國立陽明交通大學

電信工程研究所

碩士論文

Institute of Communications Engineering

National Yang Ming Chiao Tung University

Master Thesis

基於 LDPC 碼之自由搭載編碼機制設計及其於 IR-HARQ 之應用

On Designing Free-Ride Coding Schemes Using LDPC Codes and Its Applications for IR-HARQ

研 究 生： 周聖喆 (Zhou, Sheng-Zhe)

指導教授： 余俊宏 (Yu, Jiun-Hung)

張致遠 (Chang, Tofar Chih-Yuan)

中華民國 一一五年一月

January 2026

基於 LDPC 碼之自由搭載編碼機制設計及其於 IR-HARQ 之應用

On Designing Free-Ride Coding Schemes Using LDPC Codes and Its Applications for IR-HARQ

研 究 生： 周聖喆　　Student： Sheng-Zhe Zhou

指導教授： 余俊宏 博士　　Advisor： Dr. Jiun-Hung Yu

張致遠 博士　　Dr. Tofar Chih-Yuan Chang

國立陽明交通大學

電信工程研究所

碩士論文

A Thesis

Submitted to Institute of Communications Engineering

College of Electrical and Computer Engineering

National Yang Ming Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master of Science

in

Communications Engineering

January 2026

Taiwan, Republic of China

中華民國 一一五年一月

誌　謝

感謝我的家人，特別是家人的理解與支持。您們無條件的鼓勵與陪伴，讓我在遇到困難時有無限的力量；也感謝家人的經濟支持，讓我能安心投入研究。

最後，感謝所有曾經幫助過我的人，無論是短暫的指點或持續的鼓勵，都是我完成這篇論文的重要推力。謹以此篇學位論文，向大家表達最誠摯的謝意。

周聖喆 於

國立陽明交通大學 電信工程研究所

中華民國 一一五年一月

基於 LDPC 碼之自由搭載編碼機制設計及其於 IR-HARQ 之應用

學生：周聖喆　　指導教授：余俊宏 博士
張致遠 博士

國立陽明交通大學 電信工程研究所 碩士班

摘　要

在現代數位通訊系統（如地面電視傳輸、5G ／ eMBB 及各種回授鏈路）中，需在不改變既有通道編碼與資源配置的前提下，傳送少量控制或元資料位元（如發射器識別、通道狀態標記、裝置識別等），以保障系統相容性與升級成本低。傳統方法多以獨立通道或修改物理層參數載送側資訊，往往造成頻譜、功率或硬體複雜度的額外開銷，而既有 free-ride 方案雖可疊加額外位元，卻在解碼端需對所有候選進行指數級搜索，導致計算與實作負擔。

本研究提出一種接收端增強式校驗矩陣（Enhanced PCM）與聯合迭代式軟訊息傳遞解碼架構，將原始 LDPC 載荷約束與額外位元統一整合於單一坦納圖（Tanner graph）上，並透過軟消除機制交替更新載荷與額外層輸出。此法省去對所有候選的逐一評分，解碼複雜度僅隨坦納圖邊數線性成長，避免了指數級複雜度爆炸。為支援固定碼長且碼率相容之需求，結合結構化穿孔演算法，自動識別並處理增強式校驗矩陣中對應未觀測變數節點的穿孔位置，於不損及最小距離與 girth 條件下，達成載荷可靠度與額外位元可譯性的最佳化平衡。

模擬結果顯示，IR-HARQ 或 5G IR-HARQ 機制中成功解碼可額外攜帶多個位元，使頻譜效率及吞吐量提升。

關鍵字：低密度奇偶檢查碼, 造碼, 打孔, 最小距離, 吞吐量, 混合式自動重送請求

On Designing Free-Ride Coding Schemes Using LDPC Codes and Its Applications for IR-HARQ

Student : Sheng-Zhe Zhou Advisors: Dr. Jiun-Hung Yu
Dr. Tofar Chih-Yuan Chang

Institute of Communications Engineering
National Yang Ming Chiao Tung University

Abstract

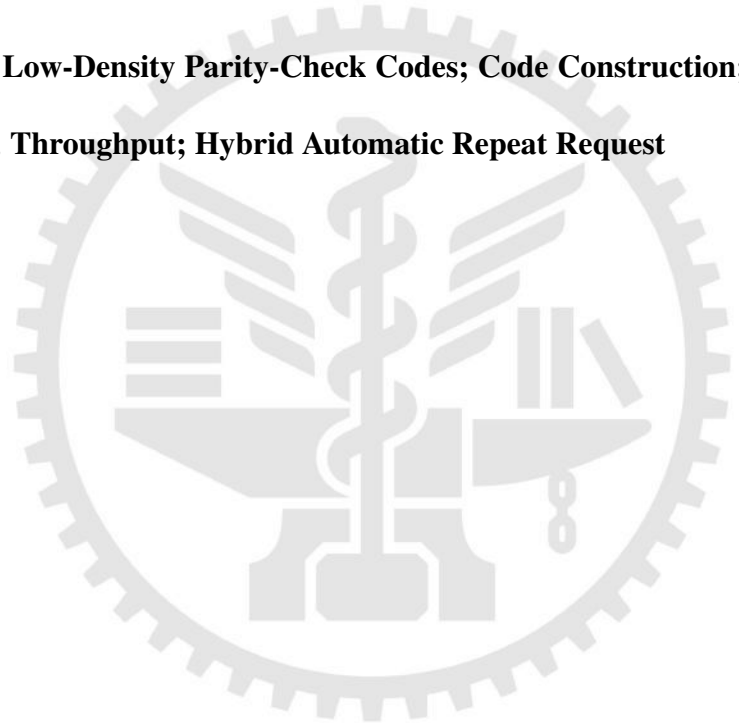
In modern digital communication systems such as terrestrial television broadcasting (DTV), 5G/eMBB, and various feedback links, it is often necessary to transmit a small number of control or metadata bits (for example, transmitter identifiers, channel state indicators, or device identifiers) without modifying the existing channel coding scheme or resource allocation. Conventional methods typically use a separate channel or modify physical layer parameters to carry side information, resulting in additional overhead in spectrum usage, power consumption, or hardware complexity. Although existing free-ride schemes can superimpose extra bits, they require exponential search over all candidates, leading to significant computational and implementation burdens.

This research proposes a receiver-side enhanced parity-check matrix (Enhanced PCM) and a joint iterative soft message passing decoding architecture that integrates the original low-density parity-check (LDPC) payload code constraints and extra code into a single tanner graph. Through a soft cancellation mechanism, payload and extra layers are updated alternately. This approach eliminates the need for an evaluation of all possibilities, reducing decoding complexity to grow linearly with the number of tanner graph edges and avoiding exponential complexity. To support fixed codeword length and rate compatibility, a structured puncturing algorithm is

applied to automatically identify and process punctured variable nodes in the enhanced parity-check matrix corresponding to unobserved channel positions. This achieves an optimal balance between payload reliability and extra-bit decodability without degrading minimum distance and girth properties.

Simulation results show that successful decoding in incremental redundancy hybrid automatic repeat request (IR-HARQ) or 5G IR-HARQ mechanisms can carry multiple additional bits, improving spectral efficiency and throughput.

Keywords: Low-Density Parity-Check Codes; Code Construction; Puncturing; Minimum Distance; Throughput; Hybrid Automatic Repeat Request



Contents

List of Figures

List of Tables

Chapter 1.

Introduction

In modern digital communication systems—from terrestrial television broadcasting (DTV) and enhanced mobile broadband (5G eMBB) to various feedback-link applications, it is often necessary to convey a small amount of control or metadata (for example, transmitter identifiers [1], acknowledgement flags, channel state markers or device identifiers) without altering the existing channel code or resource allocation. Some side-information schemes avoid extra bandwidth by selecting among predefined coding options to signal an additional bit [2] or by slightly rotating the QPSK constellation to carry metadata [3].

This work introduces an alternative "free-ride" paradigm that exploits the gap between the actual LDPC code rate and the channel capacity [4]. The idea of superimposing extra bits onto an LDPC-coded payload was first explored in [5] and subsequently extended to full LDPC-coded transmission in [6]. By applying an XOR operation to superimpose additional bits onto the existing payload codeword, the residual coding capacity can be exploited to convey metadata without any increase in occupied bandwidth or transmit power.

Recently, the free-ride coding paradigm has been extended to a wide range of advanced communication scenarios, demonstrating its versatility. For instance, free-ride coding has been applied to reduce the Age of Information (AoI) in mission-critical and multi-access networks [7, 8], and to facilitate the transmission of semantic features in wireless video surveillance systems [9]. In the context of retransmission protocols, it enables efficient feedback and superposition

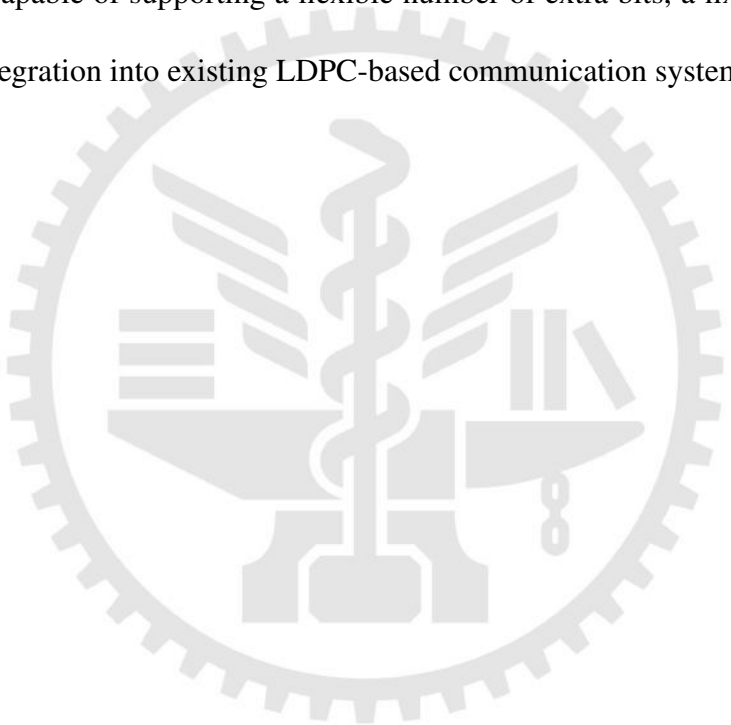
retransmission over LDPC-coded links [10]. Furthermore, the principle of free-ride coding has been exploited to construct advanced code structures with improved performance, such as spatially and globally coupled LDPC codes [11, 12], product LDPC codes [13], and to aid in the parallel decoding of PIC-LDPC codes [14]. While these works underscore the potential of free-ride coding, they also highlight the growing need for efficient, low complexity decoding architectures to support these diverse implementations robustly.

Let k_1 denote the number of extra information bits to be carried by the free-ride scheme. A key innovation of this work is a unified decoding architecture. Traditional free-ride method evaluate all 2^{k_1} possible extra bit combinations one by one. Such an exhaustive search is equivalent to maximum-likelihood (ML) or list decoding over an enlarged codeword space and causes computational effort and memory use to grow exponentially as k_1 increases.

By contrast, the proposed method constructs a single enhanced parity-check matrix at the receiver. This matrix incorporates the original LDPC parity constraints and the extra bit constraints into one tanner graph. A belief-propagation (BP) algorithm with log-likelihood-ratio (LLR) messaging then runs just once on this unified graph. At each iteration, messages from the LDPC portion serve as inputs to the extra bit portion, and the extra bit soft outputs are subtracted from the payload observations before the next LDPC update. Because decoding takes place in a single pass over one graph, complexity grows roughly in proportion to the total number of edges rather than exponentially in k_1. The transmitter requires no changes, and only a plug-in joint decoder is added at the receiver, which greatly simplifies practical deployment.

Since the transmitted codeword length is maintained, XOR extra bits produces an enhanced parity-check matrix with a column count exceeding the original payload length. In this construction, any variable nodes whose posterior LLR values remain at zero, indicating the absence of direct channel observations, are treated as punctured nodes. A structured puncturing algorithm

is then applied to identify precisely which payload positions have been effectively punctured. This algorithm evaluates the impact of each potential puncture on graph properties such as cycle length and node degree distribution, ensuring that punctured positions correspond to variable nodes with minimal degradation of the LDPC decoding threshold and minimum distance. By offsetting puncture locations away from regions where extra bits are injected, this approach achieves a balanced trade-off where payload reliability is preserved while joint decoding of both payload and extra bits remains robust. The end result is a unified transmission and decoding framework capable of supporting a flexible number of extra bits, a fixed codeword length, and seamless integration into existing LDPC-based communication systems.



Chapter 2.

Preliminary

2.1 Low-Density Parity-Check Codes

Low-density parity-check (LDPC) codes [15] are a class of linear block codes defined by an extremely sparse parity check matrix $\mathbf{H}$. In an $m \times n$ binary matrix, the number of ones in each row and column is much smaller than n and m, respectively, which is why these codes are called "low-density". First proposed by Gallager in 1962, LDPC codes were recognized for their ability to approach the shannon limit while keeping complexity under control.

Because of the sparse structure of $\mathbf{H}$, LDPC codes remain computationally efficient even at large block lengths and exhibit error-rate performance very close to the shannon limit under low signal-to-noise ratio conditions. Because of these advantages, they are now widely used in wireless communications (e.g., IEEE 802.11, 5G NR), space communications following consultative committee for space data systems (CCSDS) standards, and data storage systems such as hard disk drives and solid-state drives.

2.1.1 Definition

An (n, k) low-density parity-check (LDPC) code $\mathcal{C} \subset \mathbb{F}_2^n$ is the linear block code defined by a binary parity-check matrix $\mathbf{H}$ of size $(n-k) \times n$ such that $\mathcal{C} = \{\mathbf{c} \in \mathbb{F}_2^n : \mathbf{H}\,\mathbf{c}^T = 0\}$. The code rate is $R = k/n$. The matrix $\mathbf{H}$ is low density because it contains only a small fraction

of ones; equivalently, each row has weight much smaller than n and each column has weight much smaller than $n - k$. LDPC codes may be *regular*, with uniform row and column weights, or *irregular*, with weights drawn from distributions chosen to optimize performance.

2.1.2 Belief Propagation Algorithm

First, to convert an $m \times n$ parity-check matrix $\mathbf{H}$ into a tanner graph. In this bipartite graph, create a variable node v_j for each of the n columns of $\mathbf{H}$ and a check node c_i for each of the m rows. Draw an edge between v_j and c_i if and only if $H_{i,j} = 1$. Thus, each check node connects exactly to the set of variable nodes involved in that parity equation. An example of a parity-check matrix is shown in Fig. 2.1, and its corresponding tanner graph is illustrated in Fig. 2.2.

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

Figure 2.1: Parity-check matrix example

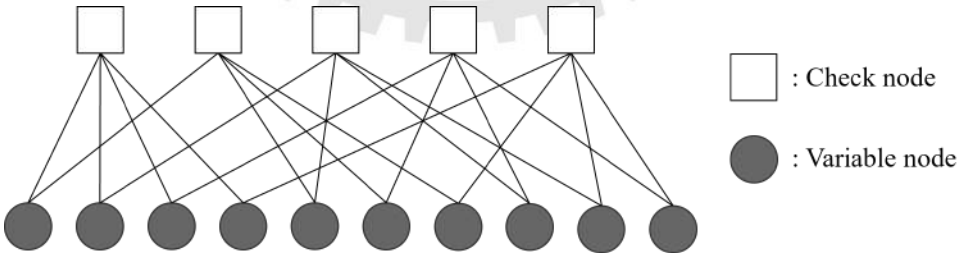


Figure 2.2: Tanner graph example

Check-To-Variable Message Updating

At each iteration t, every check node c_i computes a message to each neighboring variable node v_j as

$$m_{c_i \to v_j}^{(t)} = 2 \tanh^{-1}\left(\prod_{j' \in \mathcal{N}(i) \setminus j} \tanh\left(\tfrac{1}{2} m_{v_{j'} \to c_i}^{(t-1)}\right) \right), \tag{2.1}$$

where $\mathcal{N}(i)$ is the set of variable nodes connected to c_i, and $m_{v_{j'} \to c_i}^{(t-1)}$ are the incoming messages from all other variable nodes in the previous iteration, Fig. 2.3 for a graphical illustration of the check to variable message update.

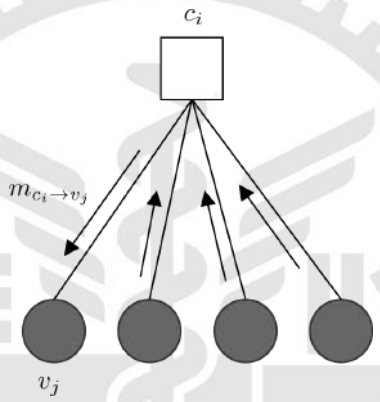


Figure 2.3: Check to variable message update

Variable-To-Check Message Updating

Each variable node v_j then updates its message to each check node c_i as

$$m_{v_j \to c_i}^{(t)} = L_j \; + \sum_{i' \in \mathcal{M}(j) \setminus i} m_{c_{i'} \to v_j}^{(t)}, \tag{2.2}$$

where L_j is the channel LLR for bit j, $\mathcal{M}(j)$ is the set of check nodes connected to v_j, and the sum runs over all incoming check-to-variable messages except from c_i. Fig. 2.4 for a graphical illustration of the variable to check message update.

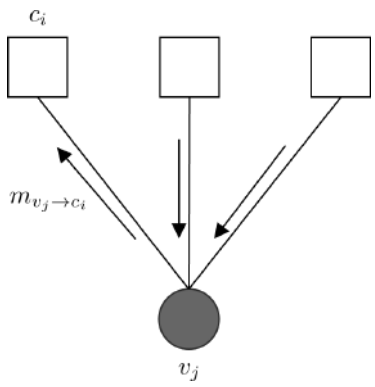


Figure 2.4: Variable to check message update

2.1.3 Sum-Product Algorithm

Assume the transmitted codeword is $\mathbf{c} = [c_0, c_1, \ldots, c_{n-1}]$ and the received channel observations are $\mathbf{y} = [y_0, y_1, \ldots, y_{n-1}]$. To compute the channel LLRs for each bit:

$$L_j \;=\; \ln \frac{P(y_j \mid c_j = 0)}{P(y_j \mid c_j = 1)}, \qquad j = 0, 1, \ldots, n-1. \tag{2.3}$$

Under additive white gaussian noise (AWGN) with zero mean and variance σ^2, the received sample is modeled as

$$y_j = (-1)^{c_j} + n_j, \quad n_j \sim \mathcal{N}(0, \sigma^2). \tag{2.4}$$

Substituting this into the LLR definition yields the simple form

$$L_j = \frac{2\, y_j}{\sigma^2} = \ln \frac{P(y_j \mid c_j = 0)}{P(y_j \mid c_j = 1)}\,. \tag{2.5}$$

These LLRs serve as the initial "variable-to-check" messages:

$$m^{(0)}_{v_j \to c_i} = L_j, \quad \forall\, (v_j, c_i) \in E, \tag{2.6}$$

The posterior LLRS is computed to make a hard decision:

$$L_j^{\text{post}} = L_j + \sum_{i \in \mathcal{M}(j)} m_{c_i \to v_j}^{(t)}. \tag{2.7}$$

where E denotes the set of edges in the tanner graph. Each iteration first updates messages at the check nodes, then updates messages at the variable nodes, and finally combines these messages to compute the updated posterior LLRs.

Algorithm 1 Sum-Product Algorithm for LDPC Decoding

Input: parity-check matrix $\mathbf{H}$ of size $(m \times n)$, received vector $\mathbf{y} = [y_0, \ldots, y_{n-1}]$, noise variance σ^2, maximum iterations $T_{\max}$

Initialization:
for $j = 0$ to $n - 1$ **do**
    Compute channel LLR L_j using (2.5)
    for each check node c_i adjacent to v_j **do**
        Initialize $m_{v_j \to c_i}^{(0)}$ using (2.6)
    end for
end for
for $t = 1$ to $T_{\max}$ **do**
    Check-node update:
    for each check node c_i **do**
        for each variable node $v_j \in \mathcal{N}(i)$ **do**
            Update $m_{c_i \to v_j}^{(t)}$ using (2.1)
        end for
    end for
    Variable-node update:
    for each variable node v_j **do**
        for each check node $c_i \in \mathcal{M}(j)$ **do**
            Update $m_{v_j \to c_i}^{(t)}$ using (2.2)
        end for
    end for
    Posterior LLR and hard decision:
    for $j = 0$ to $n - 1$ **do**
        Compute L_j^{post} using (2.7)
        if $L_j^{\text{post}} > 0$ **then**
            $\hat{c}_j \leftarrow 0$
        else
            $\hat{c}_j \leftarrow 1$
        end if
    end for
    if $\mathbf{H}\,\hat{\mathbf{c}}^T = \mathbf{0}$ **then**
        break
    end if
end for
return $\hat{\mathbf{c}} = [\hat{c}_0, \ldots, \hat{c}_{n-1}]$

Despite the simplicity of message updates, the BP algorithm is not guaranteed to converge for all LDPC codes, especially for loopy tanner graphs. Convergence behavior is typically analyzed through density evolution or extrinsic information transfer (EXIT) charts in asymptotic regimes. In finite-length codes, performance depends on cycle structures and trapping sets.

2.2 Minimum Distance of Linear Block Codes

For a q-ary (n, k) linear block code $\mathcal{C} \subseteq \mathbb{F}_q^n$, the *minimum distance* is defined as the minimum hamming weight among all nonzero codewords:

$$d_{\min} \triangleq \min_{\mathbf{x} \in \mathcal{C} \setminus \{\mathbf{0}\}} w_H(\mathbf{x}) = \min_{\substack{\mathbf{x}, \mathbf{y} \in \mathcal{C} \\ \mathbf{x} \neq \mathbf{y}}} d_H(\mathbf{x}, \mathbf{y}), \tag{2.8}$$

where $w_H(\cdot)$ denotes hamming weight and $d_H(\cdot, \cdot)$ denotes hamming distance. For linear codes the two expressions are equivalent because $\mathbf{x} - \mathbf{y} \in \mathcal{C}$. Intuitively, $d_{\min}$ measures how close two distinct codewords can be and is the central structural parameter governing error resilience.

In terms of error-control capability under hard-decision, a code with minimum distance $d_{\min}$ can detect up to $d_{\min} - 1$ symbol errors and uniquely correct

$$t = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor \tag{2.9}$$

symbol errors. With a mixture of errors and erasures (erasures at known positions), unique decoding is guaranteed whenever $2t + \rho < d_{\min}$, where t is the number of errors and ρ the number of erasures. These are worst-case guarantees; in practice, soft/ML decoding can often correct beyond t on average. At high signal-to-noise ratio (SNR), however, $d_{\min}$ together with the population of low weight codewords dominantly shapes the error floor.

The minimum distance can be characterized via either a generator or a parity-check description. If $\mathcal{C} = \{\mathbf{uG} : \mathbf{u} \in \mathbb{F}_q^k\}$ with generator matrix $\mathbf{G}$, then

$$d_{\min} \;=\; \min_{\mathbf{u}\neq\mathbf{0}} w_H(\mathbf{u}G), \tag{2.10}$$

i.e., the minimum weight among nonzero linear combinations of rows of $\mathbf{G}$. If H is a parity-check matrix of $\mathcal{C}$, then $d_{\min}$ equals the size of a smallest linearly dependent set of columns of $\mathbf{H}$. The support of such a smallest dependent set corresponds to a minimum-weight codeword, providing a structural pathway to reason about $d_{\min}$ from $\mathbf{H}$.

Let A_w denote the number of codewords of weight w. Then

$$d_{\min} \;=\; \min\{\, w \geq 1 \;:\; A_w > 0 \,\}. \tag{2.11}$$

Union-bound style high SNR approximations for error probability are dominated by $d_{\min}$ and $A_{d_{\min}}$. Therefore, good codes aim not only to increase $d_{\min}$ but also to suppress the multiplicity of small-weight codewords to lower the error floor.

Computing $d_{\min}$ exactly is generally difficult for long codes and is known to be NP-hard in general [16]. For small dimensions one can exhaustively search over all nonzero $\mathbf{u}$ to evaluate $w_H(\mathbf{uG})$. For larger codes, one may employ probabilistic algorithms [17] or heuristics such as ordered-statistics decoding (OSD) [18] to find low weight codewords and thereby obtain a lower bound on $d_{\min}$. For sparse graph codes (e.g., LDPC), short cycles and stopping sets are not equal to $d_{\min}$ but often guide searches for suspicious low weight structures.

In summary, $d_{\min}$ is a key metric for both theory and engineering practice: it provides hard guarantees for error detection/correction, governs the high SNR error floor through the weight spectrum, and interacts with code rate, structural density, and decoding complexity in

fundamental design tradeoffs.

2.3 Overview of Puncturing

Puncturing is a rate-adaptation technique for LDPC codes that increases the effective code rate by selectively omitting (or "puncturing") certain encoded bits before transmission. Given an (n, k) parent LDPC code with rate $R = k/n$, puncturing p bits yields a shortened codeword of length $n - p$ and an effective rate

$$R_{\text{eff}} = \frac{k}{n-p}\,. \tag{2.12}$$

Since only the unpunctured bits are sent over the channel, the decoder must treat the punctured positions as erasures, initializing their channel LLRs to zero.

The primary motivation for puncturing is to provide fine-grained rate flexibility without redesigning the parity-check matrix. In adaptive systems, this allows a single mother code to support multiple target rates by varying p. Moreover, by leveraging channel state information (CSI), the number of transmitted bits can be dynamically adjusted to match current channel conditions. Practical puncturing algorithms aim to select positions whose removal degrades decoding performance as little as possible-often based on metrics such as variable node degree, bit reliability estimates, or protograph structure.

2.3.1 Puncturing Scheme for Finite-Length LDPC Codes

Puncturing achieves rate-compatible LDPC codes by selectively removing parity bits. To mitigate performance loss, prior works have investigated reliability-based grouping strategies [19], optimization for serial decoding schedules [20], and non-greedy search techniques to avoid

local optima [21]. More recently, puncturing patterns have been further optimized for 5G NR applications requiring low latency decoding [22].

Among these strategies, algorithms focusing on the recoverability of variable nodes (VNs) have shown significant promise for finite-length codes. In this thesis, two representative algorithms based on node recoverability are specifically investigated: the k-Step Recoverable (k-SR) algorithm and the Recovery Tree method. The operational principles and performance comparisons of these two methods are detailed in the following subsections.

k-Step Recoverable Concept

The k-SR algorithm [23] groups punctured variable nodes by the number of decoding iterations needed for first recovery. Nodes that recover in the first iteration form G_1 (one-step recoverable, 1-SR); those that recover in two iterations form G_2 (2-SR); and so on up to G_k (k-SR). Unpunctured nodes belong to G_0. The same Fig. 2.5 illustrates both 1-SR and 2-SR nodes.

Fig. 2.5 also shows the idea of survived and dead check nodes. A check node p is survived if it has at least one edge to an unpunctured variable node; otherwise it is dead. A 1-SR variable node has at least one survived check node whose other neighbors all lie in G_0. A 2-SR node must rely on neighbors in G_1 to pass recovery messages, and so on.

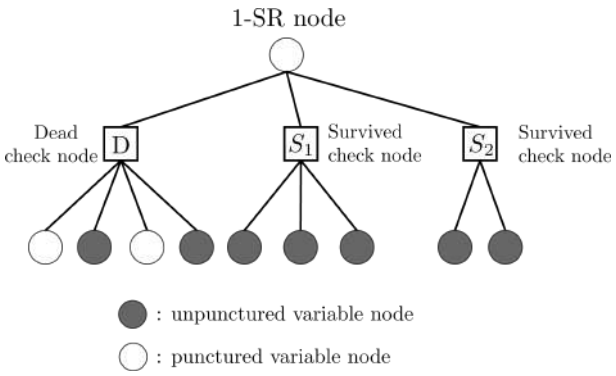


Figure 2.5: One-step recoverable node example

Formally, a punctured variable node v belongs to G_1 if there exists a check node $p \in \mathcal{N}(v)$ with

$$\mathcal{N}(p) \setminus \{v\} \subseteq G_0. \tag{2.13}$$

For $\ell > 1$, v belongs to G_ℓ if there exists $p \in \mathcal{N}(v)$ with

$$\mathcal{N}(p) \setminus \{v\} \subseteq \bigcup_{r=0}^{\ell-1} G_r. \tag{2.14}$$

A recovery tree is used to measure reliability for any $v \in G_\ell$ (see Fig. 2.6). Starting from v, the tree spans survived check nodes and their adjacent variables. The probability of recovery within ℓ iterations is

$$\Psi(v,\zeta) = \begin{cases} 1-\zeta, & v \in G_0, \\ \prod_{j=1}^{d_c(v)-1} \Psi(\gamma_j,\zeta), & v \in G_\ell,\ \ell > 0, \end{cases} \tag{2.15}$$

where ζ is the channel erasure rate and $\{\gamma_j\}$ are the other variable nodes attached to each survived check node p. The error recovery probability is then

$$P_e(v) = \tfrac{1}{2}\big(1 - \Psi(v,\zeta)\big). \tag{2.16}$$

Nodes are sorted by $P_e(v)$ in ascending order. Those with the lowest recovery error probability are punctured first, which maintains decoding performance while allowing flexible rate selection.

Nodes are sorted by $P_e(v)$ in ascending order. Those with the lowest recovery error probability are punctured first, which maintains decoding performance while allowing flexible rate selection. In particular, nodes in 1-SR [24] are given highest puncturing priority, since they

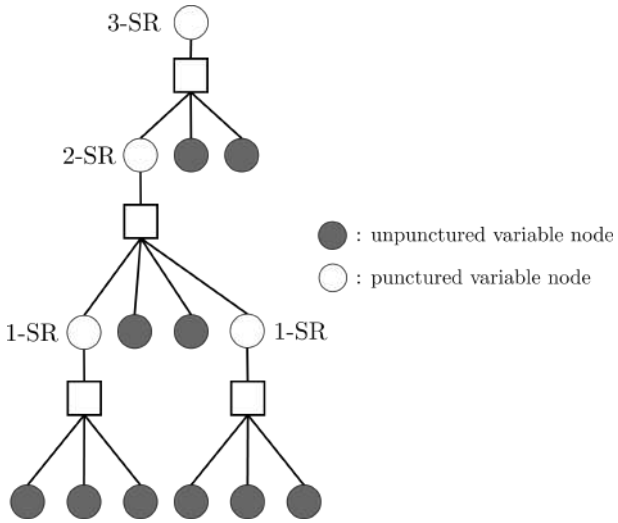


Figure 2.6: Recovery tree example

exhibit the smallest error recovery probability.

Approximate Cycle Extrinsic Message Degree and Minimum Recovery Tree for Puncturing

In a puncturing scheme proposed with minimun recovery tree [25], two key concepts, approximate cycle extrinsic message degree (ACE) and the minimum recovery tree, are introduced to guide the selection of bits to puncture. This method aims to maximize the code rate while minimizing the impact on decoding performance.

The minimum recovery tree is a structure used to evaluate the recovery capability of each variable node in the decoding process. The recovery tree consists of all survived check nodes connected to the variable node v, as well as other variable nodes connected to those check nodes. The depth or branching factor of this tree determines the recovery capability of the node. By focusing on the most reliable recovery paths, the algorithm ensures that the least critical nodes (those whose puncturing would most minimally affect decoding) are chosen first.

The reason some check nodes can be safely removed is that they no longer provide additional useful information for the recovery process, or their recovery paths are already sufficiently

reliable. When selecting bits to puncture, if certain check nodes no longer contribute to the recovery process (for example, if all their neighboring variable nodes are punctured or if they cannot provide valid recovery paths), those check nodes can be safely deleted. This reduces redundant calculations in the graph, improving computational efficiency while ensuring decoding performance is not significantly affected.

ACE [26] is a performance metric that quantifies the contribution of a variable node to the decoding process, particularly in terms of its participation in short error-correcting cycles. The ACE metric is designed to evaluate how much a node impacts the overall error recovery by considering its involvement in cycles within the tanner graph. These cycles can lead to errors in decoding if the wrong bits are punctured. ACE measures this impact by examining the error-prone cycles that a variable node participates in and prioritizes those nodes whose removal will have the least effect on the overall decoding reliability.

Together, these two metrics, the minimum recovery tree and ACE, enable the puncturing algorithm to select the bits that will have the least detrimental effect on decoding performance, while also increasing the rate. A graphical illustration of the minimum recovery tree can be seen in Fig. 2.7, which shows how recovery trees are constructed for different nodes in the graph.

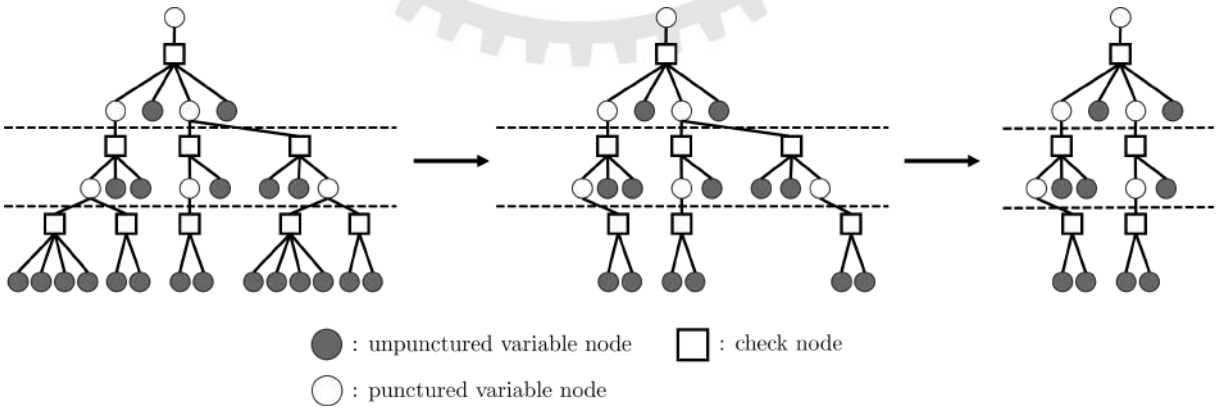


Figure 2.7: Minimum recovery tree example: A visual representation of how the recovery tree is constructed for a variable node and its associated survived check nodes.

2.4 Incremental Redundancy Hybrid Automatic Repeat Request

Automatic Repeat reQuest (ARQ) utilizes error detection and feedback (ACK/NACK) to control data transmission. The transmitter resends data upon receiving a NACK and proceeds only after an ACK. Although effective in stable channels, the absence of error correction capability causes severe throughput degradation and delay under poor channel conditions [27].

To address these limitations, hybrid ARQ (HARQ) incorporates forward error correction into the retransmission loop. A sophisticated variant known as incremental redundancy HARQ (IR-HARQ) generates multiple redundancy versions from a low rate mother code. Upon a decoding failure, the transmitter sends only new parity bits rather than repeating the entire data block. The receiver combines these incremental bits with the previously stored soft information, effectively forming a more powerful code with a decreasing rate. This adaptability allows the system to achieve significant coding gain and maintain high throughput even in varying channel conditions.

Fig. 2.8 illustrates the operation timeline of IR-HARQ. In the first round the transmitter sends partial redundancy RV0 and the receiver fails to decode and returns NAK-A. In the second round it sends a different redundancy RV1, and after combining the soft information from RV0 and RV1 the receiver still cannot decode and returns NAK-A again. In the third round it sends RV2, and after combining RV0, RV1, and RV2 the receiver finally decodes successfully and returns ACK-A. The figure also shows a favorable channel case in which RV0 alone succeeds and ACK-B is returned. Overall, IR-HARQ provides new redundancy on each retransmission and combines it at the receiver, which effectively lowers the code rate step by step and turns failure into success.

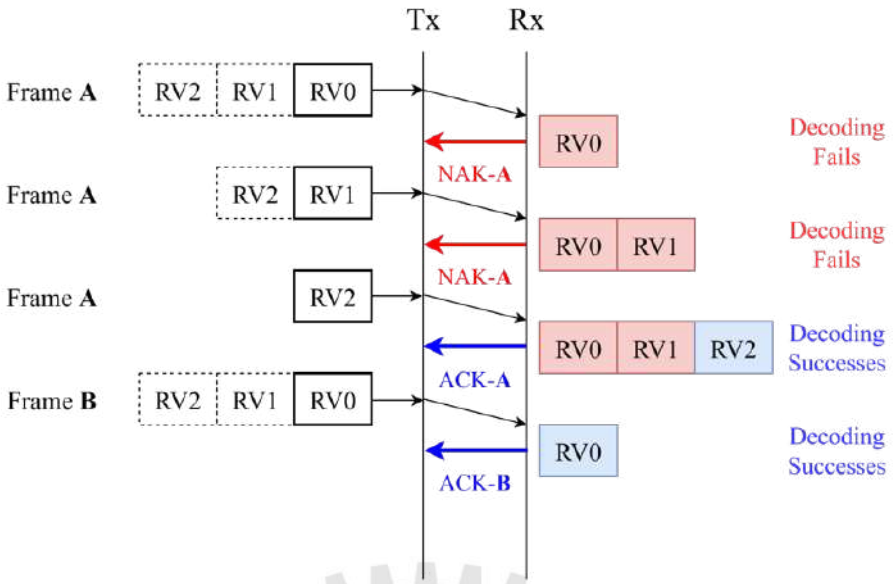


Figure 2.8: Basic diagram of IR-HARQ

2.5 5G NR LDPC

Fifth generation mobile communications (5G NR, New Radio) was first standardized by the 3GPP in Release 15 [28], aiming to meet future cellular network requirements for enhanced mobile broadband (eMBB), ultra reliable low latency communications (URLLC), and massive machine type communications (mMTC). Compared with LTE, 5G NR introduces wider frequency bands including Sub 6 GHz and millimeter wave, adjustable subcarrier spacings from 15 to 240 kHz, and a flexible time and frequency resource grid to adapt to diverse usage scenarios. To maintain high data rates and reduce encoding and decoding latency at millimeter wave frequencies, 5G NR adopted LDPC as its main channel coding scheme [30], effectively supporting real time communication performance between base stations and user equipment under high data volumes and stringent latency requirements.

5G NR employs quasi-cyclic LDPC codes that are derived from a protograph. The design begins with a small base graph $\mathbf{B}$ specifying the connections between variable and check nodes, and a lifting step expands this base graph into the working parity-check matrix $\mathbf{H}$ [30]. In

lifting, each entry of $\mathbf{B}$ is replaced by a $Z \times Z$ circulant permutation matrix or by a zero matrix, changing the lifting size Z scales the block length while preserving the topology, which yields a regular, hardware-friendly structure.

The standard defines two fixed base graphs, BG1 and BG2 [28]. After choosing a base graph and a lifting size Z, the matrix $\mathbf{H}$ is generated according to a table of shift indices. With a suitable permutation, $\mathbf{H}$ takes an approximately lower-triangular block form, the form of $\mathbf{H}$ is as shown in fig. 2.9 [30].

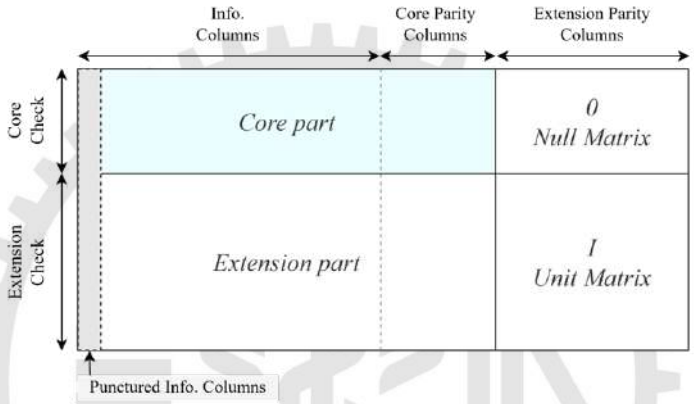


Figure 2.9: Parity check structure for the 5G NR LDPC code

The identity block in the lower right and the banded structure enable efficient back-substitution and shift-register based encoding, and they align well with layered belief-propagation decoding. In practice, BG1 is typically used for longer blocks and higher code rates, whereas BG2 targets shorter blocks and lower code rates. A common workflow is to select the base graph from the message length and target rate, and then select Z from the required block length and the desired parallelism.

This LDPC family is protograph-based, quasi-cyclic, irregular, and systematic. All codes are produced from the base graphs through lifting. A cyclic shift of a codeword remains a valid codeword due to the circulant submatrices, unequal column and row weights help approach capacity and a codeword $\mathbf{c} = (u_1, \ldots, u_K, p_1, \ldots, p_{N-K})$ contains both information and parity

bits.

After the LDPC encoder, rate matching is performed by a circular buffer together with redundancy versions, which provides multiple effective code rates and integrates naturally with HARQ as fig 2.10. To limit error propagation, the first $2Z$ dense information columns are punctured so that those positions are not transmitted and additional parity bits are sent instead, while the nominal rate $R_c = K/N$ is kept unchanged.

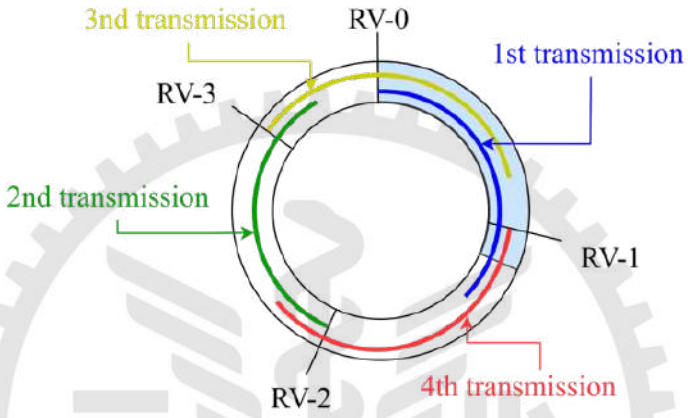


Figure 2.10: Circular buffer 5G IR-HARQ

2.6 Throughput

Throughput T under a HARQ protocol is defined as the average number of information bits successfully delivered per channel use. For an (N, K) code, where N denotes the codeword length (i.e., the number of channel uses per transmission) and K the number of information bits in each codeword, let M be the random variable representing the total number of HARQ transmissions (including initial transmission and any retransmissions) required for successful decoding. The average number of transmissions is $\mathbb{E}[M]$. Accordingly, throughput can be written as

$$T = \frac{\mathbb{E}[\text{successfully decoded bits}]}{\mathbb{E}[\text{channel uses}]} = \frac{K}{N\,\mathbb{E}[M]}. \tag{2.17}$$

Physically, the numerator K represents the fixed quantity of information bits delivered upon each successful HARQ cycle, while the denominator $N\,\mathbb{E}[M]$ captures the average total channel use cost required to achieve that success. A higher decoding failure probability raises $\mathbb{E}[M]$, which in turn reduces the overall throughput.

Throughput depends critically on system parameters. Increasing the codeword length N typically enhances error-correction capability but also increases the resource cost per HARQ round, creating a trade-off between reliability and efficiency. The code rate $R = K/N$ directly scales the numerator but can adversely affect $\mathbb{E}[M]$ if the initial decoding error rate grows. Similarly, allowing more retransmissions (larger $M_{\max}$) can lower the final failure probability at the expense of a larger average $\mathbb{E}[M]$, which may diminish throughput under poor channel conditions.

In practice, optimizing throughput involves balancing these factors. In high SNR environments where $\mathbb{E}[M] \approx 1$, selecting a higher code rate maximizes throughput. In contrast, under low SNR or severe fading, reducing the code rate to decrease $\mathbb{E}[M]$ can yield higher long-term throughput despite a lower nominal rate. System designers must therefore tailor N, K, and HARQ parameters to the operating channel conditions and application requirements.

Chapter 3.

Construction of Fully-Combined Structure Tanner Graph

3.1 Fully-Combined Tanner Graph Method

In the fully-combined tanner graph construction, selected payload and extra variable nodes are merged into a single transmitted bit by introducing, for each selected index, a new check node and a new variable node. Let $\mathcal{I}$ denote the payload VN indices chosen for puncturing algorithm. For each $i \in \mathcal{I}$, the parity-check matrix and graph are updated as follows:

Algorithm 2 Fully-Combined Tanner Graph Construction

Input: Parity-check matrix $\mathbf{H}$, index $\mathcal{I}$ of payload VNs
Output: Updated parity-check matrix $\mathbf{H}'$ representing the combined Tanner graph

$\mathbf{H}' \leftarrow \mathbf{H}$
for each position $i = 1, \ldots, c_2$ **do**
    Let $j = \mathcal{I}_i$ be the i-th puncture index
    Let p_j be the payload VN and e_i the corresponding extra VN
    Create a new variable node z_i that carries $p_j \oplus e_i$
    Create a new check node c_i enforcing

$$p_j \oplus e_i \oplus z_i = 0$$

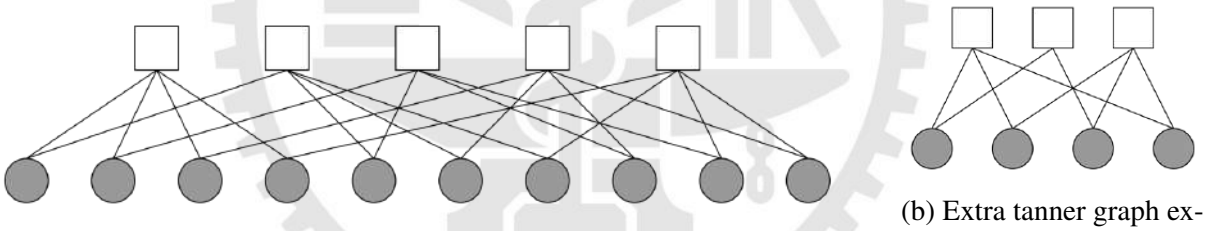
    Insert one row for c_i and one column for z_i into $\mathbf{H}'$
    Connect c_i to p_j, e_i, and z_i in the Tanner graph
end for
return $\mathbf{H}'$

Since the total number of transmitted bits must remain equal to the original payload length,

each payload – extra pair (p_j, e_i) for $i = 1, \ldots, c_2$, $j = \mathcal{I}_i$ is punctured (i.e. removed from transmission). Only the new variable nodes z_i are sent over the channel.

$$\mathbf{H}_{\text{payload}} = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix} \tag{3.1}$$

$$\mathbf{H}_{\text{extra}} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \tag{3.2}$$



(a) Payload tanner graph example

(b) Extra tanner graph example

Figure 3.1: Illustrative tanner graph examples of the payload and extra codes

Denote the original payload and extra parity-check matrices by

$$\mathbf{H}_{\text{payload}} \in \mathbb{F}_2^{r_1 \times c_1}, \quad \mathbf{H}_{\text{extra}} \in \mathbb{F}_2^{r_2 \times c_2}. \tag{3.3}$$

Illustrative examples of $\mathbf{H}_{\text{payload}}$ and $\mathbf{H}_{\text{extra}}$ are given in (3.1) and (3.2), respectively, while their corresponding tanner graph representations are shown in Fig. 3.1a and Fig. 3.1b.

$$\mathbf{H}_{\mathrm{FC}} = \left[\begin{array}{cccccccccc|cccc|cccc} 1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 1&0&0&0&1&1&1&0&0&0&0&0&0&0&0&0&0&0 \\ 0&1&0&0&1&0&0&1&1&0&0&0&0&0&0&0&0&0 \\ 0&0&1&0&0&1&0&1&0&1&0&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&1&0&1&1&0&0&0&0&0&0&0&0 \\ \hline 0&0&0&0&0&0&0&0&0&0&1&1&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&1&1&1&0&0&0&0 \\ \hline 1&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0 \\ 0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0 \\ 0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&1&0 \\ 0&0&0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&1 \end{array}\right] \tag{3.4}$$

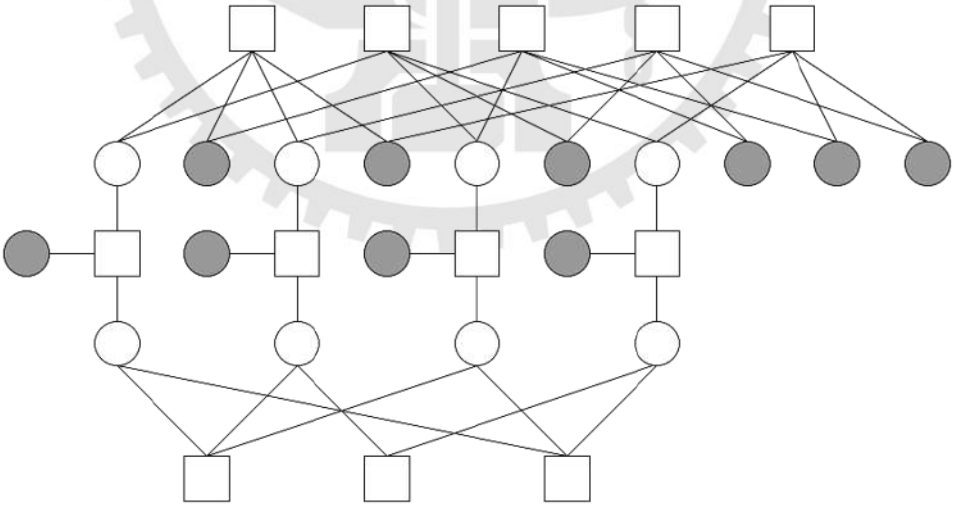


Figure 3.2: Illustrative fully-combined tanner graph example

The puncturing algorithm selects an index in $\boldsymbol{\mathcal{I}}$ and removes the corresponding columns from both $\mathbf{H}_{\mathrm{payload}}$ and $\mathbf{H}_{\mathrm{extra}}$. Each punctured payload-extra pair (p_j, e_i) for $i = 1, \ldots, c_2$,

$j = \mathcal{I}_i$ is then replaced by a new variable node z_i and a check node enforcing the XOR constraint

$$p_j \oplus e_i \oplus z_i = 0. \tag{3.5}$$

As a result, the fully-combined parity-check matrix $\mathbf{H}_{\mathrm{FC}}$ acquires the block structure illustrated by the tanner graph in Fig. 3.2, where white circles denote punctured variable nodes and black circles denote transmitted ones. An illustrative example of the corresponding parity-check matrix is given in (3.4).

The fully-combined structure tanner graph is translated into the following general form:

$$\mathbf{H}_{\mathrm{FC}} = \begin{pmatrix} \mathbf{H}_{\mathrm{payload}} & \mathbf{0}_{r_1 \times c_2} & \mathbf{0}_{r_1 \times c_2} \\ \mathbf{0}_{r_2 \times c_1} & \mathbf{H}_{\mathrm{extra}} & \mathbf{0}_{r_2 \times c_2} \\ \mathbf{M}_{\mathrm{puncpos}} & \mathbf{I}_{c_2} & \mathbf{I}_{c_2} \end{pmatrix} \in \mathbb{F}_2^{(r_1+r_2+c_2)\times(c_1+2c_2)} \tag{3.6}$$

This matrix satisfies the following structural properties:

- $\mathbf{M}_{\mathrm{puncpos}} \in \mathbb{F}_2^{c_2 \times c_1}$ is the puncture position matrix. By construction

$$\begin{cases} \sum_{j=1}^{c_1} \left[\mathbf{M}_{\mathrm{puncpos}}\right]_{i,j} = 1, & \forall\, i = 1, \ldots, c_2, \\ \sum_{i=1}^{c_2} \left[\mathbf{M}_{\mathrm{puncpos}}\right]_{i,j} \in \{0, 1\}, & \forall\, j = 1, \ldots, c_1. \end{cases} \tag{3.7}$$

  Hence each row has exactly one 1, and each column has at most one 1. Moreover, the columns containing a 1 correspond exactly to the payload VN indices in the puncture set $\mathcal{I}$.

- The identity blocks $\mathbf{I}_{c_2}$ link each extra VN and payload VN to its new combination VN z_i, thereby enforcing the XOR constraint through the bottom-right submatrix.

- All other zero blocks merely pad the matrix so that $\mathbf{H}_{\text{payload}}$ and $\mathbf{H}_{\text{extra}}$ remain in their original block positions.

This structure ensures that only those payload VNs in $\boldsymbol{\mathcal{I}}$ and their paired extra VNs appear in the lower puncture block, and that each new row (check node) touches exactly one punctured payload column, one extra column, and one combination column.

3.2 Encoding Scheme

After selecting the puncture index set $\boldsymbol{\mathcal{I}}$, the transmitter forms each transmitted bit by XOR the corresponding payload and extra bits. Concretely, let

$$\boldsymbol{p} = \big(p_1, p_2, \ldots, p_{c_1}\big), \quad \boldsymbol{e} = \big(e_1, e_2, \ldots, e_{c_2}\big) \tag{3.8}$$

Denote the original payload and extra codewords (with $c_2 = |\boldsymbol{\mathcal{I}}|$). For each puncture index $i = 1, \ldots, c_2$, $j = \mathcal{I}_i$, define the combined symbol

$$z_i = p_j \oplus e_i. \tag{3.9}$$

All other payload bits p_j with $p_j \notin \boldsymbol{\mathcal{I}}$ are transmitted unchanged, and extra bits e_i appear only through their contribution to z_i. Thus the transmitted codeword $\mathbf{x} \in \mathbb{F}_2^{c_1}$ is given componentwise by

$$x_j = \begin{cases} p_j \oplus e_i, & \text{if } p_j \in \boldsymbol{\mathcal{I}} \text{ and } i \text{ is such that } \mathcal{I}_i = j, \\ p_j, & \text{otherwise.} \end{cases} \tag{3.10}$$

This ensures $|\mathbf{x}| = c_1$, i.e. the transmitted length matches the original payload length.

In practice as Fig. 3.3, the transmitter first computes the standard LDPC encoder output

Algorithm 3 Encoding with XOR Superposition

Input: payload codeword $\mathbf{p} \in \mathbb{F}_2^{c_1}$, extra codeword $\mathbf{e} \in \mathbb{F}_2^{c_2}$, puncture index $\boldsymbol{\mathcal{I}} \subseteq \{1, \ldots, c_1\}$
Output: Transmitted codeword $\mathbf{x} \in \mathbb{F}_2^{c_1}$

1: Copy payload into $\mathbf{x}$: $\mathbf{x} \leftarrow \mathbf{p}$
2: **for** each position $i = 1, \ldots, c_2$ **do**
3: Let $j = \mathcal{I}_i$ be the i-th puncture index
4: Update the transmitted bit: $x_j \leftarrow p_j \oplus e_i$
5: **end for**
6: Transmit $\mathbf{x}$ over the channel

$\boldsymbol{p}$, and then applies the simple element-wise XOR with the extra codeword $\boldsymbol{e}$ at the puncture positions. No modification of the LDPC encoder or channel modulator is required—only the mapping from $(\boldsymbol{p}, \boldsymbol{e})$ to the transmitted word $\boldsymbol{x}$ changes. This guarantees that the total number of transmitted bits remains exactly c_1, matching the original payload length.

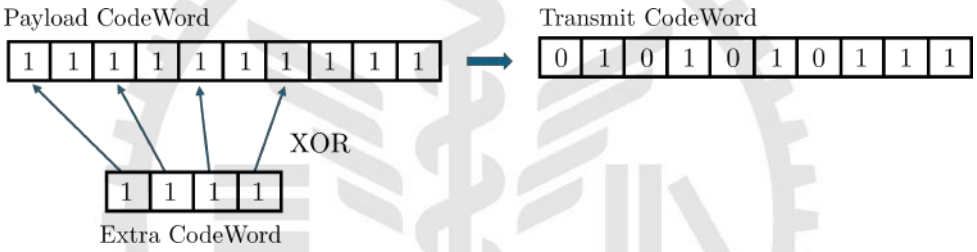


Figure 3.3: Fully-combined - Encoding scheme

3.3 Minimum Distance

Bounds on the minimum distance of $\mathbf{H}_{\mathrm{FC}}$:

Let

$$d_{\min}(\mathbf{H}_{\mathrm{payload}}) = M, \quad d_{\min}(\mathbf{H}_{\mathrm{extra}}) = N, \tag{3.11}$$

To determine the minimum distance of $\mathbf{H}_{\mathrm{FC}}$ 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 $\mathbf{H}_{\mathrm{payload}}$ portion.

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

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

$$\mathcal{C}_{\text{payload}} = \left\{ \boldsymbol{c} \in \{0,1\}^{c_1} \;\middle|\; \boldsymbol{c}\,\mathbf{H}_{\text{payload}}^{\mathbf{T}} = \vec{0},\ \text{wt}(\boldsymbol{c}) = M \right\}, \tag{3.12}$$

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

$$S = \left\{ j \in \{1, 2, \ldots, c_1\} \;\middle|\; \exists\, \boldsymbol{c} \in \mathcal{C}_{\text{payload}} :\ c_j = 1 \right\}, \tag{3.13}$$

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

Define

$$\mathcal{P} = \left\{ j : \text{column } j \text{ of } \mathbf{M}_{\text{puncpos}} \text{ has degree } 1 \right\}, \tag{3.14}$$

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

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

$$\begin{pmatrix} \mathbf{H}_{\text{payload}} & \mathbf{0}_{\mathbf{r_1} \times \mathbf{c_2}} \\ \mathbf{M}_{\text{puncpos}} & \mathbf{I}_{\mathbf{c_2}} \end{pmatrix},$$

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

$$d_{\min}^{\mathrm{tx}(1)} = M + k \tag{3.15}$$

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

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

$$\mathcal{C}_{\mathrm{extra}} = \left\{ \boldsymbol{c} \in \{0,1\}^{c_2} \;\middle|\; \boldsymbol{c}\,\mathbf{H}_{\mathrm{extra}}^{\mathbf{T}} = \vec{0}, \quad \mathrm{wt}(\boldsymbol{c}) = N \right\}, \tag{3.16}$$

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

$$T = \left\{ j \in \{1, 2, \ldots, c_2\} \;\middle|\; \exists\, \boldsymbol{c} \in \mathcal{C}_{\mathrm{extra}} :\; c_j = 1 \right\}, \tag{3.17}$$

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

Observe that in $\mathbf{H}_{\mathrm{FC}}$ these columns appear in the block

$$\begin{pmatrix} \mathbf{H}_{\mathrm{extra}} & \mathbf{0}_{\mathbf{r_2} \times \mathbf{c_2}} \\ \mathbf{I}_{\mathbf{c_2}} & \mathbf{I}_{\mathbf{c_2}} \end{pmatrix},$$

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

Selecting any column $j \in T$ from the $\mathbf{H}_{\mathrm{extra}}$ portion flips the j-th row of the lower identity block from 0 to 1. To restore that col-sum to zero, selecting 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.

$$d_{\min}^{\mathrm{tx}(2)} = 2N \tag{3.18}$$

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,

$$d_{\min}(\mathbf{H}_{\mathrm{FC}}) \geq \min(M + k,\ 2N) = \min(d_{\min}^{\mathrm{tx}(1)}, d_{\min}^{\mathrm{tx}(2)}). \tag{3.19}$$

3.4 Partial Structure

The partial structure splits the extra variable nodes into two disjoint subsets:

$$\mathcal{E}_{\mathrm{tx}} = \{\, i \in \{1, \ldots, c_2\} \mid \text{extra VN } e_i \text{ transmitted directly}\}, \tag{3.20}$$

$$\mathcal{E}_{\mathrm{merge}} = \{1, \ldots, c_2\} \setminus \mathcal{E}_{\mathrm{tx}}. \tag{3.21}$$

Each e_i with $i \in \mathcal{E}_{\mathrm{tx}}$ is sent over the channel as a standalone variable node. No merging or additional check nodes are introduced for these bits.

For each $j \in \mathcal{E}_{\mathrm{merge}}$, the extra VN e_j is punctured and merged with its corresponding payload VN p_j via a new combination VN z_j. A new check node c_j enforces the parity constraint

$$p_j \oplus e_j \oplus z_j = 0. \tag{3.22}$$

Only z_j is transmitted; both p_j and e_j are removed from the channel output.

Since the total transmission length remains fixed at the original payload size, sending extra bits requires puncturing an equal number of payload bits. As a result, payload decoding performance incurs a slight degradation.

$$\mathbf{H}_{\text{Partial}} = \left[\begin{array}{cccccccccc|cccc|cc} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{array}\right]$$

Figure 3.4: Partial structure parity-check matrix example

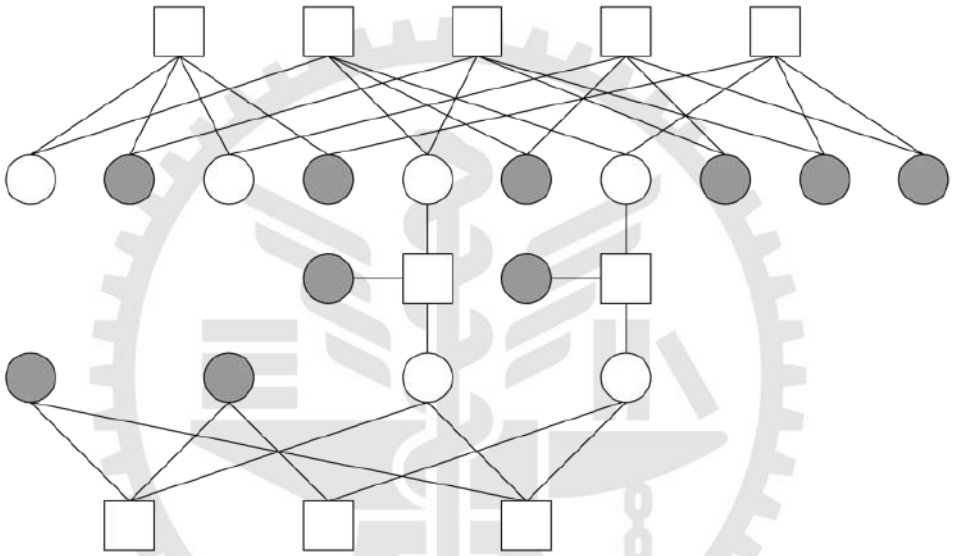


Figure 3.5: Illustrative partial structure tanner graph example

The partial tanner graph structure is illustrated in Figure 3.5, where white circles denote punctured variable nodes and black circles denote transmitted ones. The corresponding parity-check matrix representation derived from this tanner graph structure is shown in Figure 3.4.

The partial structure parity-check matrix following general form

$$\mathbf{H}_{\text{Partial}} = \begin{pmatrix} \mathbf{H}_{\text{payload}} & \mathbf{0}_{r_1 \times c_2} & \mathbf{0}_{r_1 \times c_2} \\ \mathbf{0}_{r_1 \times c_1} & \mathbf{H}_{\text{extra}} & \mathbf{0}_{r_1 \times c_2} \\ \mathbf{M}_{\text{payloadpunc}} & \mathbf{M}_{\text{extrapunc}} & \mathbf{I}_{|\mathcal{E}_{\text{merge}|}} \end{pmatrix}, \tag{3.23}$$

where

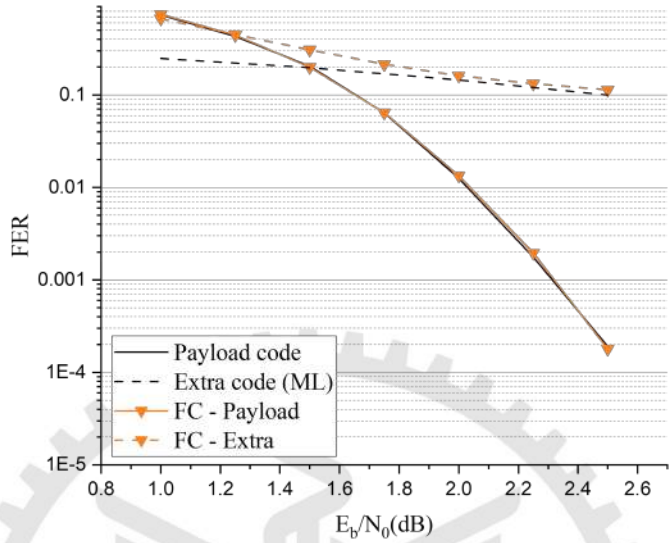
- $M_{\text{payloadpunc}} \in \mathbb{F}_2^{|\mathcal{E}_{\text{merge}}| \times c_1}$ has exactly one 1 in each row, where the 1 in row i marks the payload VN punctured in the ith merge.
- $M_{\text{extrapunc}} \in \mathbb{F}_2^{|\mathcal{E}_{\text{merge}}| \times |c_2|}$ has exactly one 1 in each row, where the 1 in row i marks the extra VN merged in the ith operation.
- $I \in \mathbb{F}_2^{|\mathcal{E}_{\text{merge}}| \times |\mathcal{E}_{\text{merge}}|}$ is the identity matrix, corresponding to the new combined VNs z_i.

3.5 Simulation Result

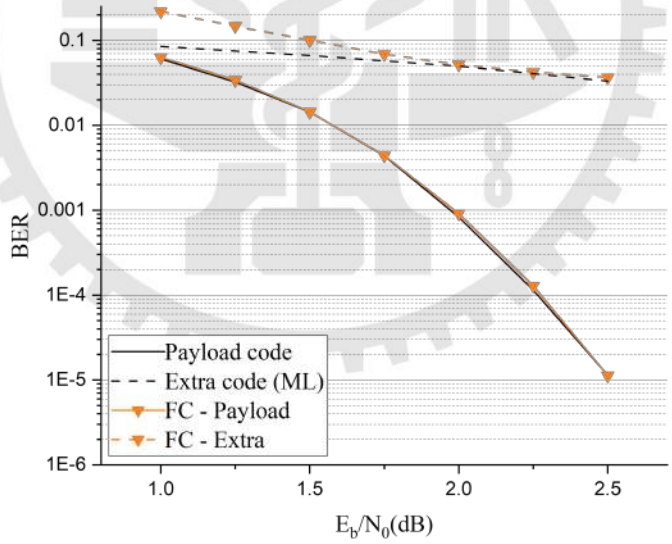
Table 3.1: Simulation Code Information

Code	Category	Codeword Length	Information Length	Code Rate
PEG LDPC	Payload	1008	504	0.5
LDPC	Extra	10	5	0.5
BCH	Extra	15	7	0.47

This simulation uses PEG-LDPC (1008, 504) for the payload combined with two extra codes: LDPC (10, 5) and BCH (15, 7). For the BP decoding process, the maximum number of iterations is set to 200. Under BPSK modulation over an AWGN channel, Fig. 3.6 presents both the frame error rate (FER) and bit error rate (BER) performance of the LDPC extra code, showing that it is slightly worse at low E_b/N_0 but quickly converges toward ML performance as E_b/N_0 increases. Likewise, Fig. 3.7 demonstrates the same FER and BER trends for the BCH extra code. Meanwhile, the payload BER and FER under the fully-combined structure remain virtually identical to those of pure PEG-LDPC, indicating no loss in payload performance.

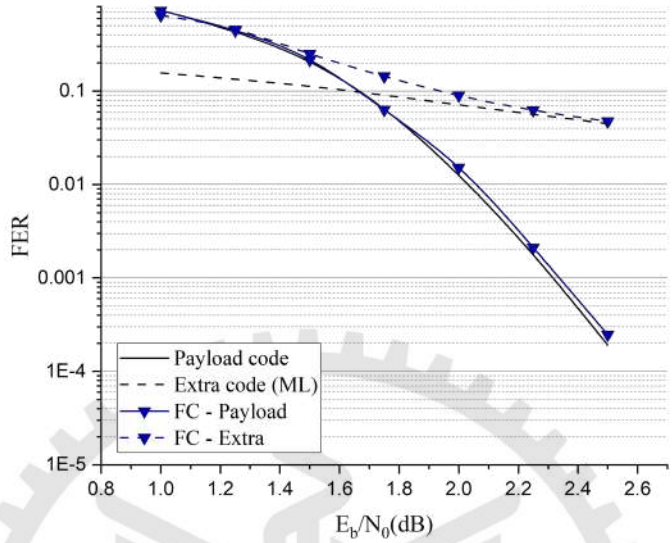


(a) FER

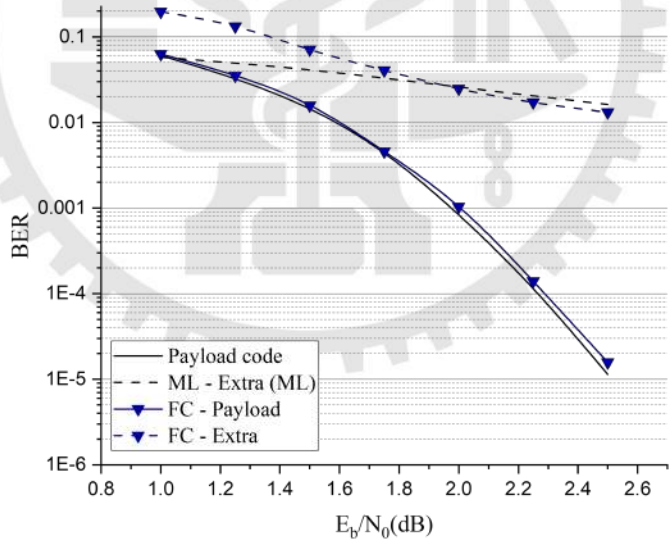


(b) BER

Figure 3.6: Performance of the fully-combined structure with payload H(1008, 504) and extra H(10, 5)



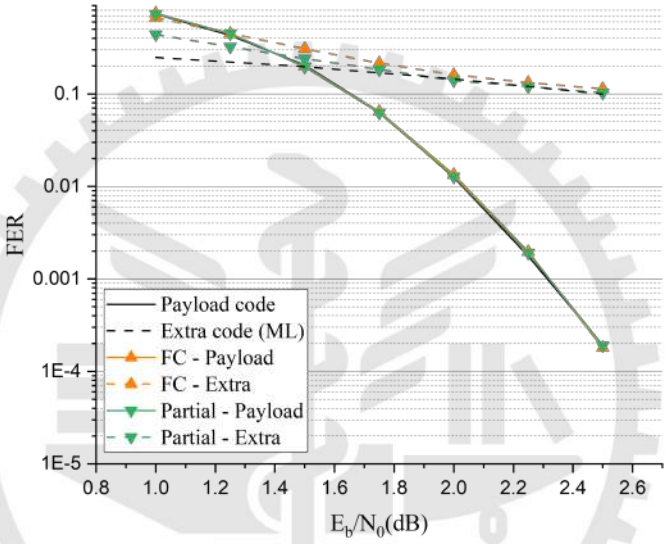
(a) FER



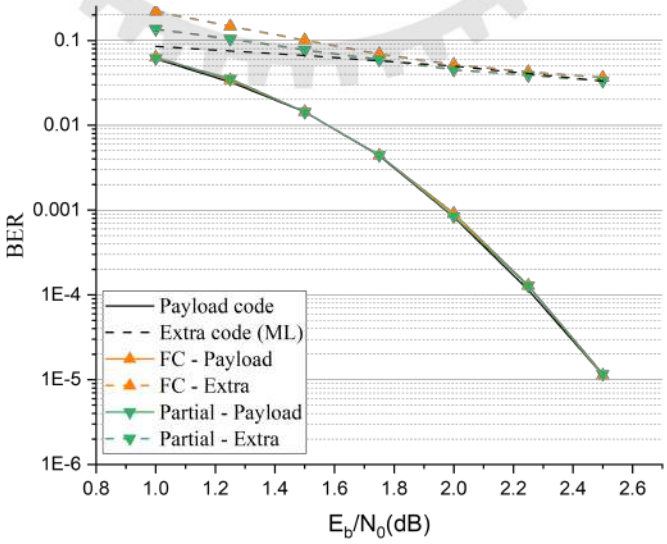
(b) BER

Figure 3.7: Performance of the fully-combined structure with payload H(1008, 504) and extra BCH(15, 7)

In the partial structure, to transmit additional extra bits within the fixed payload length, an equal number of payload bits are punctured and the remaining extra bits are merged with their corresponding payload bits into new symbols z_j. As shown in Fig. 3.8 and Fig. 3.9, this approach introduces only a slight FER、BER degradation for the payload, while in the low E_b/N_o region the directly transmitted extra bits yield a substantial improvement in error-correction performance; as E_b/N_o increases, the extra BER quickly converges to near optimal ML decoding levels.

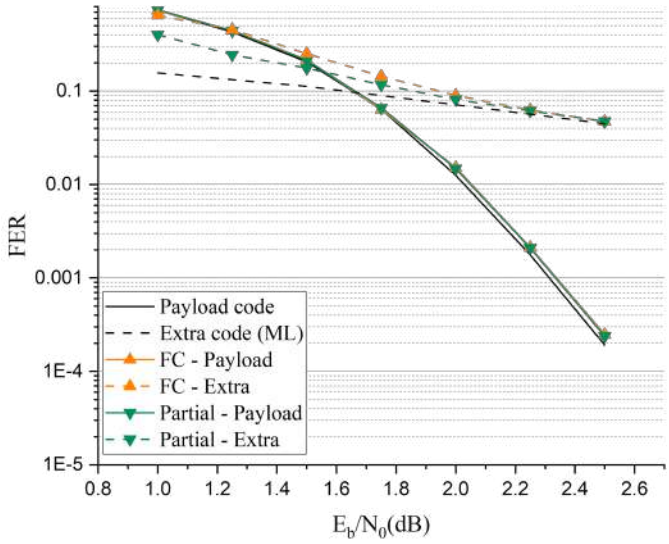


(a) FER

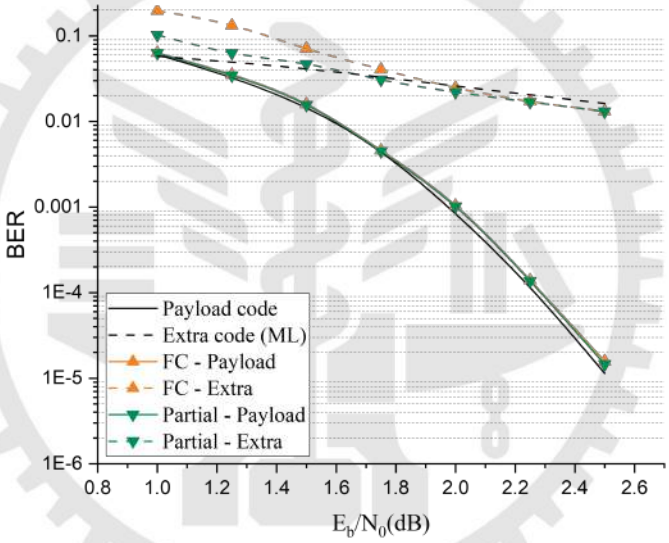


(b) BER

Figure 3.8: Performance of the partial structure with payload H(1008, 504) and extra H(10, 5)



(a) FER



(b) BER

Figure 3.9: Performance of the partial structure with payload H(1008, 504) and extra BCH(15, 7)

Chapter 4.

Construction of the Enhanced Combined Tanner Graph

4.1 Enhanced-Combined Tanner Graph Method

The enhanced structure builds upon the fully-combined method by introducing a two-stage XOR-based merging strategy. In the first stage, each extra bit is XORed with an unmerged payload bit to form a first-level merge node. In the second stage, this first-level merge node is further XORed with another unmerged payload bit to generate a second-level merge node. Through this progressive merging process, each extra bit is redundantly connected to two distinct payload bits, which provides richer feedback during iterative decoding and significantly enhances the error-correction performance of the extra bits.

The adoption of a two-stage XOR merging strategy is primarily motivated by the need to enhance information flow during belief propagation decoding. In the first-stage XOR, each extra bit is introduced into the tanner graph by coupling it with a payload bit, thereby establishing a fundamental dependency between the payload and extra layers. However, a single XOR connection may lead to limited message diversity and insufficient feedback paths during iterative decoding. To address this limitation, a second-stage XOR is applied by further connecting the intermediate merge node to another payload bit. This additional connection creates alterna-

tive message-passing routes between the payload and extra bits, effectively increasing graph connectivity and improving the exchange of soft information. As a result, the two-stage XOR structure mitigates the formation of short cycles and reduces the risk of trapping sets, leading to faster convergence and more reliable decoding of the extra bits.

Prior to the enhanced-combined tanner graph construction, the payload index sets $\mathcal{I}_1$, $\mathcal{I}_2$, and $\mathcal{J}$ are deterministically identified based on the payload tanner graph structure. Specifically, $\mathcal{I}_1$ and $\mathcal{I}_2$ are determined by a puncturing algorithm applied to the payload variable nodes, where the selected nodes are assigned to be punctured at different merging stages. The index set $\mathcal{J}$ is selected from the remaining unpunctured payload variable nodes and is constrained to exclude nodes that are adjacent to those in $\mathcal{I}_2$. This constraint is imposed to prevent the formation of short cycles, particularly length-6 cycles, in the enhanced tanner graph. As a result, the resulting index sets enable a structured merging process with explicit cycle avoidance, rather than a random selection of payload bits.

The enhanced-combined tanner graph construction procedure is summarized in Algorithm 4.

Algorithm 4 Enhanced-Combined Tanner Graph Construction

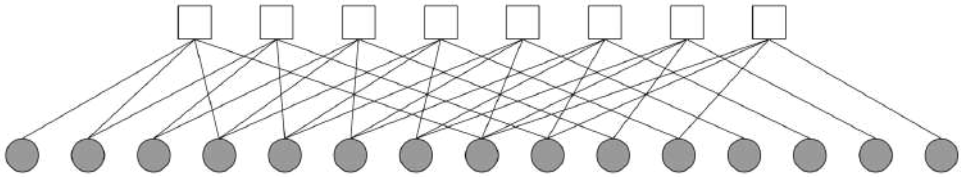
Input: Parity-check matrix $\mathbf{H}$, disjoint payload index sets $\boldsymbol{\mathcal{I}_1}, \boldsymbol{\mathcal{I}_2}, \boldsymbol{\mathcal{J}}$
Output: Enhanced parity-check matrix $\mathbf{H}_{\text{Enhanced}}$

$\mathbf{H}_{\text{Enhanced}} \leftarrow \mathbf{H}$
for each $i = 1, \ldots, c_2$ **do** # Stage 1: original merge (payload bits in $\boldsymbol{\mathcal{I}_1}$ will be punctured)
    Let $j = \mathcal{I}_{1,i}$ be the i-th puncture index
    Create variable node z_i and check node c_i
    Enforce $p_j \oplus e_i \oplus z_i = 0$
    Insert row for c_i and column for z_i into $\mathbf{H}_{\text{Enhanced}}$
    Connect c_i to p_j, e_i, and z_i
    Puncture payload VN p_j
end for
for each $i = 1, \ldots, c_2$ **do** # Stage 2: intermediate merge (payload bits in $\boldsymbol{\mathcal{J}}$ not punctured)
    Let $j = \mathcal{J}_i$ be the i-th puncture index
    Create variable node y_i and check node d_i
    Enforce $y_i \oplus p_j \oplus e_i = 0$
    Insert row for d_i and column for y_i into $\mathbf{H}_{\text{Enhanced}}$
    Connect d_i to y_i, p_j, and e_i
end for
for each $i = 1, \ldots, c_2$ **do** # Stage 3: final merge (payload bits in $\boldsymbol{\mathcal{I}_2}$ will be punctured)
    Let $j = \mathcal{I}_{2,i}$ be the i-th puncture index
    Create variable node x_i and check node f_i
    Enforce $y_i \oplus p_j \oplus x_i = 0$
    Insert row for f_i and column for x_i into $\mathbf{H}_{\text{Enhanced}}$
    Connect f_i to y_i, p_j, and x_i
    Puncture payload VN p_j
end for
return $\mathbf{H}_{\text{Enhanced}}$

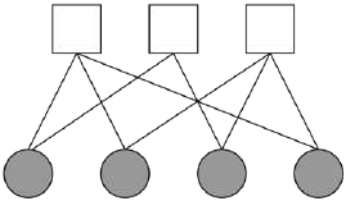
To guarantee that the final transmission length matches the original payload length, puncturing is applied in three steps. First, all extra bits are punctured. Next, in the first stage , which corresponds to the original merge, each payload node involved in the first merge is punctured. Then, in the second stage, each newly created intermediate merge node is punctured. Finally, in the third stage, each payload node involved in the second merge is punctured. After these staged punctures, only the most recently added merge nodes from each stage remain for transmission, yielding exactly the original payload bit count.

$$\mathbf{H}_{\text{payload}} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.1}$$

$$\mathbf{H}_{\text{extra}} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \tag{4.2}$$



(a) Payload tanner graph example



(b) Extra tanner graph example

Figure 4.1: Illustrative tanner graph examples of the payload and extra codes

Illustrative examples of the payload and extra parity-check matrices are given in (4.1) and (4.2), respectively, while their corresponding tanner graph representations are shown in Fig. 4.1a and Fig. 4.1b.

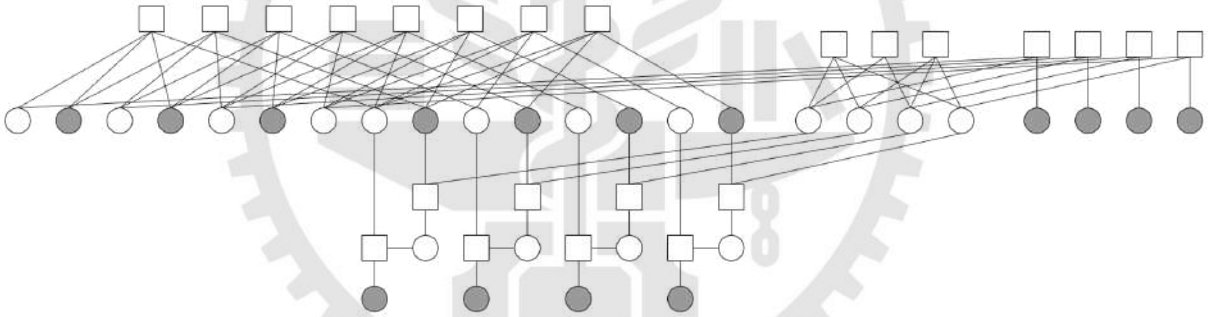


Figure 4.2: Enhanced-combine tanner graph with payload and extra example

$$
\mathbf{H}_{\text{Enhanced}} = \left[\begin{array}{cccccccccccccccc|cccc|cccc|cccc}
1&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
\hline
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&0&0&0&0&0&0&0&0\\
\hline
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0\\
\hline
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&0&1\\
\hline
0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0\\
0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1
\end{array}\right]
\tag{4.3}
$$

As a result, the combined parity-check matrix $\mathbf{H}_{\text{Enhanced}}$ acquires the block structure illustrated by the tanner graph in Fig. 4.2. In that figure, white circles denote punctured variable nodes, whereas black circles denote transmitted ones. An illustrative example of the corresponding parity-check matrix is given in (4.3).

As illustrated in Fig. 4.2, the enhanced tanner graph is translated into the following general form:

$$\mathbf{H}_{\text{Enhanced}} = \begin{pmatrix} \mathbf{H}_{\text{payload}} & \mathbf{0}_{r_1\times c_2} & \mathbf{0}_{r_1\times c_2} & \mathbf{0}_{r_1\times c_2} & \mathbf{0}_{r_1\times c_2} \\ \mathbf{0}_{r_2\times c_1} & \mathbf{H}_{\text{extra}} & \mathbf{0}_{r_2\times c_2} & \mathbf{0}_{r_2\times c_2} & \mathbf{0}_{r_2\times c_2} \\ \mathbf{M}_{\text{puncpos}} & \mathbf{I}_{c_2} & \mathbf{I}_{c_2} & \mathbf{0}_{c_2\times c_2} & \mathbf{0}_{c_2\times c_2} \\ \mathbf{M}_{\text{Enhanced1}} & \mathbf{I}_{c_2} & \mathbf{0}_{c_2\times c_2} & \mathbf{I}_{c_2} & \mathbf{0}_{c_2\times c_2} \\ \mathbf{M}_{\text{Enhanced2}} & \mathbf{0}_{c_2\times c_2} & \mathbf{0}_{c_2\times c_2} & \mathbf{I}_{c_2} & \mathbf{I}_{c_2} \end{pmatrix} \in \mathbb{F}_2^{(r_1+r_2+3c_2)\times(c_1+4c_2)} \tag{4.4}$$

Here:

- $\mathbf{H}_{\text{payload}} \in \mathbb{F}_2^{r_1\times c_1}$ and $\mathbf{H}_{\text{extra}} \in \mathbb{F}_2^{r_2\times c_2}$ are the original payload and extra submatrices.
- $\mathbf{M}_{\text{puncpos}} \in \mathbb{F}_2^{c_2\times c_1}$ is the puncture position matrix. By construction

$$\begin{cases} \sum_{j=1}^{c_1} \left[\mathbf{M}_{\text{puncpos}}\right]_{i,j} = 1, & \forall\, i = 1,\ldots,c_2, \\ \sum_{i=1}^{c_2} \left[\mathbf{M}_{\text{puncpos}}\right]_{i,j} \in \{0,1\}, & \forall\, j = 1,\ldots,c_1. \end{cases} \tag{4.5}$$

  Hence each row has exactly one 1, and each column has at most one 1. Moreover, the columns containing a 1 correspond exactly to the payload VN indices in $\mathcal{I}_1$.

- $\mathbf{M}_{\text{Enhanced1}} \in \mathbb{F}_2^{c_2 \times c_1}$ is the first enhancement merge position matrix. By construction

$$\begin{cases} \sum_{j=1}^{c_1} \left[\mathbf{M}_{\text{Enhanced1}}\right]_{i,j} = 1, & \forall\, i = 1, \ldots, c_2, \\ \sum_{i=1}^{c_2} \left[\mathbf{M}_{\text{Enhanced1}}\right]_{i,j} \in \{0,1\}, & \forall\, j = 1, \ldots, c_1. \end{cases} \tag{4.6}$$

  Hence each row has exactly one 1, each column at most one 1. Moreover, the columns containing a 1 correspond exactly to the payload VN indices in $\boldsymbol{\mathcal{J}}$.

- $\mathbf{M}_{\text{Enhanced2}} \in \mathbb{F}_2^{c_2 \times c_1}$ is the second enhancement merge position matrix. By construction

$$\begin{cases} \sum_{j=1}^{c_1} \left[\mathbf{M}_{\text{Enhanced2}}\right]_{i,j} = 1 & \forall\, i = 1, \ldots, c_2, \\ \sum_{i=1}^{c_2} \left[\mathbf{M}_{\text{Enhanced2}}\right]_{i,j} \in \{0,1\} & \forall\, j = 1, \ldots, c_1. \end{cases} \tag{4.7}$$

  Thus each row has exactly one 1, each column at most one 1. Moreover, the columns containing a 1 correspond exactly to the payload VN indices in $\boldsymbol{\mathcal{I}_2}$.

- $\mathbf{I}_{c_2} \in \mathbb{F}_2^{c_2 \times c_2}$ is the stage-link identity matrix. As an identity matrix, it has exactly one 1 per row and per column, linking merge VNs across stages and enforcing the XOR parity constraints between consecutive merge layers.

4.2 Encoding Scheme

In practice, the transmitter first computes the standard LDPC encoder output

$$\boldsymbol{p} = (p_1, p_2, \ldots, p_{c_1}) \in \mathbb{F}_2^{c_1}, \tag{4.8}$$

and holds the extra bits

$$\boldsymbol{e} = (e_1, e_2, \ldots, e_{c_2}) \in \mathbb{F}_2^{c_2}. \tag{4.9}$$

The enhanced XOR merge encoding then proceeds in two simple loops (Algorithm 5):

Algorithm 5 Enhanced XOR Merge Encoding

Input: Payload $\boldsymbol{p} \in \mathbb{F}_2^{c_1}$, Extra $\boldsymbol{e} \in \mathbb{F}_2^{c_2}$, index sequences $\boldsymbol{\mathcal{I}_1}, \boldsymbol{\mathcal{J}}, \boldsymbol{\mathcal{I}_2} \subseteq \{1, \ldots, c_1\}$
Output: Transmitted codeword $\boldsymbol{x} \in \mathbb{F}_2^{c_1}$

Initialize $\boldsymbol{x} \leftarrow \boldsymbol{p}$ # copy payload into output
for $i = 1, \ldots, c_2$ **do** # Stage 1: Origin merge
    $j \leftarrow \mathcal{I}_{1,i}$
    $x_j \leftarrow p_j \oplus e_i$
end for
for $i = 1, \ldots, c_2$ **do** # Stage 2: Enhanced merge
    $l \leftarrow \mathcal{J}_i$
    $k \leftarrow \mathcal{I}_{2,i}$
    $x_k \leftarrow p_l \oplus p_k \oplus e_i$
end for

In the first stage, for each $i = 1, \ldots, c_2$ the payload bit at position $j = \mathcal{I}_{1,i}$ is overwritten with the XOR of its original value and the extra bit e_i. Concretely,

$$x_j = p_j \oplus e_i, \tag{4.10}$$

In the second stage, each extra bit e_i is further combined with two distinct payload bits. Denote by $\ell = \mathcal{J}_i$ and $k = \mathcal{I}_{2,i}$ the two payload positions for the i-th merge. The final transmitted bit at position k is then computed as

$$x_k = p_\ell \oplus p_k \oplus e_i, \tag{4.11}$$

so that the extra information is seamlessly embedded into the payload without changing the overall codeword length.

All other positions $m \notin \{\mathcal{I}_{1,i}, \mathcal{I}_{2,i} \mid i = 1, \ldots, c_2\}$ remain as $x_m = p_m$. No changes are

made to the LDPC encoder or the channel modulator. Since exactly two payload bits are overwritten per extra bit and no new positions are introduced, the total transmitted length remains

$$|\boldsymbol{x}| \;=\; c_1, \tag{4.12}$$

identical to the original payload length.

4.3 Minimum Distance

Bounds on the minimum distance of $\mathbf{H}_{\text{Enhanced}}$:

Assume

$$d_{\min}(\mathbf{H}_{\text{payload}}) = M, \quad d_{\min}(\mathbf{H}_{\text{extra}}) = N. \tag{4.13}$$

Moreover, let

$$\mathbf{H}_{\text{Enhanced}} = \begin{pmatrix} \mathbf{H}_{\text{payload}} & \mathbf{0}_{r_1\times c_2} & \mathbf{0}_{r_1\times c_2} & \mathbf{0}_{r_1\times c_2} & \mathbf{0}_{r_1\times c_2} \\ \mathbf{0}_{r_2\times c_1} & \mathbf{H}_{\text{extra}} & \mathbf{0}_{r_2\times c_2} & \mathbf{0}_{r_2\times c_2} & \mathbf{0}_{r_2\times c_2} \\ \mathbf{M}_{\text{paypuncpos}} & \mathbf{I}_{c_2} & \mathbf{I}_{c_2} & \mathbf{0}_{c_2\times c_2} & \mathbf{0}_{c_2\times c_2} \\ \mathbf{M}_{\text{Enhanced1}} & \mathbf{I}_{c_2} & \mathbf{0}_{c_2\times c_2} & \mathbf{I}_{c_2} & \mathbf{0}_{c_2\times c_2} \\ \mathbf{M}_{\text{Enhanced2}} & \mathbf{0}_{c_2\times c_2} & \mathbf{0}_{c_2\times c_2} & \mathbf{I}_{c_2} & \mathbf{I}_{c_2} \end{pmatrix}, \tag{4.14}$$

where $\mathbf{M}_{\text{puncpos}}, \mathbf{M}_{\text{Enhanced1}}, \mathbf{M}_{\text{Enhanced2}} \in \mathbb{F}_2^{c_2\times c_1}$ each have exactly one 1 per row and at most

one 1 per column. Equivalently, if

$$\mathbf{P}' = \begin{pmatrix} \mathbf{0}_{r_2 \times c_1} \\ \mathbf{M}_{\text{puncpos}} \\ \mathbf{M}_{\text{Enhanced1}} \\ \mathbf{M}_{\text{Enhanced2}} \end{pmatrix}, \tag{4.15}$$

then each row of $\mathbf{P}'$ has degree 1 and each column degree ≤ 1.

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

$$\begin{pmatrix} \mathbf{H}_{\text{payload}} & \mathbf{0}_{r_1 \times c_2} & \mathbf{0}_{r_1 \times c_2} & \mathbf{0}_{r_1 \times c_2} \\ \mathbf{M}_{\text{puncpos}} & \mathbf{I}_{c_2} & \mathbf{0}_{c_2 \times c_2} & \mathbf{0}_{c_2 \times c_2} \\ \mathbf{M}_{\text{Enhanced1}} & \mathbf{0}_{c_2 \times c_2} & \mathbf{I}_{c_2} & \mathbf{0}_{c_2 \times c_2} \\ \mathbf{M}_{\text{Enhanced2}} & \mathbf{0}_{c_2 \times c_2} & \mathbf{I}_{c_2} & \mathbf{I}_{c_2} \end{pmatrix}.$$

The minimum distance $d_{\min}(\mathbf{H}_{\text{Enhanced}})$ may be influenced by two types of enhancement rows:

- rows arising from $\mathbf{M}_{\text{puncpos}}$
- rows arising from $\mathbf{M}_{\text{Enhanced1}}$ and $\mathbf{M}_{\text{Enhanced2}}$.

Consider the $\mathbf{M}_{\text{puncpos}}$ case with submatrix

$$\begin{pmatrix} \mathbf{H}_{\text{payload}} & \mathbf{0}_{\mathbf{r_1}\times\mathbf{c_2}} \\ \mathbf{M}_{\text{puncpos}} & \mathbf{I}_{\mathbf{c_2}} \end{pmatrix}.$$

Let

$$\mathcal{C}_{\text{payload}} = \left\{ \boldsymbol{c} \in \{0,1\}^{c_1} \;\middle|\; \boldsymbol{c}\,\mathbf{H}_{\text{payload}}^{\mathbf{T}} = \vec{0},\ \mathrm{wt}(\boldsymbol{c}) = M \right\}, \tag{4.16}$$

and define

$$S = \left\{ j \in \{1,\ldots,c_1\} \mid \exists\, \boldsymbol{c} \in \mathcal{C}_{\text{payload}} :\ c_j = 1 \right\}. \tag{4.17}$$

Moreover, let

$$\mathcal{P}_p = \left\{ j : \text{column } j \text{ of } M_{\text{puncpos}} \text{ has degree } 1 \right\}, \quad k_p = |S \cap \mathcal{P}_p|. \tag{4.18}$$

Then selecting any $j \in S \cap \mathcal{P}_p$ flips the jth row of the lower identity block from 0 to 1, requiring an additional column j of the identity block to cancel it.

Consider the $\mathbf{M}_{\text{Enhanced1}}$ and $\mathbf{M}_{\text{Enhanced2}}$ case with submatrix

$$\begin{pmatrix} \mathbf{H}_{\text{payload}} & \mathbf{0}_{\mathbf{r_1}\times\mathbf{c_2}} & \mathbf{0}_{\mathbf{r_1}\times\mathbf{c_2}} \\ \mathbf{M}_{\text{Enhanced1}} & \mathbf{I}_{\mathbf{c_2}} & \mathbf{0}_{\mathbf{c_2}\times\mathbf{c_2}} \\ \mathbf{M}_{\text{Enhanced2}} & \mathbf{I}_{\mathbf{c_2}} & \mathbf{I}_{\mathbf{c_2}} \end{pmatrix}.$$

Let

$$\mathcal{R}_{e1} = \left\{ i \;\middle|\; \exists\, j \in S :\ (\mathbf{M}_{\text{Enhanced1}})_{i,j} = 1 \right\}. \quad k_{e1} = \left|\mathcal{R}_{e1}\right|. \tag{4.19}$$

Whenever a payload column j lies in $\mathcal{R}_{e1}$, it flips one row of $\mathbf{M}_{\text{Enhanced1}}$ and triggers a secondary flip below. Consequently, each such row i forces two extra identity columns to restore

both flips, resulting in a contribution of $2k_{e1}$.

Let

$$\mathcal{R}_{e2} = \big\{\, i \;\big|\; \exists\, j \in S :\ (\mathbf{M}_{\text{Enhanced2}})_{i,j} = 1 \ \wedge\ i \notin \mathcal{R}_{e1} \big\}, \quad k_{e2} = \big|\mathcal{R}_{e2}\big|. \tag{4.20}$$

Similarly, whenever a payload column j lies in $\mathcal{R}_{e2}$ (but not in $\mathcal{R}_{e1}$), it flips exactly one $\mathbf{M}_{\text{Enhanced2}}$ row and does not trigger any further flips. Each such row i therefore requires one identity column to cancel that flip. Thus these flips contribute a total of k_{e2} extra columns.

Combining the contributions from both types of enhancement rows yields

$$d_{\min}^{tx(1)}(\mathbf{H}_{\text{Enhanced}}) = M + k_p + 2\,k_{e1} + k_{e2}. \tag{4.21}$$

Case 2: Zero-sum from the extra block

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

$$\begin{pmatrix} \mathbf{H}_{\text{extra}} & \mathbf{0}_{\mathbf{r_2}\times\mathbf{c_2}} & \mathbf{0}_{\mathbf{r_2}\times\mathbf{c_2}} & \mathbf{0}_{\mathbf{r_2}\times\mathbf{c_2}} \\ \mathbf{I}_{\mathbf{c_2}} & \mathbf{I}_{\mathbf{c_2}} & \mathbf{0}_{\mathbf{c_2}\times\mathbf{c_2}} & \mathbf{0}_{\mathbf{c_2}\times\mathbf{c_2}} \\ \mathbf{I}_{\mathbf{c_2}} & \mathbf{0}_{\mathbf{c_2}\times\mathbf{c_2}} & \mathbf{I}_{\mathbf{c_2}} & \mathbf{0}_{\mathbf{c_2}\times\mathbf{c_2}} \\ \mathbf{0}_{\mathbf{c_2}\times\mathbf{c_2}} & \mathbf{0}_{\mathbf{c_2}\times\mathbf{c_2}} & \mathbf{I}_{\mathbf{c_2}} & \mathbf{I}_{\mathbf{c_2}} \end{pmatrix}$$

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

1. **Initial flips.** Selecting column j of H_{extra} flips the jth row of both the second and third identity blocks from 0 to 1.

2. **Cancel rows 2 and 3.**

   - To restore row 2 to zero, select column j of the second identity block.

- To restore row 3 to zero, select column j of the third identity block.

At this point three columns have been used: one from $\mathbf{H}_{\mathrm{extra}}$ plus two identity columns.

3. **Chain-reaction flip.** Selecting the third identity column also flips row 4 from 0 to 1 (because that identity block has a 1 in row 4).

4. **Final cancellation.** To restore row 4 to zero, select column j of the fourth identity block.

$$1_{(\mathbf{H}_{\mathrm{extra}})} + 1_{\text{(cancel row 2)}} + 1_{\text{(cancel row 3)}} + 1_{\text{(cancel row 4)}} = 4.$$

Therefore, for each weight-N codeword one must pick its N columns plus $3N$ additional identity columns, for a total of $4N$. Hence in this case

$$d_{\min}^{tx(2)} = 4N. \tag{4.22}$$

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. Therefore, we have:

$$d_{\min}(\mathbf{H}_{\mathrm{Enhanced}}) \geq \min(M + k_p + 2\,k_{e1} + k_{e2},\ 4N) = \min(d_{min}^{tx(1)}, d_{min}^{tx(2)}). \tag{4.23}$$

4.4 Enhanced & Partial Structure

As illustrated in Fig. 4.3, the partial and enhanced structures are merged, and the extra VNs are divided into transmitted and non-transmitted groups. In that figure, white circles denote variable nodes that are punctured and black circles denote those that are actually transmitted.The

corresponding parity-check matrix representation derived from this tanner graph structure is shown in Fig. 4.4.

$$\mathcal{E}_{\text{tx}} \;=\; \{\, i \in \{1, \ldots, c_2\} \mid \text{extra VN } e_i \text{ transmitted directly}\}, \tag{4.24}$$

$$\mathcal{E}_{\text{non-tx}} = \{1, \ldots, c_2\} \setminus \mathcal{E}_{\text{tx}}. \tag{4.25}$$

Since the total transmission length remains fixed at the original payload size, sending extra bits requires puncturing an equal number of payload bits. As a result, payload decoding performance incurs a slight degradation.

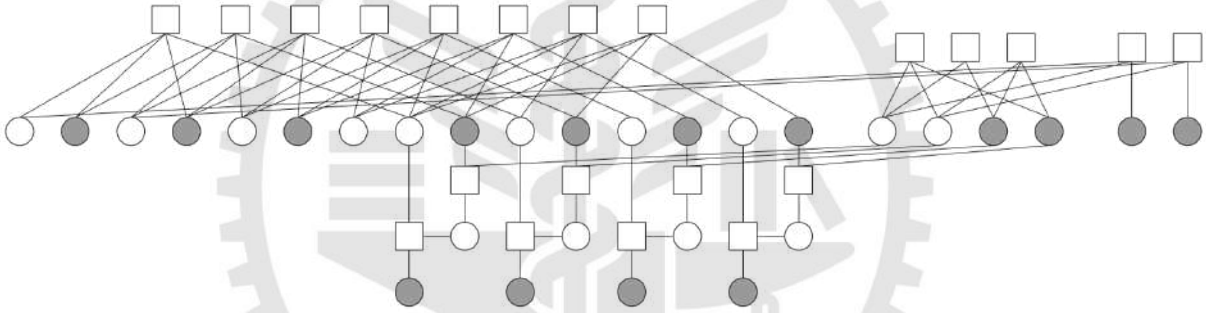


Figure 4.3: Enhanced & Partial structure tanner graph example

The resulting parity-check matrix takes the general form:

$$\mathbf{H}_{\text{Enhanced\&Partial}} = \begin{pmatrix} \mathbf{H}_{\text{payload}} & \mathbf{0}_{r_1 \times c_2} & \mathbf{0}_{r_1 \times |\mathcal{E}_{\text{non-tx}}|} & \mathbf{0}_{r_1 \times c_2} & \mathbf{0}_{r_1 \times c_2} \\ \mathbf{0}_{r_2 \times c_1} & \mathbf{H}_{\text{extra}} & \mathbf{0}_{r_2 \times |\mathcal{E}_{\text{non-tx}}|} & \mathbf{0}_{r_2 \times c_2} & \mathbf{0}_{r_2 \times c_2} \\ \mathbf{M}_{\text{payloadpuncpos}} & \mathbf{M}_{\text{extrspunc}} & \mathbf{I}_{|\mathcal{E}_{\text{non-tx}}|} & \mathbf{0}_{c_2 \times c_2} & \mathbf{0}_{c_2 \times c_2} \\ \mathbf{M}_{\text{Enhanced1}} & \mathbf{I}_{c_2} & \mathbf{0}_{c_2 \times |\mathcal{E}_{\text{non-tx}}|} & \mathbf{I}_{c_2} & \mathbf{0}_{c_2 \times c_2} \\ \mathbf{M}_{\text{Enhanced2}} & \mathbf{0}_{c_2 \times c_2} & \mathbf{0}_{c_2 \times |\mathcal{E}_{\text{non-tx}}|} & \mathbf{I}_{c_2} & \mathbf{I}_{c_2} \end{pmatrix}, \tag{4.26}$$

where

- $M_{\text{payloadpunc}} \in \mathbb{F}_2^{|\mathcal{E}_{\text{non-tx}}| \times c_1}$ has exactly one 1 in each row, where the 1 in row i marks the

$$\mathbf{H}_{\text{Enhanced\&Partial}} = \left[\begin{array}{cccccccccccccc|cc|cc|cc}
1&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1&1&0&1&0&0&0&1&0&0&0&0&0&0&0&0&0\\
\hline
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&0&0&0&0\\
\hline
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0\\
\hline
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&0&0&1\\
\hline
0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&1
\end{array}\right]$$

Figure 4.4: Enhanced & Partial structure parity-check matrix example

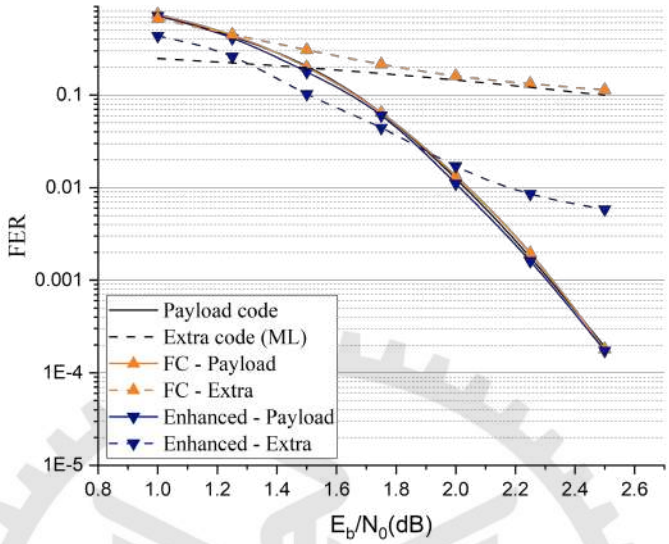
payload VN punctured in the ith merge.

- $M_{\text{extrapunc}} \in \mathbb{F}_2^{|\mathcal{E}_{\text{non-tx}}| \times c_2}$ has exactly one 1 in each row, where the 1 in row i marks the extra VN merged in the ith operation.

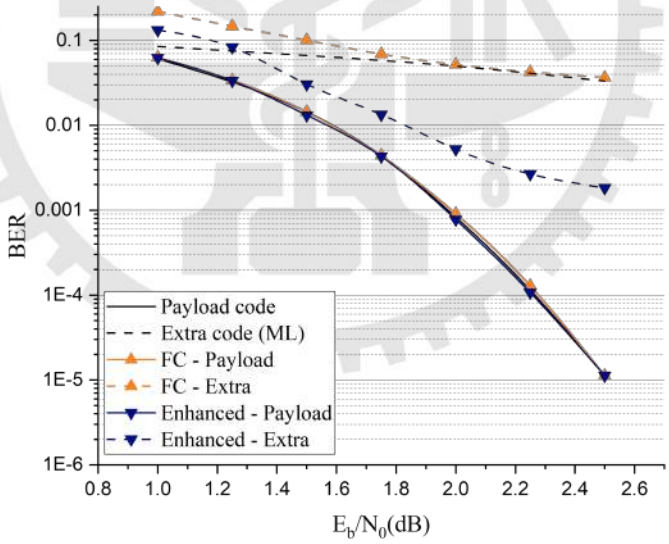
4.5 Simulation Result

The parameters of the simulation codes are listed in Table 3.1. The system employs a PEG–LDPC (1008, 504) code for the payload, combined with two extra codes: LDPC (10, 5) and BCH (15, 7). For the BP decoding process, the maximum number of iterations is set to 200. Under BPSK modulation over an AWGN channel, Fig. 4.5 shows that at low E_b/N_0 the LDPC extra code falls short of the ML bound, but as E_b/N_0 increases its performance improves rapidly and surpasses the reference ML bound. Likewise, Fig. 4.6 demonstrates the same trend for the BCH extra code. Meanwhile, the payload BER and FER under the enhanced structure remain virtually identical to pure PEG–LDPC, indicating no loss in payload performance.

It is worth emphasizing that the apparent performance gain of the extra bits beyond the conventional ML bound does not contradict optimal detection theory. This phenomenon arises from the joint decoding structure enabled by the proposed enhanced scheme. Specifically, the extra bits are superimposed onto the payload bits through XOR-based free-ride operations, allowing them to inherit the strong coding protection originally designed for the payload code. During belief propagation decoding, reliable payload information effectively serves as side information that assists the decoding of the extra bits, resulting in an additional coding gain. In contrast, the standalone ML detector treats the extra bits independently and therefore cannot exploit this payload-assisted decoding advantage.

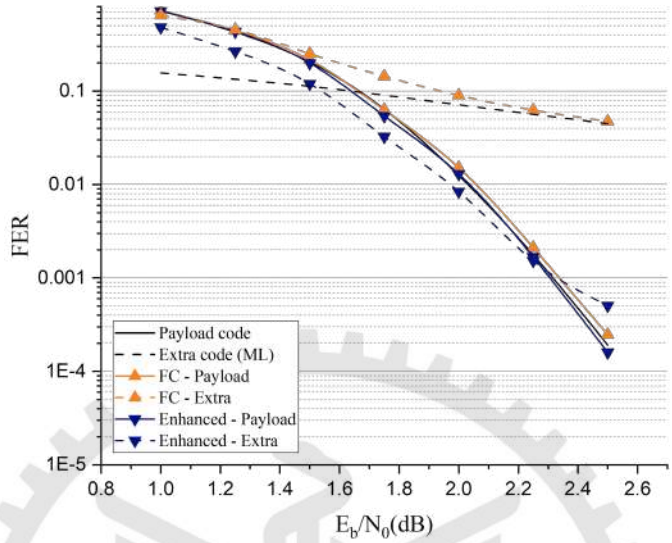


(a) FER

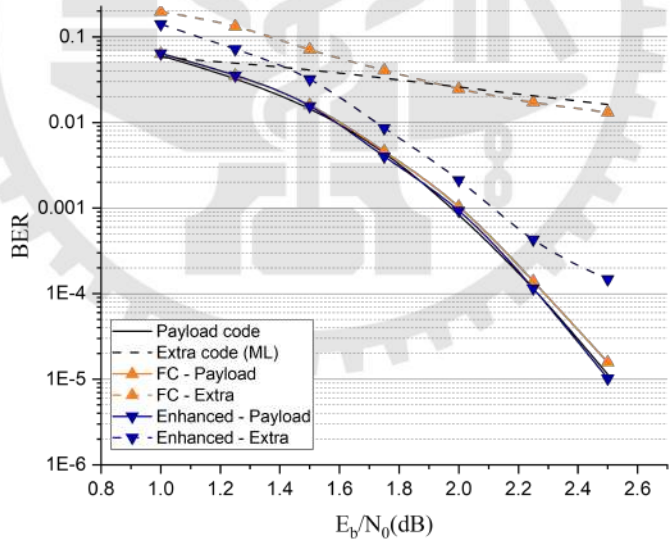


(b) BER

Figure 4.5: Performance of the enhanced structure with payload H(1008, 504) and extra H(10, 5)



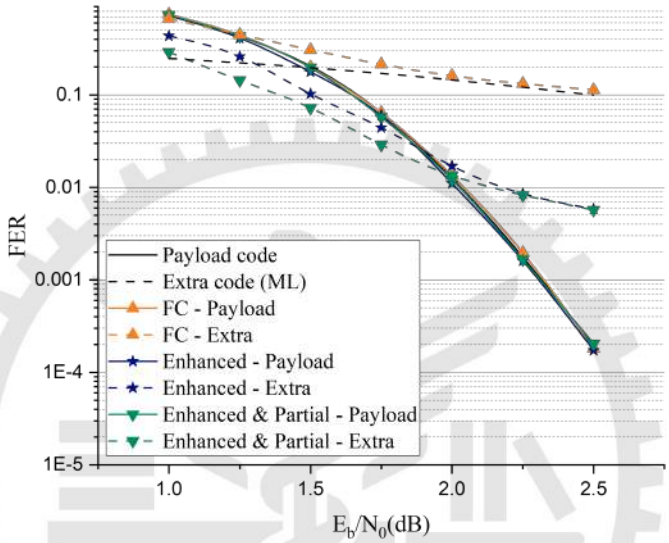
(a) FER



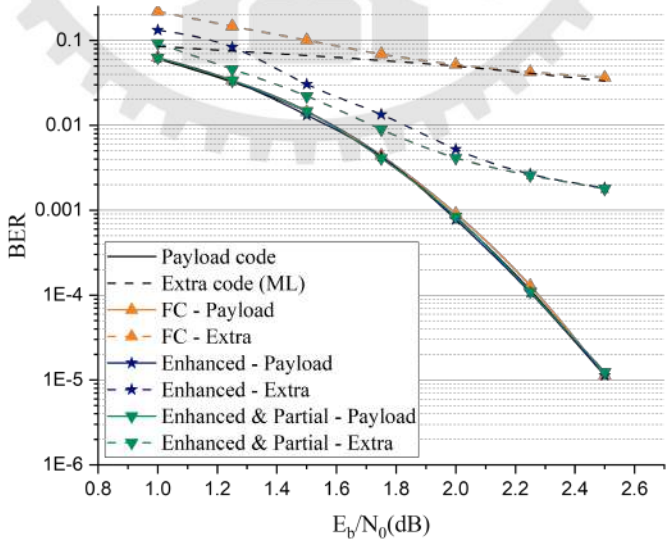
(b) BER

Figure 4.6: Performance of the enhanced structure with payload H(1008, 504) and extra BCH(15, 7)

In the Enhanced & Partial structure, to transmit additional extra bits within the fixed payload length, an equal number of payload bits are punctured and the remaining extra bits are merged with their corresponding payload bits into new symbols. As shown in Fig. 4.7 and Fig. 4.8, this approach introduces only a slight FER and BER degradation for the payload. However, in the low E_b/N_0 region, the directly transmitted extra bits yield a substantial improvement in error-correction performance; as E_b/N_0 increases, the extra bits BER drops rapidly.

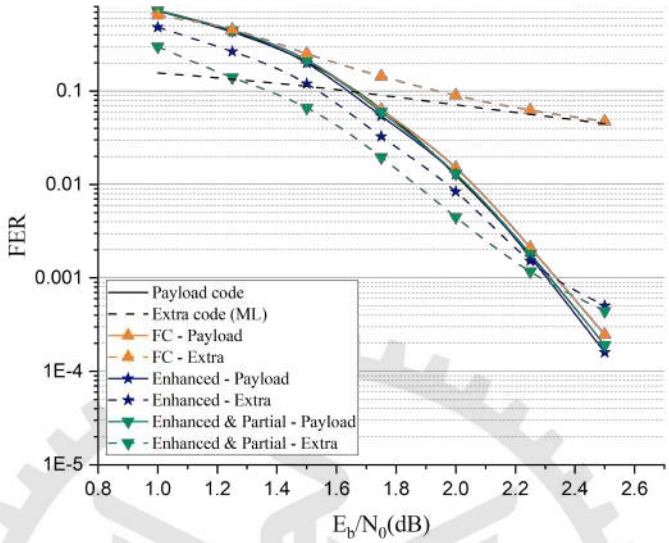


(a) FER

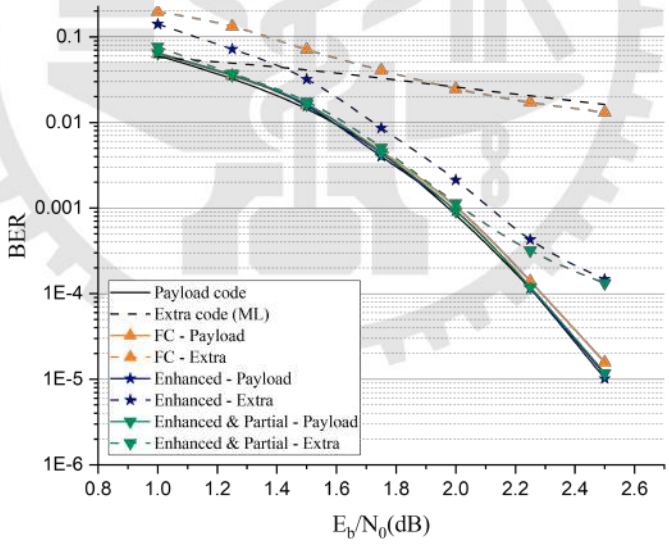


(b) BER

Figure 4.7: Performance of the enhanced & partial structure with payload H(1008, 504) and extra H(10, 5)



(a) FER



(b) BER

Figure 4.8: Performance of the enhanced & partial structure with payload H(1008, 504) and extra BCH(15, 7)

4.6 Computational Complexity Analysis

To evaluate the computational overhead introduced by the proposed method, the average decoding time of the enhanced structure is compared with the original free-ride scheme. All measurements were conducted under identical hardware conditions, recording the average time required to successfully decode both payload and extra bits per frame. It is worth noting that the original free-ride scheme relies on an exhaustive search over the extra-bit candidates, resulting in a computational complexity on the order of $\mathcal{O}(2^k)$.

As illustrated in Fig. 4.9 and Fig. 4.10, contrary to the intuition that a larger parity-check matrix would incur higher decoding latency, the enhanced structure achieves a significant reduction in average decoding time compared to the original free-ride method. Although the enhanced structure introduces additional nodes and edges, its improved error-correction capability enables the belief propagation algorithm to converge more rapidly. As a result, substantially fewer decoding iterations are required to recover the correct codeword, leading to a lower overall processing time. These results demonstrate that the enhanced structure not only improves decoding reliability but also enhances computational efficiency.

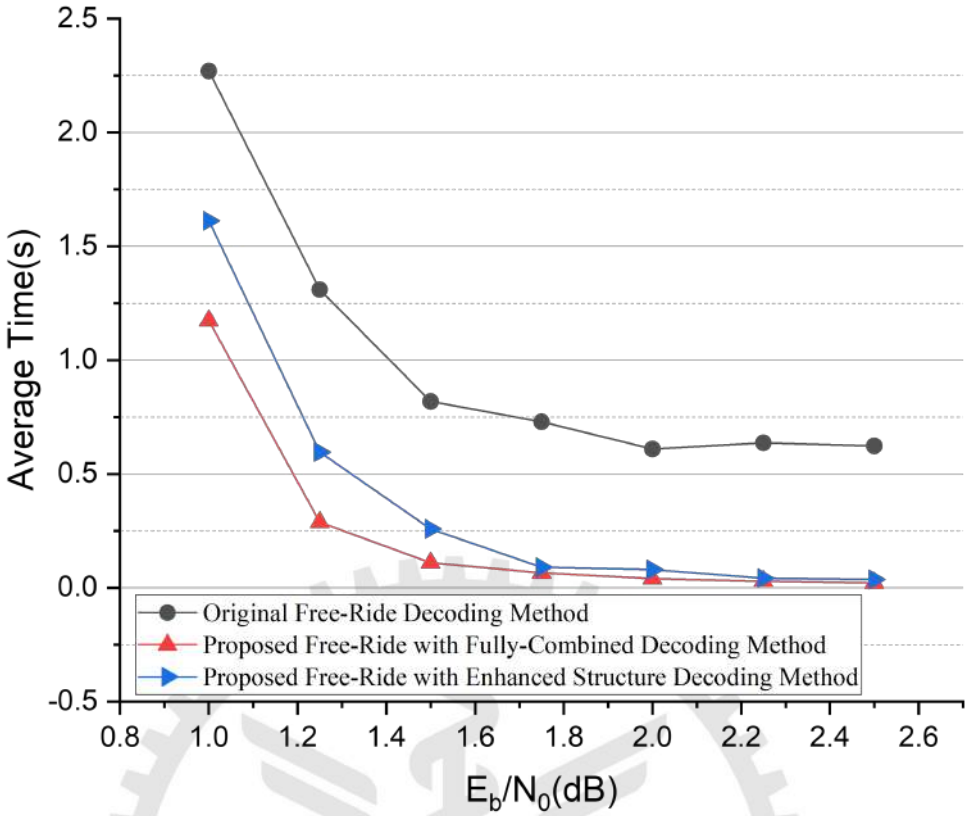


Figure 4.9: Average decoding time comparison - payload H(1008,504), extra H(10,5)

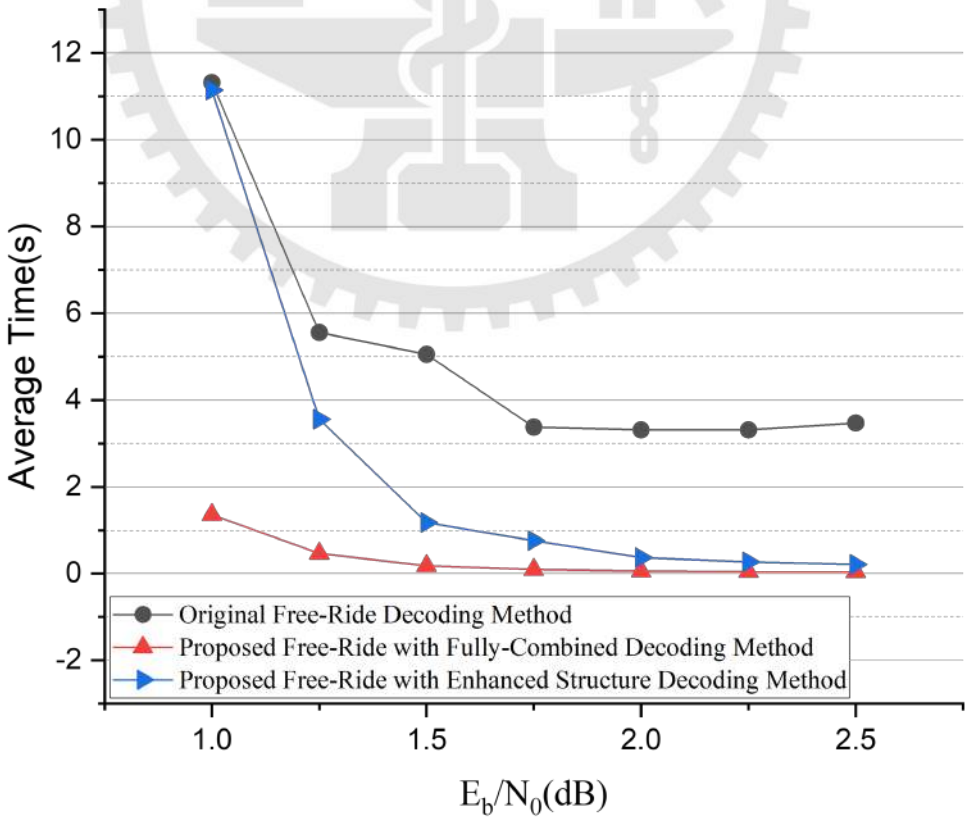


Figure 4.10: Average decoding time comparison - payload H(1008,504), extra BCH(15,7)

Chapter 5.

Incremental Redundancy Hybrid Automatic Repeat reQuest

5.1 Redundancy Control and Decoding Mechanism for Free-Ride HARQ

In the incremental redundancy hybrid automatic repeat request (IR-HARQ) framework adopted in this work, two redundancy versions (RVs) are defined to control the bit-level retransmission behavior:

5.1.1 Payload Version

For the payload scheme, the two redundancy versions are defined as follows:

- **RV0:** On the first transmission attempt, transmit the majority of payload bits. Specifically, all payload bits p_j except a small subset reserved for retransmission are sent to the receiver.
- **RV1:** If the initial decoding fails and a retransmission is triggered, transmit exactly those payload bits that were omitted in RV0. This completes the transmission of the full payload $\{p_j\}$.

5.1.2 Free-Ride Version

For the free-ride scheme (enhanced structure), the two redundancy versions are defined as follows:

- **RV0:** On the first transmission, payload bits in $\mathcal{I}$ are XOR combined with their corresponding extra bits to form the transmitted codeword. The remaining payload bits (those not involved in the XOR) are omitted (punctured) and not sent.
- **RV1:** If the initial decoding fails, retransmit only the payload bits that were omitted in RV0. No extra bits are sent in this retransmission.

5.1.3 Decoding Strategy for Free-Ride Scheme

Upon receiving RV1, the receiver first attempts enhanced joint decoding over the combined tanner graph to recover both payload and extra bits. If this joint decoding succeeds, both sets of bits are delivered. If it fails, the receiver falls back to a standard LDPC decode on the payload alone, forgoing any attempt to recover extra bits and focusing solely on payload recovery. The overall two-stage decoding and fallback process is illustrated in Fig. 5.1, which shows the transition from joint decoding to payload-only decoding. This two-stage decode and fallback ensures that, even when the free-ride mechanism cannot recover all extra information, the original payload is still reliably restored.

Algorithm 6 Free-Ride Decoding Mechanism

Input: Received soft information $\mathbf{y}^{(0)}$ (RV0), $\mathbf{y}^{(1)}$ (RV1)
Output: Decoded payload $\hat{\boldsymbol{p}}$ and extra bits $\hat{\boldsymbol{e}}$

```
# RV0 Decoding
Attempt enhanced joint decoding on y^(0) over the combined tanner graph
if joint decoding success then
    p̂, ê ← joint decoder outputs
    return p̂, ê            # success on RV0
else
    Request RV1 retransmission
end if

# RV1 Decoding
Receive y^(1), combine with previous LLRs
Attempt enhanced joint decoding on the combined graph
if joint decoding success then
    p̂, ê ← joint decoder outputs
    return p̂, ê            # success on RV1
else
    ê ← forgo extra decoding
    Perform standard LDPC decode on payload subgraph
    if payload decoding converges then
        p̂ ← payload decoder output
        return p̂           # payload only
    else
        Declare decoding failure
    end if
end if
```

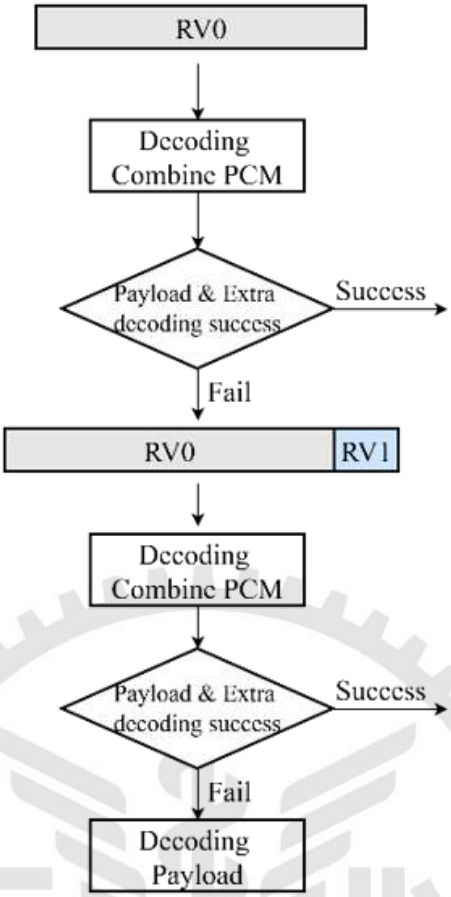


Figure 5.1: Free-Ride IR-HARQ flow

5.1.4 Proposed IR-HARQ Simulation Result

The following figures illustrate the throughput performance of the proposed free-ride hybrid IR-HARQ scheme compared to the payload scheme. Throughput is a critical metric, representing the successful rate of information delivery.

It is noted that the results in chapters 3 and 4 are presented in terms of E_b/N_0, while the throughput performance in this chapter is evaluated using E_s/N_0. The relationship between the two is given by

$$E_s/N_0(dB) = E_b/N_0(dB) + 10\log_{10}(R \cdot \log_2 M), \tag{5.1}$$

where R denotes the effective code rate and M is the modulation order.

Based on the simulation results Fig. 5.2 and Fig. 5.3 shown in the throughput curves, the

free-ride hybrid IR-HARQ scheme consistently provides a performance gain over the standard payload scheme, particularly at higher channel quality. In the low E_s/N_0 region, the throughput of both schemes is nearly identical, indicating that the additional complexity does not degrade payload throughput under low E_s/N_0 conditions. As the E_s/N_0 improves, the free-ride hybrid scheme achieves a noticeably higher peak throughput.

In Fig. 5.2, the proposed scheme achieves a maximum throughput gain of approximately 6% when $E_s/N_0 > 0$ dB, whereas in Fig. 5.3, the peak improvement is around 3% for $E_s/N_0 > -0.75$ dB. This throughput enhancement is obtained at the cost of an increased receiver-side decoding structure due to joint payload and extra-bit processing. Nevertheless, the enlarged decoding structure enables more reliable information exchange during belief propagation, leading to higher successful decoding rates. These results indicate that the proposed free-ride hybrid IR-HARQ scheme achieves a favorable trade-off between decoding complexity and throughput efficiency under favorable channel conditions.

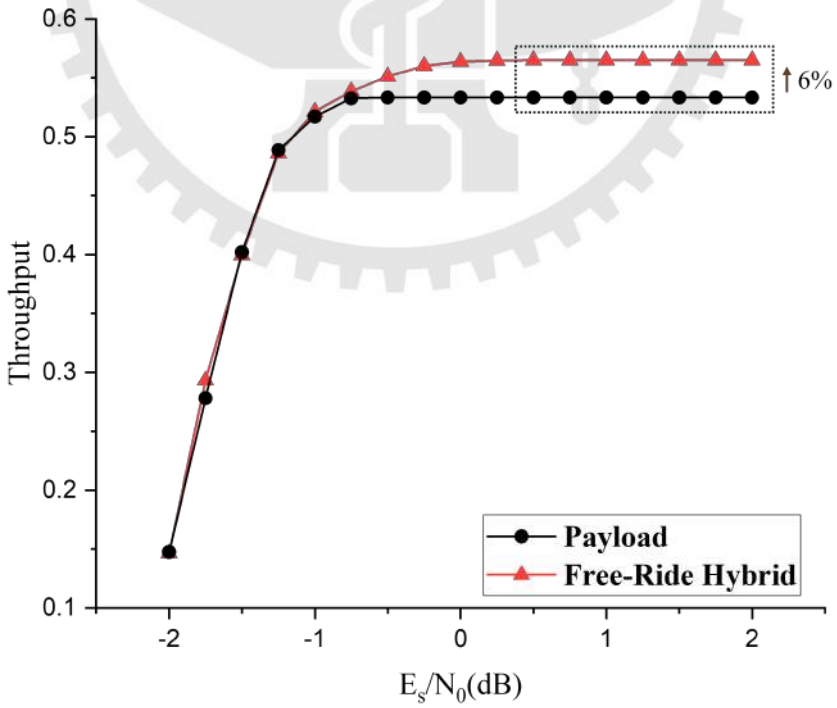


Figure 5.2: Throughput comparison: Payload-only vs Free-Ride - PEG-LDPC(1008,504) with BCH(63,30), 200 iterations

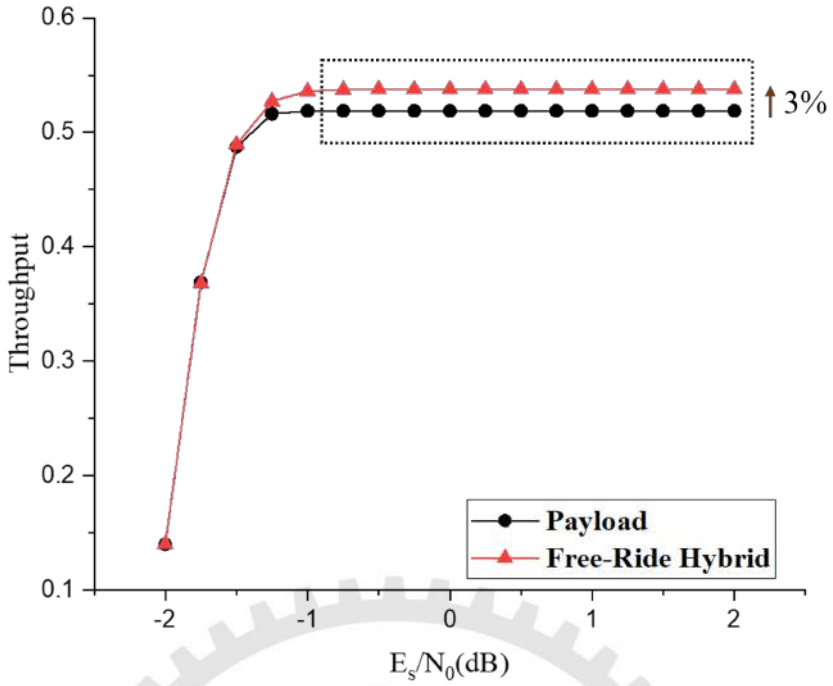


Figure 5.3: Throughput comparison: Payload-only vs Free-Ride - LDPC(2640,1320) with LDPC(96,48), 200 iterations

5.2 5G IR-HARQ

In this study, the transmission procedure of the proposed free-ride IR-HARQ adheres to the redundancy version mechanism specified in the 5G IR-HARQ protocol. As illustrated in Fig. 2.10, the 5G IR-HARQ process defines four redundancy versions (RV0–RV3), each corresponding to a distinct puncturing pattern of the mother code. To simplify the simulation while preserving alignment with the standard HARQ structure, employ only two redundancy versions, namely RV0 and RV2. In this configuration, RV0 serves as the initial transmission and carries the systematic bits together with a portion of the parity bits, whereas RV2 is used for the subsequent transmission and supplies additional parity bits that enable incremental-redundancy combining at the receiver.

5.2.1 Payload Version

For the payload scheme, the two redundancy versions are defined as follows:

- **RV0:** Transmit the systematic bits together with a portion of the parity bits.
- **RV2:** Transmit additional parity bits to enable incremental-redundancy combining at the receiver.

5.2.2 Free-Ride Version

In the free-ride design, the payload is first encoded with the 5G NR LDPC mother code. Before rate matching, an auxiliary bit stream is superimposed by XOR onto a designated index set $\mathcal{I}$ of payload bit positions: for indices $j \in \mathcal{I}$ the transmitted bit is $p_j \oplus e_j$, whereas for $j \notin \mathcal{I}$ it remains p_j. The resulting composite sequence is then processed by the standard 5G NR rate-matching pipeline to produce the RV patterns used in transmission.

- **RV0:** Apply the 5G NR RV0 rate-matching pattern to the LDPC mother code built from $\{\tilde{p}_j\}$, which delivers the systematic bits and a portion of the parity bits. At the receiver, a payload-only LDPC decoding is first attempted using the soft information from RV0. The extra bits are not explicitly decoded at this stage.
- **RV2:** Upon a NACK, transmit additional parity bits by applying the 5G NR RV2 rate-matching pattern of the same mother code. The receiver combines RV0 and RV2 soft information and then performs an enhanced joint decoding over the combined tanner graph to recover both payload and extra bits.

5.2.3 Free-Ride Decoding Strategy under 5G IR-HARQ Framework

Upon receiving RV0, the receiver first attempts payload-only LDPC decoding using the available soft information.If this initial decoding attempt fails, the transmitter issues a retransmission according to the IR-HARQ redundancy version schedule. Upon reception of the retransmission, the receiver performs incremental-redundancy combining of the soft information from RV0 and RV2, and then attempts an enhanced joint decoding over the combined tanner graph to recover both the payload and the extra bits. If the joint decoding attempt is unsuccessful, the receiver falls back to a payload-only decoding of the combined RV0 、RV2 soft information. The overall two-stage decoding and fallback process is illustrated Fig. 5.4.

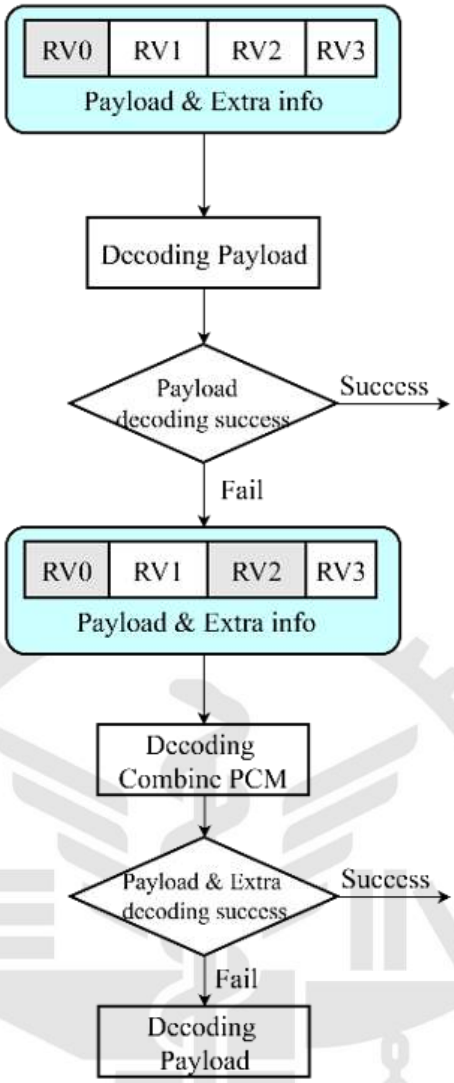


Figure 5.4: Free-Ride 5G IR-HARQ flow

5.2.4 Proposed 5G IR-HARQ Simulation Result

As shown in Figs. 5.5 and 5.6, the proposed free-ride on 5G IR-HARQ scheme increases throughput by up to 5% for E_s/N_0 in the range -3 to 0.5 dB compared to the standard payload-only method. This gain comes from enhanced joint decoding after combining the initial transmission (RV0) and the retransmission (RV2), which helps recover extra bits. Importantly, this process does not degrade the main payload's performance. This benefit holds true for different decoder iterations (20 and 50), showing that free-ride boosts overall spectral efficiency without compromising the primary service.

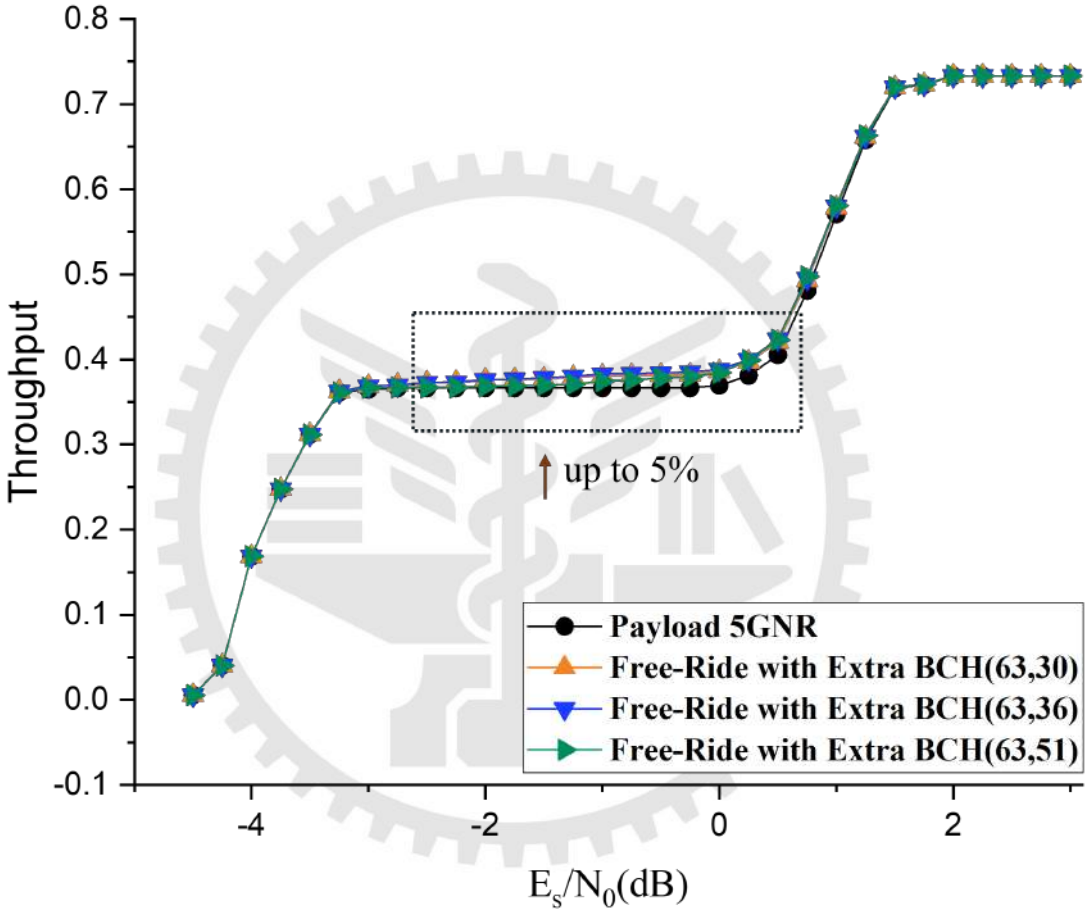


Figure 5.5: Throughput comparison: Payload-only vs Free-Ride - LDPC(1904,616) with BCH(63), 20 iterations.

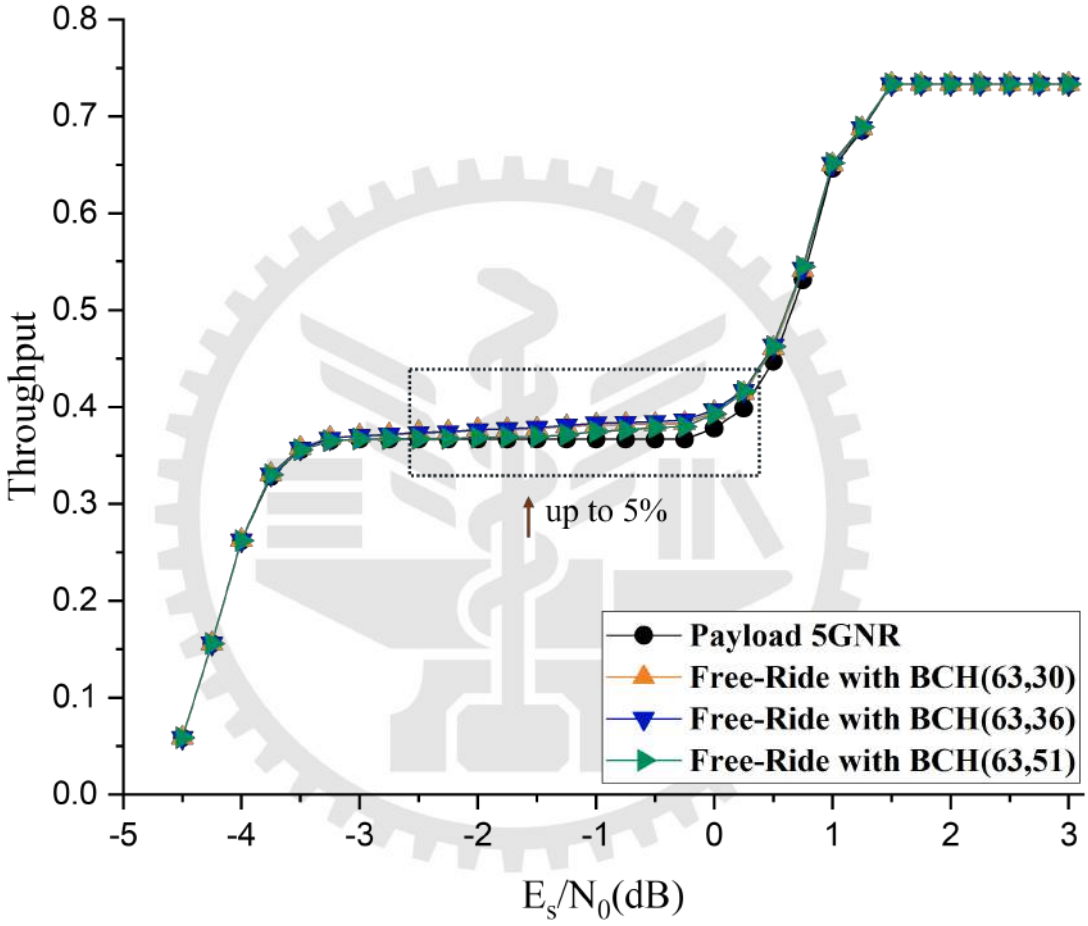


Figure 5.6: Throughput comparison: Payload-only vs Free-Ride - LDPC(1904,616) with BCH(63), 50 iterations.

Chapter 6.

Conclusions and Future Works

6.1 Conclusions

This thesis proposes a novel joint decoding architecture aimed at carrying auxiliary meta-data bits without altering existing LDPC codes or resource allocation. Unlike conventional methods that often require additional bandwidth or suffer from high computational costs, the proposed scheme innovatively integrates the parity-check constraints of both payload and extra bits into a single unified tanner graph. This design enables the simultaneous recovery of both information streams within a single decoding process. To optimize overall performance, three distinct structures fully-combined, partial, and enhanced were developed and analyzed. Crucially, a major breakthrough in computational efficiency is achieved by reducing the decoding complexity from the exponential growth of traditional ML methods to linear complexity with respect to the number of graph edges. This theoretical advantage is explicitly validated by average decoding time simulations, which confirm that the proposed scheme significantly lowers the decoding latency, resolving the critical implementation bottleneck for practical deployment.

On the system level, the free-ride mechanism is successfully integrated into the standard 5G IR-HARQ framework. A robust decoding flow is proposed to guarantee the reliability of the primary payload while achieving additional coding gains. Specifically, the proposed scheme demonstrates a throughput enhancement of up to 6% in high SNR regimes compared to the

standard payload-only transmission. The scheme is practically implemented using standard 5G LDPC codes and validated within the 5G IR-HARQ framework, confirming its practical feasibility and seamless compatibility with next-generation communication standards.

6.2 Future Works

Future research will focus on systematic optimization of the proposed free-ride coding scheme and its integration into advanced systems. Firstly, efforts will concentrate on the optimization of puncturing patterns. This includes investigating the most effective puncturing strategies when adapting the mechanism to different payload codes (varying code rates and block lengths), ensuring the design is generalizable and maintains high coding efficiency. Secondly, the optimization of the IR-HARQ transmission strategy will be pursued. This involves in-depth analysis of the transmission order of RVs and the potential for dynamic adjustment of RV size during retransmission, aiming to maximize spectral efficiency and system resilience in time-varying wireless channels.

Furthermore, this thesis reveals a notable phenomenon, embedding short codes (as extra bits) within long payload codewords using the proposed structure yields significant performance advantages. This suggests a unique benefit of such heterogeneous length combinations. The detailed theoretical mechanisms and root causes behind this specific performance gain warrant further investigation in future research.

References

[1] Xianbin Wang, Y. Wu and B. Caron, "Transmitter identification using embedded pseudo random sequences," in *IEEE Transactions on Broadcasting*, vol. 50, no. 3, pp. 244-252, Sept. 2004.

[2] E. G. Larsson and R. Moosavi, "Piggybacking an Additional Lonely Bit on Linearly Coded Payload Data," in *IEEE Wireless Communications Letters*, vol. 1, no. 4, pp. 292-295, August 2012.

[3] E. S. Kang, S. W. Hong and D. S. Han, "Improved speed near field communication with rotated QPSK constellation and hidden data transmission," *IEEE international Symposium on Broadband Multimedia Systems and Broadcasting*, Seoul, Korea (South), 2012.

[4] X. Huang, X. Ma, L. Lin and B. Bai, "Accessible Capacity of Secondary Users," in *IEEE Transactions on Information Theory*, vol. 60, no. 8, pp. 4722-4738, Aug. 2014.

[5] S. Cai, S. Zhao, and X. Ma, “ Packing additional bits into LDPC coded data, ” *Electron. Lett.*, vol. 55, no. 18, pp. 997–998, Sep. 2019.

[6] 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.

[7] M. Xie, J. Gong, Q. Wang and X. Ma, "Achieving Two-Level Age by Free-Ride Coding in Preemptive Mission-Critical Networks," *2022 IEEE International Conference on Communications Workshops (ICC Workshops)*, Seoul, Korea, Republic of, 2022.

[8] M. Xie, J. Gong, Q. Wang, S. Cai and X. Ma, "Reducing Age of Extra Data by Free Riding on Coded Transmission in Multiaccess Networks," *2022 IEEE Wireless Communications and Networking Conference (WCNC)*, Austin, TX, USA, 2022.

[9] J. Chen, Y. Wang, Q. Wang, H. Wan and X. Ma, "Free-Ride Transmission of Semantic Features in Wireless Video Surveillance Systems," *2024 IEEE Wireless Communications and Networking Conference (WCNC)*, Dubai, United Arab Emirates, 2024.

[10] Q. Wang, S. Cai, Y. Wang and X. Ma, "Free-Ride Feedback and Superposition Retransmission Over LDPC Coded Links," in *IEEE Transactions on Communications*, vol. 71, no. 1, pp. 13-25, Jan. 2023.

[11] X. Ma, Q. Wang, M. Xie and S. Cai, "Implicit Globally-Coupled LDPC Codes Using Free-Ride Coding," *2022 IEEE Wireless Communications and Networking Conference (WCNC)*, Austin, TX, USA, 2022.

[12] Q. Wang, S. Cai and X. Ma, "Free-Ride Coding for Constructions of Coupled LDPC Codes," in *IEEE Transactions on Communications*, vol. 71, no. 3, pp. 1259-1270, March 2023.

[13] X. Ma, Q. Wang, S. Cai and X. Xie, "Implicit Partial Product-LDPC Codes Using Free-Ride Coding," *ICC 2022 - IEEE International Conference on Communications*, Seoul, Korea, Republic of, 2022.

[14] F. Meng, Q. Wang, S. Cai and X. Ma, "Parallel Decoding of PIC-LDPC Codes Aided by Free-Ride Coding," *2022 IEEE/CIC International Conference on Communications in China (ICCC)*, Sanshui, Foshan, China, 2022.

[15] R. Gallager, "Low-density parity-check codes," in *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21-28, January 1962.

[16] E. Berlekamp, R. McEliece and H. van Tilborg, "On the inherent intractability of certain coding problems (Corresp.)," in *IEEE Transactions on Information Theory*, vol. 24, no. 3, pp. 384-386, May 1978.

[17] J. S. Leon, "A probabilistic algorithm for computing minimum weights of large error-correcting codes," in *IEEE Transactions on Information Theory*, vol. 34, no. 5, pp. 1354-1359, Sept. 1988.

[18] M. P. C. Fossorier and Shu Lin, "Soft decision decoding of linear block codes based on ordered statistics for the Rayleigh fading channel with coherent detection," in *IEEE Transactions on Communications*, vol. 45, no. 1, pp. 12-14, Jan. 1997.

[19] H. Y. Park, J. W. Kang, K. S. Kim and K. C. Whang, "Efficient Puncturing Method for Rate-Compatible Low-Density Parity-Check Codes," in *IEEE Transactions on Wireless Communications*, vol. 6, no. 11, pp. 3914-3919, November 2007.

[20] 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.

[21] L. Zhou, W. -C. Huang, R. Zhao and Y. -C. He, "A non-greedy puncturing method for rate-compatible LDPC codes," in *Journal of Communications and Networks*, vol. 19, no. 1, pp. 32-40, February 2017.

[22] R. Wiesmayr, D. Nonaca, C. Dick and C. Studer, "Optimizing Puncturing Patterns of 5G NR LDPC Codes for Few-Iteration Decoding," *2024 58th Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA, USA, 2024.

[23] 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.

[24] 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.

[25] 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.

[26] Tao Tian, C. R. Jones, J. D. Villasenor and R. D. Wesel, "Selective avoidance of cycles in irregular LDPC code construction," in *IEEE Transactions on Communications*, vol. 52, no. 8, pp. 1242-1247, Aug. 2004.

[27] A. Ahmed, A. Al-Dweik, Y. Iraqi, H. Mukhtar, M. Naeem and E. Hossain, "Hybrid Automatic Repeat Request (HARQ) in Wireless Communications Systems and Standards: A Contemporary Survey," in *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2711-2752, Fourthquarter 2021.

[28] 3GPP, “Technical specification (TS) 38.212. 5G NR multiplexing and channel coding (Release 15),” July, 2018.

[29] Q. Wang, S. Cai, L. Chen and X. Ma, "A Throughput-Enhanced HARQ Scheme for 5G System via Partial Superposition," in *IEEE Communications Letters*, vol. 24, no. 10, pp. 2162-2166, Oct. 2020.

[30] T. Richardson and S. Kudekar, "Design of Low-Density Parity Check Codes for 5G New Radio," in *IEEE Communications Magazine*, vol. 56, no. 3, pp. 28-34, March 2018.

Appendix

Appendix A: Alternative Redundancy Version Combinations for 5G IR-HARQ

Motivation

In chapter 5.2, the performance of the proposed free-ride scheme was evaluated using the standard redundancy version sequence (RV0, RV2) as specified in 5G NR. To provide a comprehensive analysis of the robustness of the proposed method under different retransmission strategies, this appendix presents simulation results for alternative two-transmission combinations, including RV0 combined with RV1 (RV0+RV1) and RV0 combined with RV3 (RV0+RV3).

Simulation Results

Fig. A.1, Fig. A.2 and Fig. A.3 illustrate the throughput performance of the RV0+RV1 combination under iteration limits of 20, 30, and 50.

For the RV0+RV1 configuration, the throughput performance is generally inferior to that of the baseline payload-only scheme across different iteration limits 20, 30, and 50. This behavior mainly stems from the fact that both RV0 and RV1 correspond to early portions of the rate-matching sequence in 5G IR-HARQ, where systematic and closely related parity bits are transmitted. As a result, the superimposed extra bits in the free-ride scheme introduce additional coupling on these early payload positions, which degrades the reliability of payload decoding.

Increasing the number of decoding iterations slightly alleviates this effect but cannot fully compensate for the structural limitation of the RV0+RV1 combination.

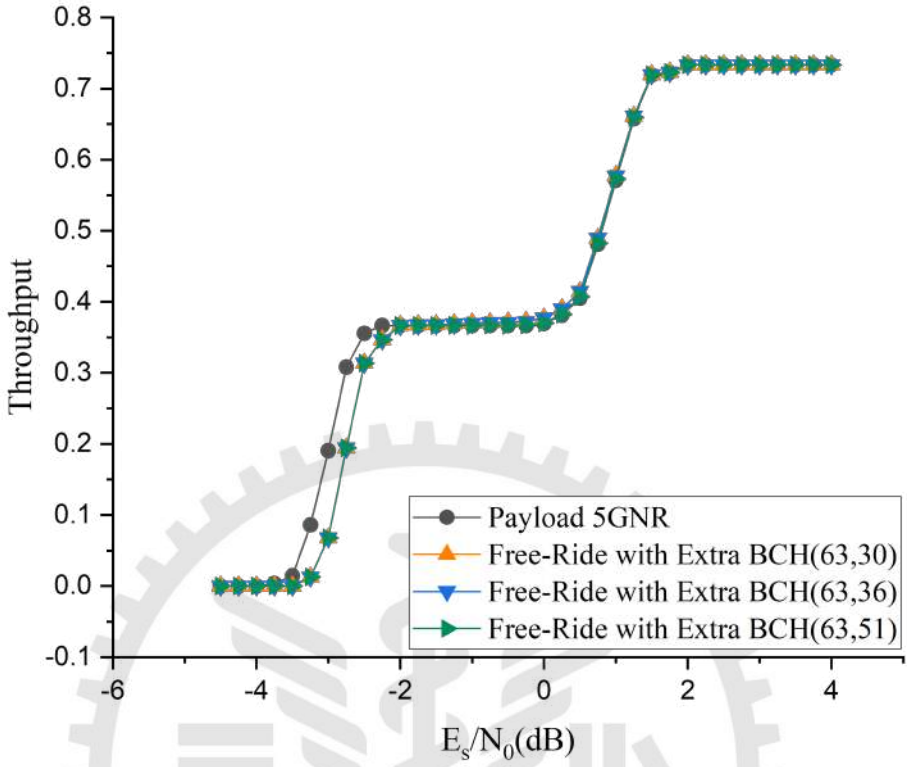


Figure A.1: Throughput comparison for RV0+RV1 with 20 decoding iterations.

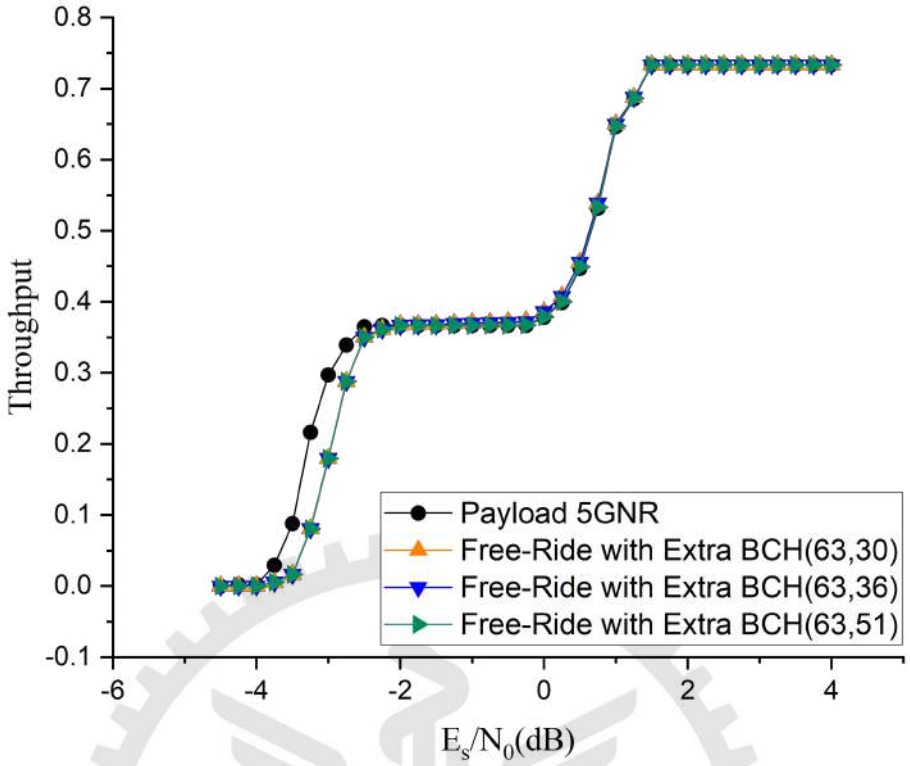


Figure A.2: Throughput comparison for RV0+RV1 with 30 decoding iterations.

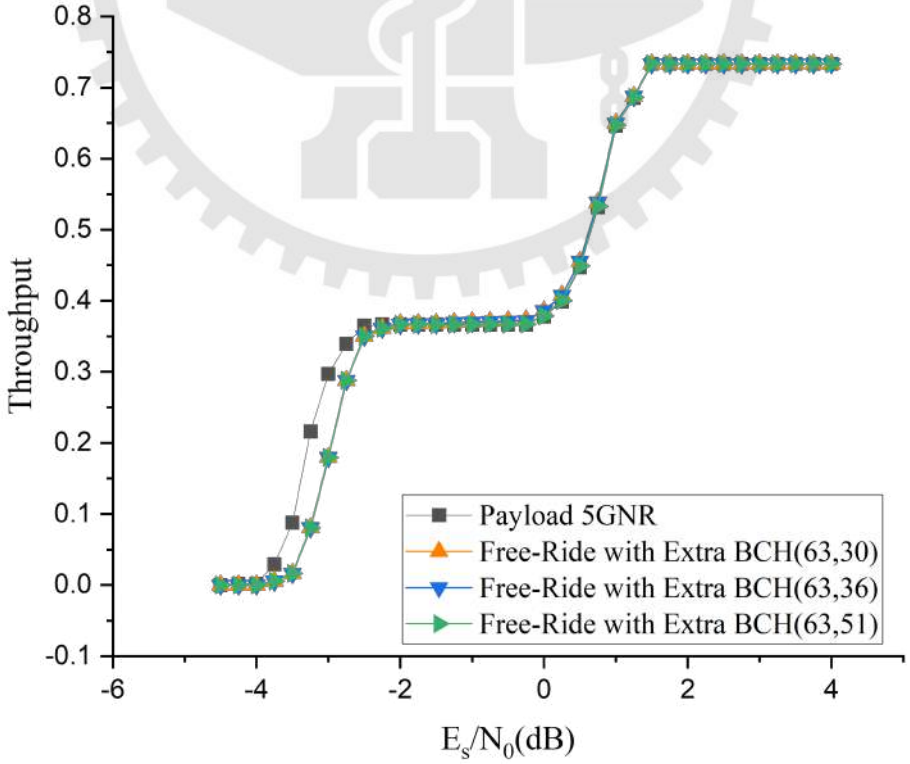


Figure A.3: Throughput comparison for RV0+RV1 with 50 decoding iterations.

Fig. A.4, Fig. A.5 and Fig. A.6 illustrate the throughput performance of the RV0+RV3 combination under iteration limits of 20, 30, and 50.

In contrast, the RV0+RV3 configuration exhibits throughput performance that is nearly identical to the payload-only scheme. This is because most payload bits connected to the extra layer in the enhanced structure are primarily associated with parity positions transmitted in RV2 rather than RV3. Although RV3 partially overlaps with earlier redundancy versions, it does not provide sufficient new information to enable reliable decoding of the extra bits. Consequently, the free-ride mechanism effectively collapses to payload-only decoding, resulting in comparable throughput performance.

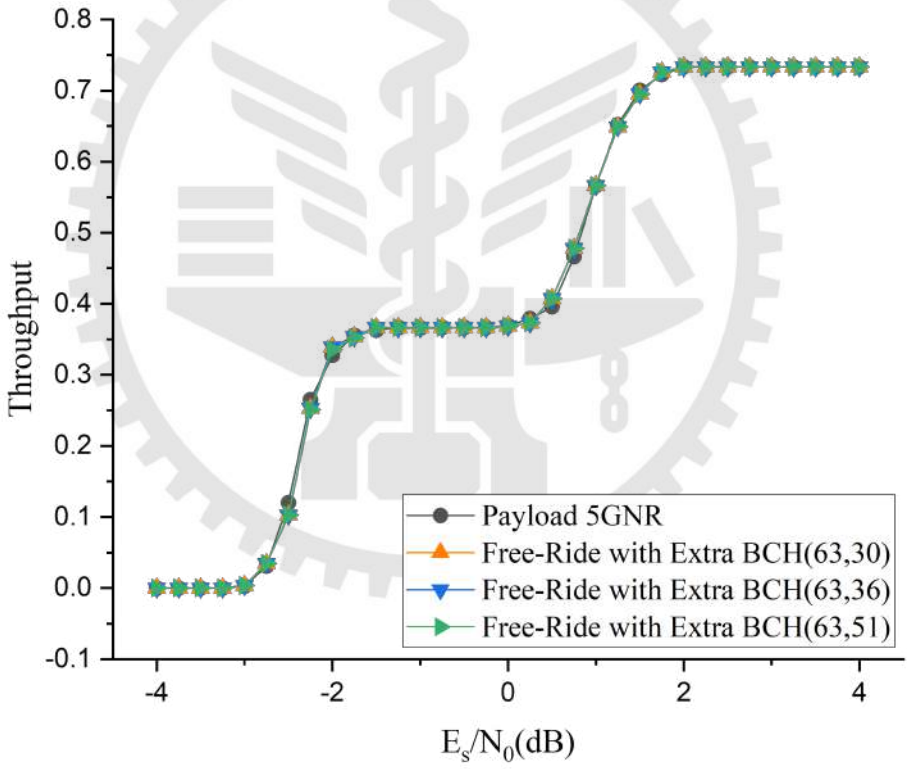


Figure A.4: Throughput comparison for RV0+RV3 with 20 decoding iterations.

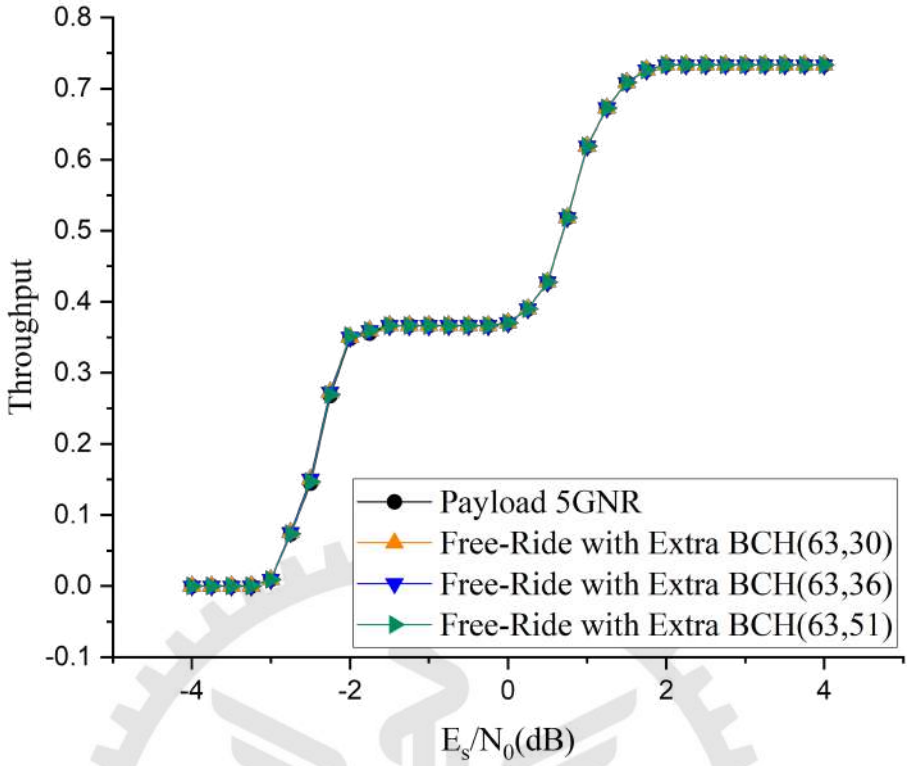


Figure A.5: Throughput comparison for RV0+RV3 with 30 decoding iterations.

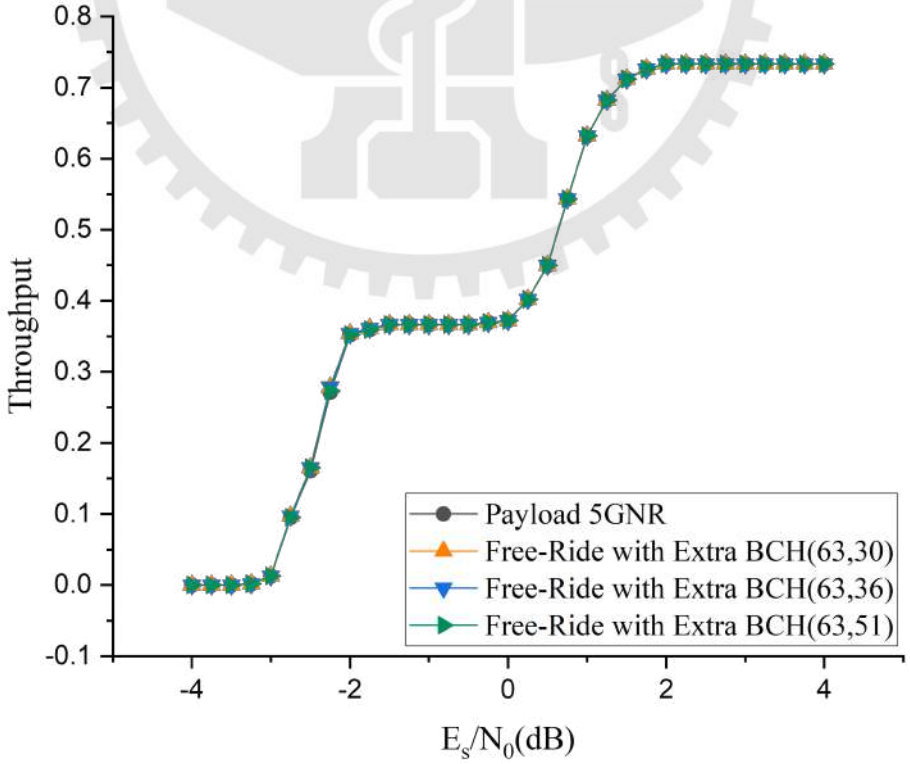


Figure A.6: Throughput comparison for RV0+RV3 with 50 decoding iterations.