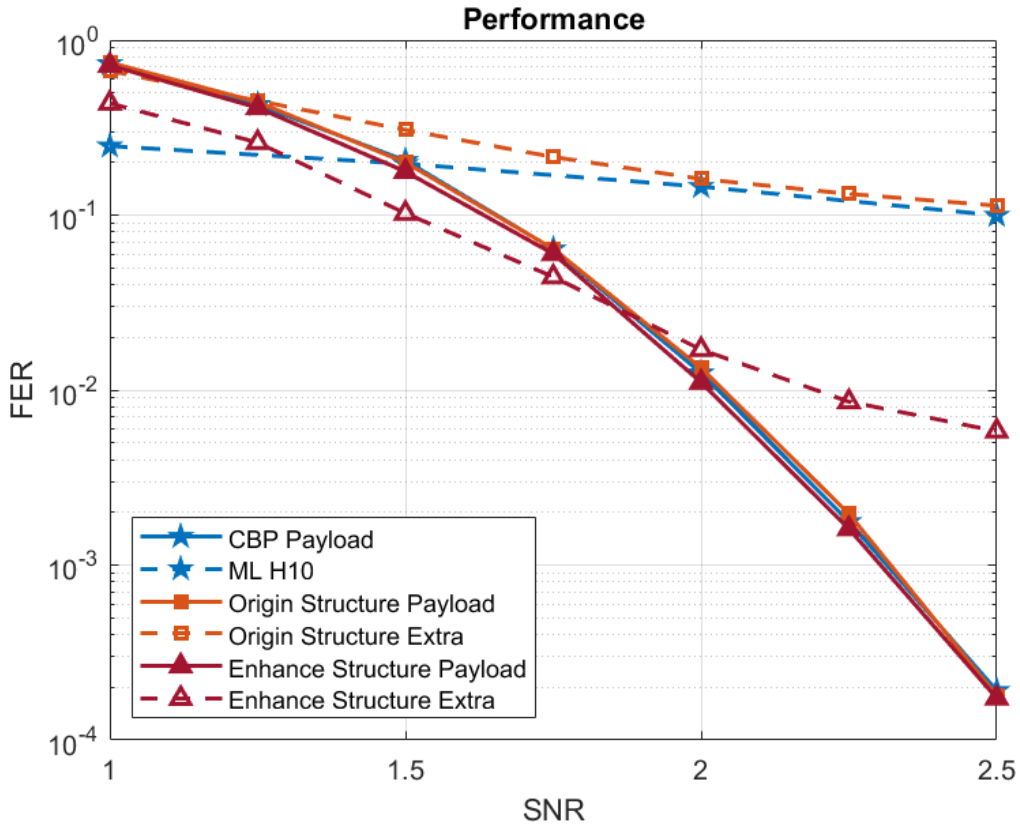
  Puncture payload VN p_i

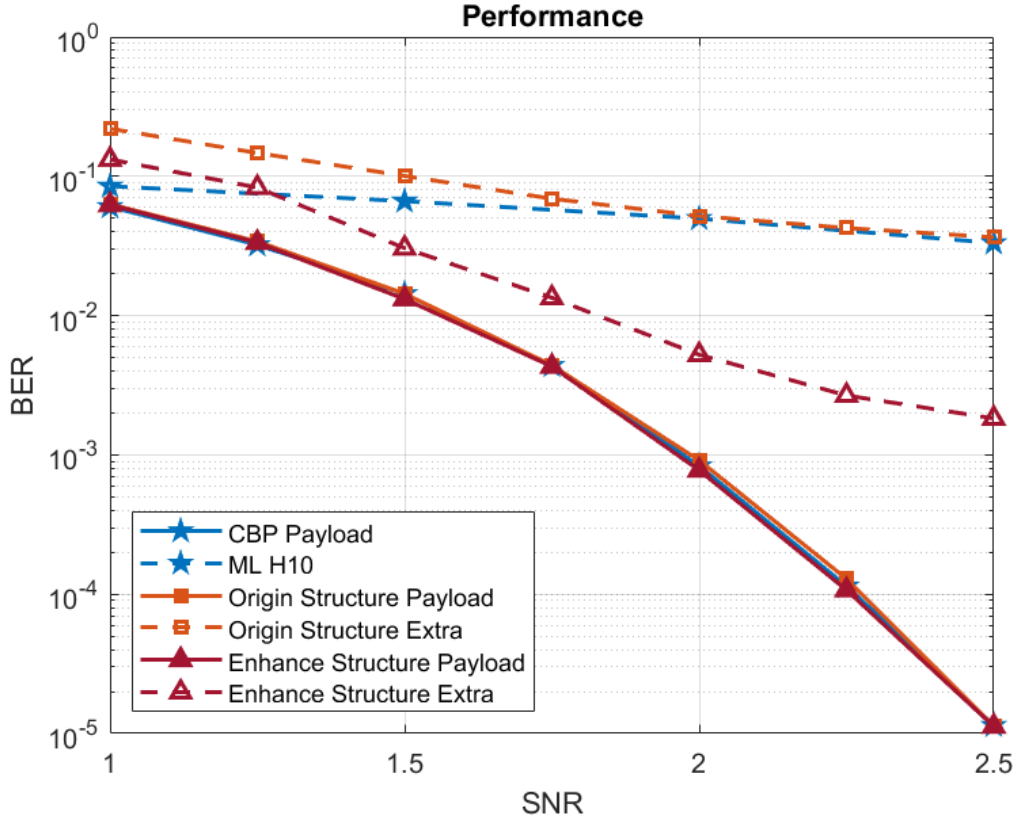
end for

return $\mathbf{H}_{\text{enh}}$

Enhanced Structure – Simulation

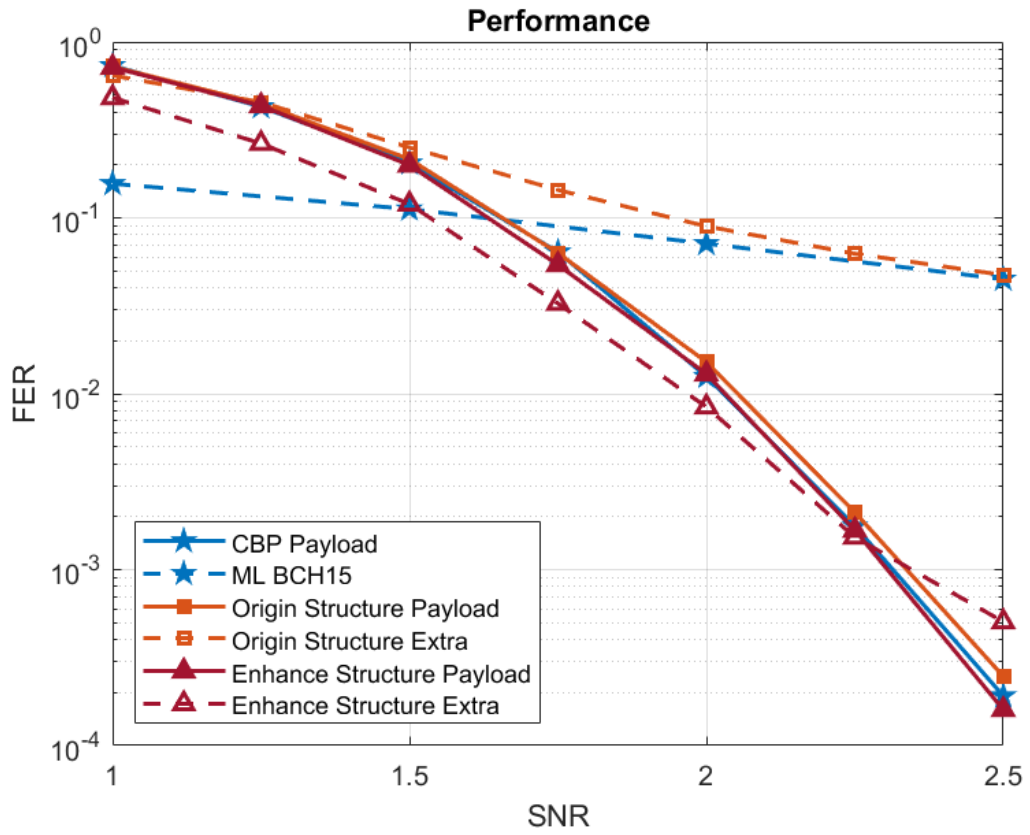
- Payload : PEG-P1
- Extra : LDPC-E1

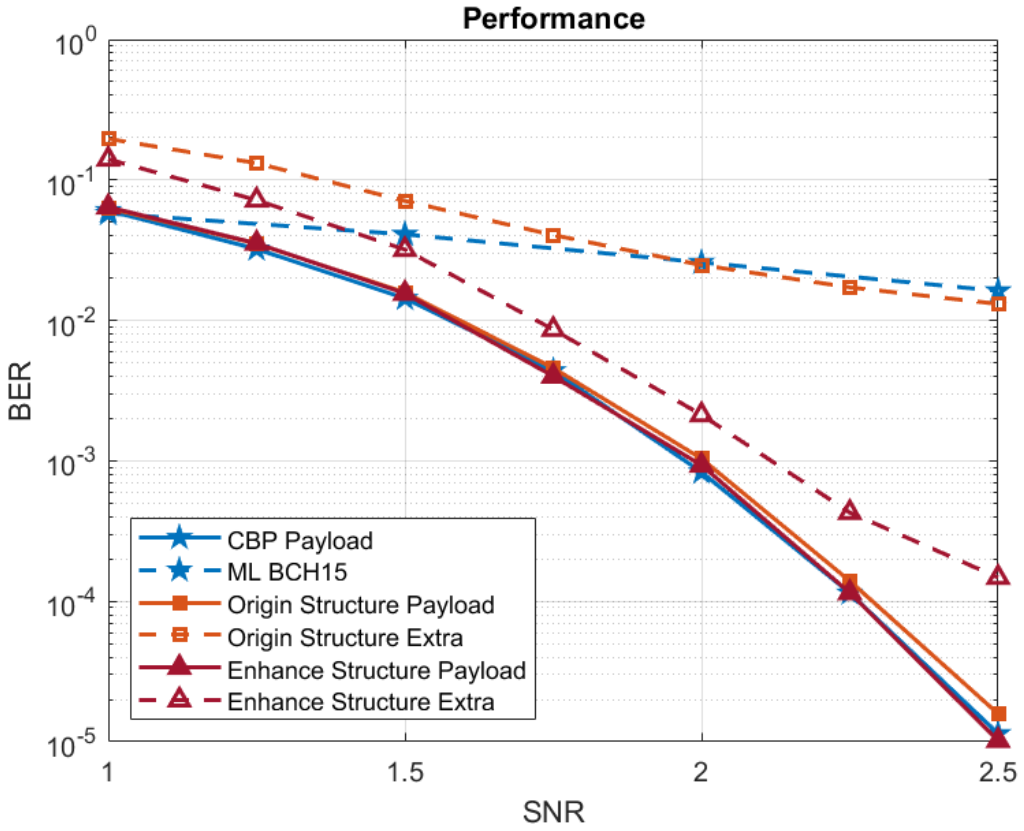




Enhanced Structure – Simulation

- Payload : PEG-P1
- Extra : BCH-E2





Enhanced Structure – minimum distance

Assume

$$d_{\min}(\mathbf{H}_{\text{payload}}) = M, \quad d_{\min}(\mathbf{H}_{\text{extra}}) = N.$$

To determine $d_{\min}(\mathbf{H}_{\text{Enhanced}})$ (the smallest number of columns summing to zero), two cases must be considered:

Case 1: Zero-sum from the payload block

Columns from the payload portion and the three enhancement rows form the submatrix

$$\begin{pmatrix} \mathbf{H}_{\text{payload}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{M}_{\text{puncpos}} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{M}_{\text{Enhanced1}} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{M}_{\text{Enhanced2}} & \mathbf{0} & \mathbf{I} & \mathbf{I} \end{pmatrix}.$$

Enhanced Structure – minimum distance

Let

$$\mathcal{C}_{\text{payload}} = \{\, \vec{c} \in \{0,1\}^{c_1} \mid \vec{c}\mathbf{H}_{\text{payload}}^{T} = 0,\ \text{wt}(\vec{c}) = M \},$$

and

$$S = \{\, j \mid \exists\, \vec{c} \in \mathcal{C}_{\text{payload}} :\ c_j = 1 \}.$$

Define

$$\mathcal{P}_p = \{\, j \mid (\mathbf{M}_{\text{puncpos}})_{*,j} = 1 \}, \quad k_p = |S \cap \mathcal{P}_p|,$$

$$\mathcal{R}_{e1} = \{\, i \mid \exists\, j \in S : (\mathbf{M}_{\text{Enhanced1}})_{i,j} = 1 \}, \quad k_{e1} = |\mathcal{R}_{e1}|,$$

$$\mathcal{R}_{e2} = \{\, i \mid \exists\, j \in S : (\mathbf{M}_{\text{Enhanced2}})_{i,j} = 1,\ i \notin \mathcal{R}_{e1} \}, \quad k_{e2} = |\mathcal{R}_{e2}|.$$

A weight-M payload pattern thus requires

$$M + k_p + 2\,k_{e1} + k_{e2}$$

columns to cancel all flips in the puncture and enhancement identity blocks.
Hence

$$d_{\min}^{(1)} = M + k_p + 2\,k_{e1} + k_{e2}.$$

Enhanced Structure – minimum distance

Case 2: Zero-sum from the extra block

Columns from the extra portion and all four identity blocks form the submatrix

$$\begin{pmatrix} \mathbf{H}_{\text{extra}} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{I} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{I} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{I} \end{pmatrix}.$$

Each weight-N extra codeword flips four identity-block rows in cascade, requiring a total of $4N$ columns to restore zero-sum. Thus

$$d_{\min}^{(2)} = 4N.$$

Taking the smaller of the two cases gives

$$d_{\min}(\mathbf{H}_{\text{Enhanced}}) \;\geq\; \min\big(M + k_p + 2k_{e1} + k_{e2},\; 4N\big).$$

Enhanced Structure – Observation

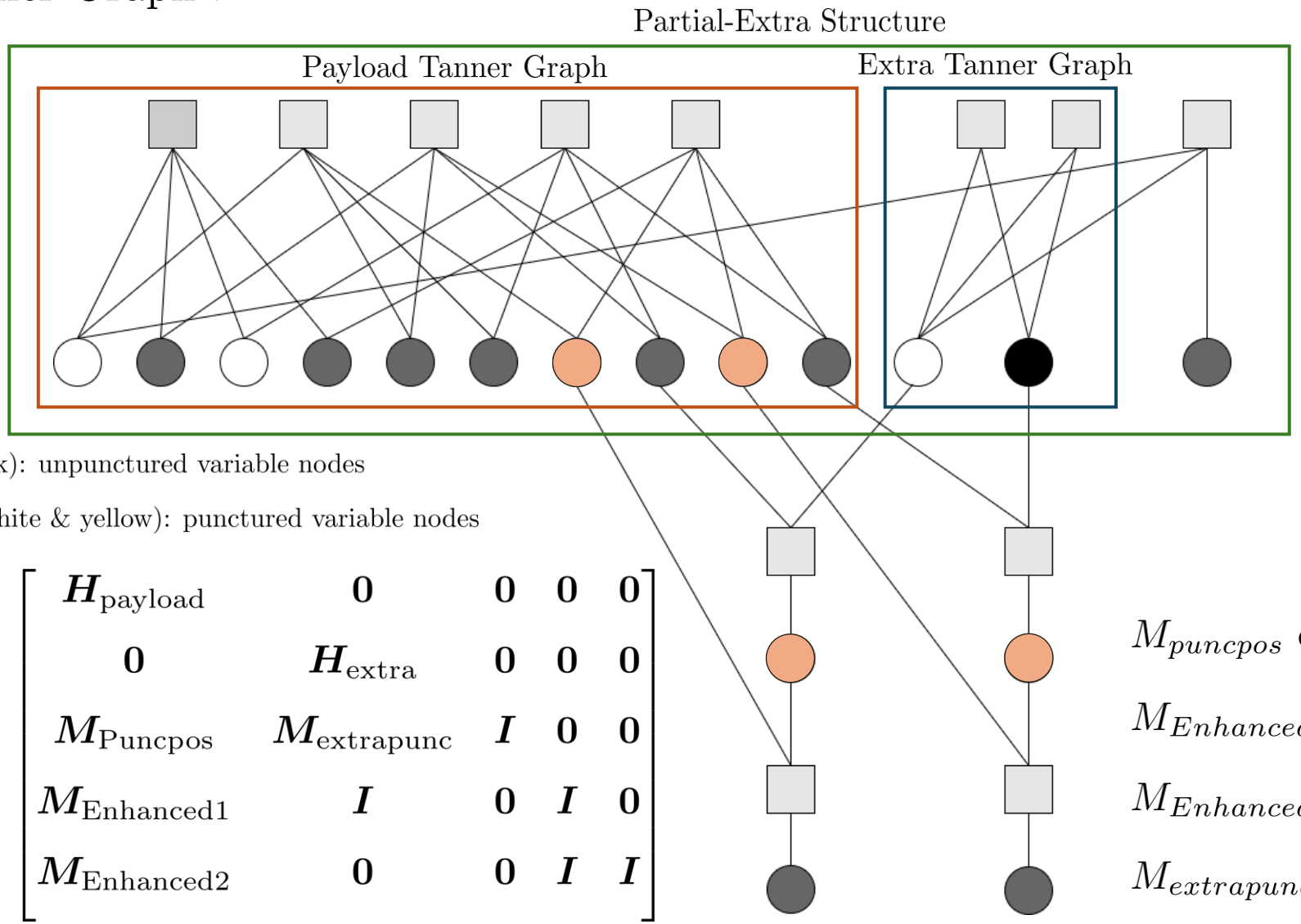
- We observed a significant improvement in the performance of the extra bits, while the performance of the payload remained unchanged.
- The Enhanced Structure can be seen as a large parity-check matrix, where both the payload and the extra bits are part of it. This is why the extra bits can perform even better than maximum likelihood decoding.

Outline

- **Introduction**
- **Free-Ride Concept**
- **Tanner Graph Combination**
- **Puncture Concept**
  - One–Step Recoverable Node Scheme
  - Rate–Compatible Puncturing Scheme
- **Origin Structure**
- **Partial-Extra Structure**
- **Enhanced Structure**
- **Partial-Extra & Enhanced Structure**
- **Reference**

Partial-Extra & Enhanced Structure

Combine Tanner Graph :



- filled circles(black): unpunctured variable nodes
- unfilled circles(white & yellow): punctured variable nodes

$$\boldsymbol{H}_{Combine}: \begin{bmatrix} \boldsymbol{H}_{\text{payload}} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{H}_{\text{extra}} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{M}_{\text{Puncpos}} & \boldsymbol{M}_{\text{extrapunc}} & \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{0} \\ \boldsymbol{M}_{\text{Enhanced1}} & \boldsymbol{I} & \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{M}_{\text{Enhanced2}} & \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{I} & \boldsymbol{I} \end{bmatrix}$$

$M_{puncpos}$ each row degree is 1.

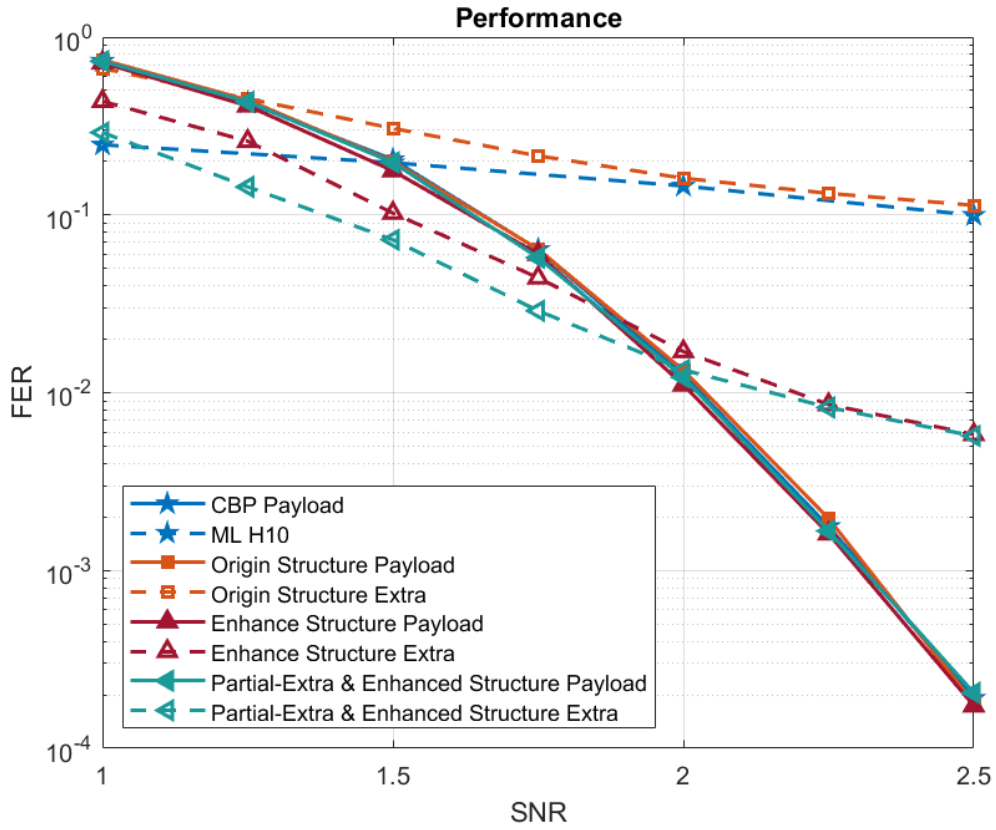
$M_{Enhanced1}$ each row degree is 1.

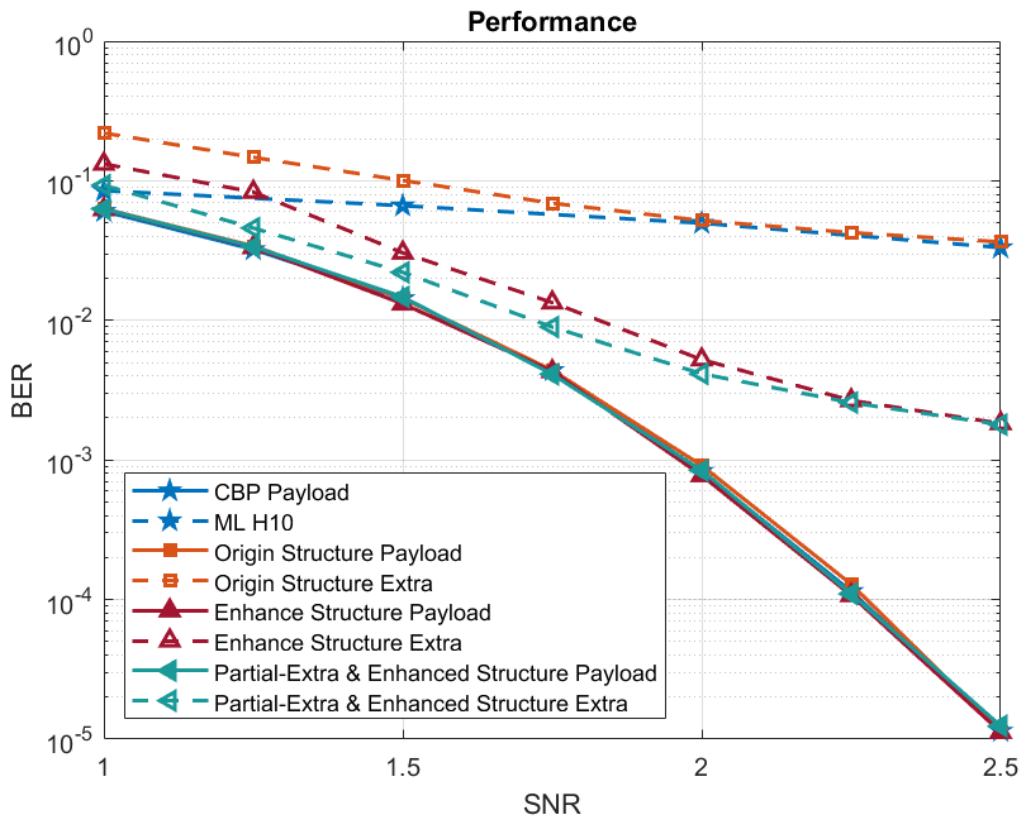
$M_{Enhanced2}$ each row degree is 1.

$M_{extrapunc}$ each row degree is 1.

Partial-Extra & Enhanced Structure – Simulation

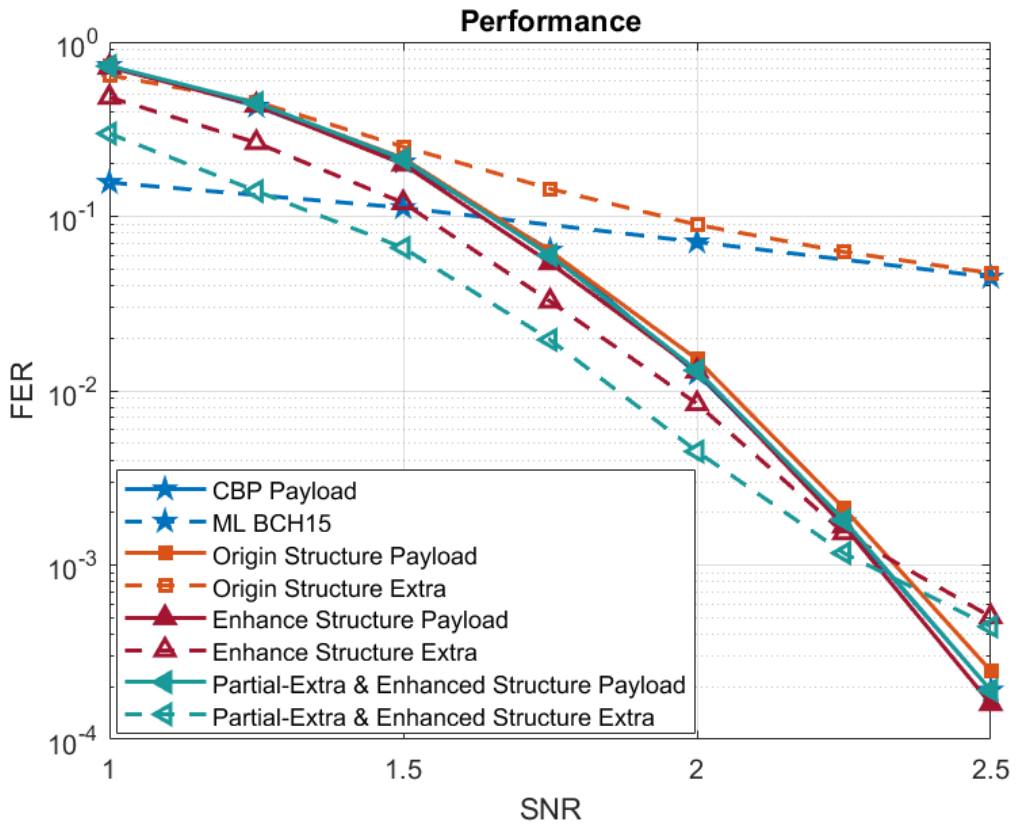
- Payload : PEG-P1
- Extra : LDPC-E1

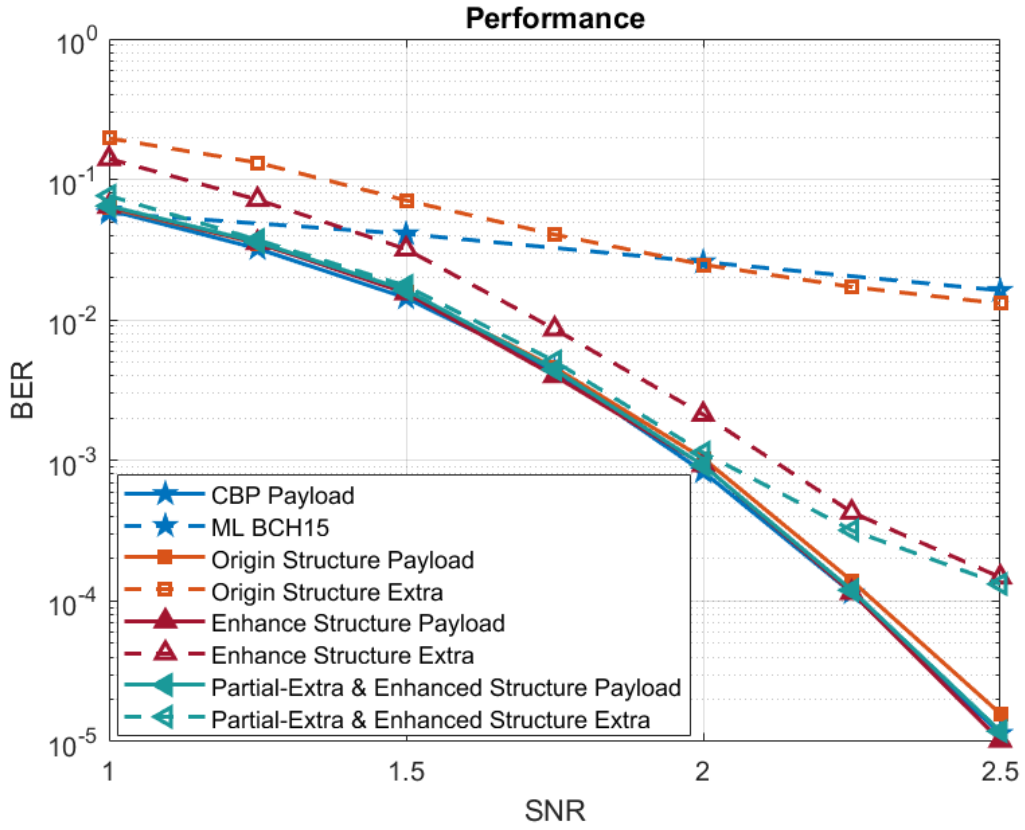




Partial-Extra & Enhanced Structure – Simulation

- Payload : PEG-P1
- Extra : BCH-E2





Partial-Extra & Enhanced Structure– Observation

- After combining the partial-extra structure and the enhanced structure, Extra's performance shows a significant improvement at low SNR.
- The Combined Structure can be seen as a large parity-check matrix, where both the payload and the extra bits are part of it. This is why the extra bits can perform even better than maximum likelihood decoding.

Outline

- **Introduction**
- **Free-Ride Concept**
- **Tanner Graph Combination**
- **Puncture Concept**
  - One–Step Recoverable Node Scheme
  - Rate–Compatible Puncturing Scheme
- **Origin Structure**
- **Partial-Extra Structure**
- **Enhanced Structure**
- **Partial-Extra & Enhanced Structure**
- **Reference**

Reference

[1] S. Cai, S. Zhao and X. Ma, "Free Ride on LDPC Coded Transmission," in IEEE Transactions on Information Theory, vol. 68, no. 1, pp. 80-92, Jan. 2022.

[2] L. Zhang, F. Ma and L. L. Cheng, "A Puncturing Scheme for Low-Density Parity-Check Codes Based on 1-SR Nodes," 2012 IEEE Vehicular Technology Conference (VTC Fall), Quebec City, QC, Canada, 2012.

[3] R. Asvadi and A. H. Banihashemi, "A Rate-Compatible Puncturing Scheme for Finite-Length LDPC Codes," in IEEE Communications Letters, vol. 17, no. 1, pp. 147-150, January 2013.

[4] Jeongseok Ha, Jaehong Kim, D. Klinc and S. W. McLaughlin, "Rate-compatible punctured low-density parity-check codes with short block lengths," in IEEE Transactions on Information Theory, vol. 52, no. 2, pp. 728-738, Feb. 2006

[5] H. Li and L. Zheng, "Efficient Puncturing Scheme for Irregular LDPC Codes Based on Serial Schedules," in IEEE Communications Letters, vol. 19, no. 9, pp. 1508-1511, Sept. 2015

Reference

[6] Q. Wang, L. Chen and X. Ma, "A new HARQ scheme for 5G systems via interleaved superposition retransmission," in China Communications, vol. 20, no. 4, pp. 1-11, April 2023