Interleaver Design for Turbo Codes Based on Complete Knowledge of Low-Weight Codewords of RSC Codes

Kwame Ackah Bohulu and Chenggao Han
Graduate School of Informatics and Engineering,
The University of Electro-Communications,
1-5-1 Chofugaoka, Chofu-shi, Tokyo, 182–8585, Japan
Email: {bohulu, han.ic}@uec.ac.jp

***Abstract*—In this paper, we present a novel design framework for Turbo codes (TCs). Referring to the generator functions of the employed recursive systematic convolutional (RSC) codes, we develop a method to determine not only the multiplicity, but also detailed patterns of low-weight (LW) input and LW parity-check (LWPC) sequences for the component codes. Next, we select the component RSC code based on the knowledge obtained by the proposed method and derive expressions necessary to evaluate the free distance d_f of the resultant TC. Finally, we propose a novel interleaver class, named Coset interleaver, with its parameter selection criteria to enlarge d_f. Based on the proposed design framework, we succeed in designing a rate $1/3$ TC with $d_f = 42$ for frame size $N = 1024$. At an E_b/N_0 of 2dB, our proposed design demonstrates a bit-error rate (BER) of 1.232×10^{-10}, which represents a significant improvement of 6.18×10^{-8} and 2.207×10^{-8} when compared to the BERs exhibited by the Quadratic Permutation Polynomial (QPP, $d_f = 30$) and S-random ($d_f = 21$) interleavers, respectively.**

***Index Terms*—RSC codes, low-weight codewords, Turbo Code, codeword patterns, interleaver design**

I. Introduction

The Turbo code (TC) [1] is known to be the first error-correcting code to achieve near Shannon limit performance over AWGN channels, and it has been adopted by many industrial standards, *e.g.*, IEEE 802.16e [2] and long-term evolution (LTE) [3]. There are many code constructions which are considered to be members of the TC family. By TC, we mean parallel concatenated convolutional codes (PCCC), *i.e.* codes which are generally constructed by concatenating recursive systematic convolutional (RSC) codes (usually of the same kind) in parallel via an interleaver. A well-designed TC realizes a large free distance d_f via the interleaver, which maps each input sequence with a *low-weight (LW) parity-check* (LWPC) sequence in a RSC code onto that with a high-weight one in other RSC codes. Thus, the TC design framework is essential in both RSC code selection and interleaver design, and requires the complete knowledge, that is, multiplicity and detailed patterns, of all *low-weight codewords* (LWCWs) of the employed RSC code and missing even one may result in TC with an imperfect error-correcting performance.

The transfer function (TF) of RSC code is a tool that can be used to obtain information about the different codeword weights in the code as well as their corresponding multiplicities, *i.e.*, the distance spectrum. However, the complexity of calculating TF increases with the number of states of RSC code, and other methods such as Mason's Rule [4] must be used. As an added downside, it provides no information regarding the detailed patterns of LWCWs. Research into more efficient methods for finding the distance spectrum of the RSC code has been carried out over the years, including the *fast algorithm for searching a tree* (FAST) algorithm [5] and the *bidirectional efficient algorithm for searching trees* (BEAST) algorithm [6]. Even though these algorithms can be used to obtain the multiplicity of the LWCWs, they do not reveal their patterns. However, it is possible to adapt these algorithms to reveal the LWCW patterns because they use the RSC code state diagram in some form, though the required complexity and accuracy depends on the specific algorithm

Perhaps, due to this possibility, there are very few, if any studies, where discussion concerning the detailed patterns of LWCWs for RSC codes is considered independent of interleaver design for TCs, as in [7], [8]. However, in [7], [8], there is rarely any discussion with respect to LWCWs generated as a result of input patterns of Hamming weight 3 and 4, which may be of importance for certain RSC codes, such as the $(5/7)_8$ RSC code. This is evident in the interleaver design method used in [8]. To the best of our knowledge, there is no systematic method that provides information about the LWCW patterns for a given RSC code that does not require the use of a state diagram.

In this paper, we propose a novel design framework for TCs based on complete knowledge of LWCWs of rate $r = 1/2$ RSC codes. First, we develop a method to determine the detailed patterns for LW input and LWPC sequences of Hamming weight up to 4. The complexity of this proposed method is independent of the number of states of RSC code.

For the actual construction of the TC, we select RSC codes based on the proposed method and present the expression of d_f for the resultant TC. Then, we propose a novel interleaver class, named *coset interleaver*, with its parameter selection criteria to enlarge d_f. Based on the proposed design framework, for frame size $N = 1024$, we succeed in designing a rate $1/3$ TC with $d_f = 42$. Compared to S-random [7] ($d_f = 21$) and Quadratic Permutation Polynomial (QPP) [8] ($d_f = 30$) interleavers, our design suppresses the error floor level to approximately 10^{-2} and 3×10^{-3}, respectively, at $E_b/N_0 = 2$dB.

II. Preliminaries

A polynomial in x with degree M is an expression of the form

$$v(x) = \sum_{m=0}^{M} v_m x^m, \tag{1}$$

where $v_m,\ 0 \le m \le M$ are called the *coefficients* and $v_M \neq 0$. If $v_M = 1,\ v(x)$ is called a *monic* polynomial. We say the total number of the non-zero coefficients of $v(x)$ is the *Hamming weight* of $v(x)$, denoted as $w_H(v(x))$.

For a prime number p, if the addition and multiplication of two elements in the integer set $\{0, 1, \cdots, p-1\}$ are performed on the terms $\mod p$, we call the set a Galois field, denoted as $\mathrm{GF}(p)$. If the coefficients in (1) are elements of $\mathrm{GF}(p)$, $v(x)$ is called a *polynomial over* $\mathrm{GF}(p)$.

For two polynomials $v(x)$ and $w(x)$ with degrees M and N, respectively, the addition and multiplication over $\mathrm{GF}(p)$ are defined as

$$v(x) + w(x) = \sum_{m=0}^{\max\{M,N\}} [v_m + w_n]_p x^m, \tag{2}$$

and

$$v(x)w(x) = \sum_{m=0}^{M+N} \sum_{i=0}^{m} [v_i w_{m-i}]_p x^m, \tag{3}$$

respectively.

We say a monic polynomial is a *prime polynomial* if it cannot be represented by multiplication of some lower-degree polynomials. For two polynomials $v(x)$ and $w(x)$ over $\mathrm{GF}(p)$, $w(x) \neq 0$, there exists polynomials $q(x)$ and $r(x)$ over $\mathrm{GF}(p)$ such that

$$v(x) = w(x)q(x) + r(x), \tag{4}$$

with $\deg(r(x)) < \deg(w(x))$. We represent $r(x)$ in the expression (4) as

$$r(x) \equiv v(x) \mod w(x), \tag{5}$$

and call it the *remainder polynomial*, while $q(x)$ is called the *quotient polynomial* of the division of $v(x)$ by $w(x)$.

Let $v(x)$ be a prime polynomial over $\mathrm{GF}(p)$ with $\deg(v(x)) = M > 1$ and $\mathcal{V}(x)$ be the polynomial set of size p^M containing all polynomials over $\mathrm{GF}(p)$ with degree less than M. Then, the *extension field of* $\mathrm{GF}(p)$, denoted by $\mathrm{GF}\left(p^M\right)$, is the set $\mathcal{V}(x)$ with addition and multiplication over $\mathrm{GF}(p)$, where the multiplication is carried out modulo-$v(x)$ over $\mathrm{GF}(p)$.

For each non-zero element $x \in \mathrm{GF}\left(p^M\right)$, there exist integers ϵ such that $x^\epsilon = 1$ and the least positive integer among them is called the *order* of x. We say that elements with order $\epsilon = p^M - 1$ are *primitive elements*. For $\mathrm{GF}\left(p^M\right)$ generated by a prime polynomial $v(x)$ with $\deg(v(x)) = M$, if x is a primitive element in $\mathrm{GF}\left(p^M\right)$, then $v(x)$ is called a *primitive polynomial*. Finally, the root of $v(x)$ is the non-zero element $\varphi \in \mathrm{GF}\left(p^M\right)$, such that $v(\varphi) = 0$. If $v(x)$ is a primitive polynomial, the order of φ is $\epsilon = p^M - 1$, otherwise ϵ is a division of $p^M - 1$, denoted by $\epsilon | p^M - 1$. Moreover, the elements $\varphi^i,\ 0 \le i \le \epsilon - 1$, are all distinct from each other.

III. Component Code Selection for TCs

In this section, we introduce a systematic method to determine the detailed patterns of the input sequences that are most likely to produce LWCWs based on the generator function of RSC code.

A. The Characteristics of LWCWs

Assuming a tail-biting scheme, *i.e.*, codes terminated in such a way that the initial and final states are the same after N information bits are transmitted, without using any extra bits, we consider a rate $r = 1/2$ RSC code for the sake of simplicity. Then, the outputs of any RSC code are determined by the input sequence $b(x)$, states of the shift register, and feedforward and feedback connections of the shift register determined by its generator function, which can be written as

$$\boldsymbol{G} = \left[1 \ \frac{f(x)}{g(x)}\right], \tag{6}$$

where 1 replicates $b(x)$, and the output bit sequence $h(x)$, called the *parity-check* (PC) sequence, is associated with the feedforward and feedback connections of the shift registers, specified by $f(x)$ and $g(x)$, respectively. For a given $\boldsymbol{G}$, the RSC code outputs are a mixture of the input and PC sequences as

$$c(x) = b(x^2) + xh(x^2), \tag{7}$$

where

$$h(x) = f(x)g^{-1}(x)b(x). \tag{8}$$

From (7), it is trivial that

$$w_H(c(x)) = w_H(b(x)) + w_H(h(x)), \tag{9}$$

and hence, each LWCW is a combination of LW input and LWPC sequences.

RSC codes posses an infinite impulse response characteristic, and under the assumption of large frame sizes, the presence of $g^{-1}(x)$ in (8) may involve a particular input sequence that repeats a large number of times, thus yielding a high-weight PC sequence. The LWPC sequence occurs if and only if

$$b(x) \bmod g(x) \equiv 0. \tag{10}$$

Any $b(x)$ satisfying (10) is called a *return-to-zero* (RTZ) input sequence. Obviously, each LW input sequence is a RTZ input sequence and can be factorized as:

$$b(x) = a(x)g(x), \tag{11}$$

and substituting (11) into (8), we can characterize the LWPC sequence as

$$\begin{aligned} h(x) &= f(x) \cdot g^{-1}(x) \cdot a(x)g(x) \\ &= a(x)f(x). \end{aligned} \tag{12}$$

In this paper, we attempt to find $a(x)$ values that satisfy (11) and (12) simultaneously for the LW input and LWPC sequences, respectively. However, since there is no essential mathematical difference between them, in this section we present a method for determining the LW input sequence patterns for $w_H(b(x)) = 2, 3$, and 4.

B. The Pattern Identification of LW Input Sequence

Let $g(x)$ be a polynomial over GF(2) and we assume it can be factorized into K prime polynomials over GF(2) as:

$$g(x) = \prod_{k=0}^{K-1} g_k^{\gamma_k}(x), \tag{13}$$

where $\gamma_0, \gamma_1, \cdots, \gamma_{K-1}$ are positive integers and let φ_k be a root of $g_k(x)$ of order ϵ_k.

Now, starting from the simplest case, $K = 1$, *i.e.*, $g(x) = g_0^{\gamma_0}(x)$, we consider the solution of

$$b(x) \mod g(x) \equiv 0. \tag{14}$$

Then we know from (11) that each root of $g(x)$ is also a root of $b(x)$ and we distinguish the cases $\gamma_0 = 1$ and $\gamma_0 > 1$. For the former case, since all φ_0^i, $0 \leq i < \epsilon_0$, are distinct from each other, the equation

$$b(\varphi_0^i) = 0, \quad 0 \leq i < \epsilon_0, \tag{15}$$

is the necessary and sufficient condition of (14), wheareas it is necessary but not sufficient for the latter case. Thus, for the case $\gamma_0 > 1$, we obtain extra conditions using differential equations as

$$\left.\frac{d^{(j)}b(x)}{dx^j}\right|_{x=\varphi_0^i} = 0, \quad 0 \leq i < \epsilon_0,\ 1 \leq j < \gamma_0, \tag{16}$$

where the derivation is calculated using the *Hasse derivative* defined as

$$\frac{d^j x^k}{dx^j} = \begin{cases} [{}_kC_j]_2\, x^{k-j}, & k \geq j \\ 0, & \text{otherwise} \end{cases}, \tag{17}$$

for the binomial coefficient ${}_kC_j$, where $[\alpha]_\beta$ denotes the remainder α divided by β.

For the case where $K > 1$, we may repeat the above solver for the roots φ_k, $0 < k < K$, and take the intersection of the results.

1) The Weight-2 LW Inputs: Each *weight-2 LW* (W2LW) input can be written as

$$b_2(x) = x^u\left(1 + x^\alpha\right), \quad p \in \mathbb{Z}^+, \tag{18}$$

without loss of generality. Thus, we have from (15) that

$$(\varphi_0^i)^\alpha = 1, \quad 0 \leq i < \epsilon_0, \tag{19}$$

and since the order of φ_0 is the least integer satisfying $\varphi_0^{\epsilon_0} = 1$, α should satisfy the condition $[\alpha]_{\epsilon_0} = 0$. Thus,

$$b_2(x) = x^u\left(1 + x^{p\tau}\right), \tag{20}$$

where $\alpha = p\tau,\ \tau = \epsilon_0$. The multiplicity of the W2LW inputs starting from position $u = 0$ is given by

$$N_2 = \left\lfloor \frac{N}{\tau} \right\rfloor. \tag{21}$$

2) The Weight-3 LW Inputs: Without loss of generality, the *weight-3 LW* (W3LW) inputs can be written as

$$b_3(x) = x^v\left(1 + x^\alpha + x^\beta\right),\ \alpha < \beta \tag{22}$$

and hence, (α, β) should satisfy the condition

$$\varphi_0^\alpha + \varphi_0^\beta = 1. \tag{23}$$

The pairs (α, β) satisfying (23) can be determined by referring to the table of the extended field for GF $\left(2^M\right)$. Let (m, n) be such a pair and let $\mathbb{M} = \{\epsilon_0\ell + m\}_{\ell\in\mathbb{Z}}$ and $\mathbb{N} = \{\epsilon_0\ell + n\}_{\ell\in\mathbb{Z}}$ be two indexed sets. Then, it is obvious that each pair $(\alpha, \beta) \in \mathbb{M} \otimes \mathbb{N}$ satisfies (23). For a fixed α, on the other hand, since $\alpha + \ell$, $0 \leq \ell < \epsilon_0$, are distinct from each other, any integer β that satisfies (23) must be such that $[\beta]_{\epsilon_0} = n$. Letting $v = 0$, then the multiplicity of W3 inputs can be expressed by

$$N_3 = |\mathbb{M} \otimes \mathbb{N}| \sum_{\alpha=1}^{N_2-1} (N_2 - \alpha) = |\mathbb{M} \otimes \mathbb{N}| \frac{N_2(N_2-1)}{2}, \tag{24}$$

where $\mathbb{M} \otimes \mathbb{N}$ denotes the *tensor product* between $\mathbb{M}$ and $\mathbb{N}$.

3) The Weight-4 LW Inputs: By representing the *weight-4 LW* (W4LW) inputs as

$$b_4(x) = x^u\left(1 + x^{p\tau}\right) + x^v\left(1 + x^{q\tau}\right), \tag{25}$$

for $u < v$, we consider the following equation

$$x^u\left(1 + x^{p\tau}\right) = x^v\left(1 + x^{q\tau}\right). \tag{26}$$

Letting $u = 0$ for simplicity. Then, the equation (26) holds if $1 + x^{p\tau} = 1 + x^{q\tau} = 0$ or $1 + x^{p\tau} \in \mathrm{GF}(p^M)$. For the former case, the solution is trivial as $[p\tau]_{\epsilon_0} = [q\tau]_{\epsilon_0} = 0$. For the latter case, on the other hand, the equation implies either $x^v = 1$ or $x^{v+q\tau} = 1$, and by exchanging variables, can be formulated to the former case. Thus, the W4LW inputs can be represented as

$$b_4(x) = b_2(x) + b_2'(x), \tag{27}$$

where $b_2'(x) = x^v\left(1 + x^{q\tau}\right)$. The multiplicity for $u = 0$ is bounded by

$$N_4 < NN_2^2. \tag{28}$$

While W4SCs of the form in (27) always exist, there might exist those that do not conform to (27) depending on $g(x)$. Fortunately, such W4LW inputs can be avoided by selecting $g(x)$ such that $x^\alpha + x^\beta + x^\eta \neq 1$, where $0 < \alpha, \beta, \eta < \epsilon_0$ and are distinct from each other.

TABLE I: Polynomial examples

Example	K	γ	type	$g(x)$	order(s) of root(s)	period τ
1	1	$\gamma_0 = 1$	primitive	$g_0(x) = 1 + x + x^2$	$\epsilon_0 = 3$	3
2			prime	$g_0(x) = 1 + x + x^2 + x^3 + x^4$	$\epsilon_0 = 5$	5
3		$\gamma_0 = 2$	primitive	$g_0^2(x) = (1 + x)^2 = 1 + x^2$	$\epsilon_0 = 1$	2
4		$\gamma_0 = 4$	primitive	$g_0^4(x) = (1 + x)^4 = 1 + x^4$	$\epsilon_0 = 1$	4
5	2	$\gamma_0 = 1$	primitive	$g_0 = 1 + x$	$\epsilon_0 = 1$	7
		$\gamma_1 = 1$	primitive	$g_1 = 1 + x^2 + x^3$	$\epsilon_1 = 7$	

C. Examples

We demonstrate our method for the polynomials shown in Table I, which take the form in (13).

1) W2LW Inputs: We determine the period τ, with the values shown in Table I. For the cases $\gamma_0 > 1$, we solved the following second order differential equation

$$\frac{db(x)}{dx} = [p\tau]_2 x^{p\tau-1} = 0, \tag{29}$$

in Example 4, while the following additional equations are obtained for Example 5 as

$$\begin{cases} \frac{d^2b(x)}{dx^2} = \left[\frac{p\tau(p\tau-1)}{2}\right]_2 x^{p\tau-2} = 0 \\ \frac{d^3b(x)}{dx^3} = \left[\frac{p\tau(p\tau-1)(p\tau-2)}{6}\right]_2 x^{p\tau-3} = 0. \end{cases} \tag{30}$$

For the case $K > 1$, τ is determined using the least common divisor, *i.e.*, $\tau = \mathrm{lcm}(\epsilon_0, \epsilon_1)$.

2) W3LW Inputs: The W3LW inputs are identified by referring to Table II.

TABLE II: Elements of GF(2^M)

$g(x)$	$1 + x + x^2$	$1 + x^2 + x^3$	$1 + x + x^2 + x^3 + x^4$
Power	Polynomial		
x^0	1	1	1
x	x	x	x
x^2	$1 + x$	x^2	x^2
x^3		$1 + x^2$	x^3
x^4		$1 + x + x^2$	$1 + x + x^2 + x^3$
x^5		$1 + x$	
x^6		$x + x^2$	

For Example 1, it can be confirmed that there is a pair $(1, 2)$ satisfying $x^1 + x^2 \equiv 1 \mod g(x)$. Thus, if we let $\mathbb{M} = \{3\ell + 1\}_{\ell \in \mathbb{Z}}$ and $\mathbb{N} = \{3\ell + 2\}_{\ell \in \mathbb{Z}}$, $x^\alpha + x^\beta \equiv 1 \mod g(x)$ for each pair $(\alpha, \beta) \in \mathbb{M} \otimes \mathbb{N}$. Therefore the W3LW inputs can be expressed as

$$b_3(x) = 1 + x^{3\ell+1} + x^{3\ell'+2}, \quad \ell, \ell' \in \mathbb{Z}.$$

For Examples 2-4, there is no pair (m, n) satisfying $x^m + x^n \equiv 1$, hence no W3LW inputs are associated with the $g(x)$'s.

In Example 5, the feedforward polynomial is written as $g(x) = g_0(x)g_1(x)$ and the W3LW inputs are determined by taking the intersection of those obtained from $g_0(x)$ and $g_1(x)$. In Example 5, there are no W3LW inputs since $g_0(x)$ does not yield any W3LW inputs.

3) W4LW Inputs: The W4LW inputs can be easily obtained from the W2LW inputs. W4LW inputs that do not conform to (27) exist for Example 5.

D. Selection of RSC Code

The selection of RSC codes proper as TC components begins with determining $g(x)$. We wish to utilize $g(x)$ without W3LW inputs and, from the analysis and examples in Section III, the W3LW inputs can be avoided if $g(x)$ contains the factor $(1 + x)$. Once $g(x)$ is determined, we can calculate $a_2(x) = b_2(x)/g(x)$ and select $f(x)$ to maximize the weight for $h_2(x) = a_2(x)f(x)$ within the period τ.

Example 6. Consider $g_1(x) = 1 + x + x^2$ and $g_2(x) = 1 + x^4$. According to the guidelines above, it is obvious that the complexity required to design an interleaver for $g_2(x)$ is less and $g(x) = g_1(x)$ is chosen. The possible valid polynomials for $f(x)$ are $1 + x + x^4$, $1 + x^2 + x^4$, $1 + x^3 + x^4$ and $1 + x + x^2 + x^3 + x^4$. It is easily confirmed that $f(x) = 1 + x + x^2 + x^3 + x^4$ is the most suitable polynomial, and, the selected values of $f(x)$, $g(x)$ yield the $(37/21)_8$ RSC code.

E. Weight Expression of PC Sequence

For the selected RSC codes, the complete knowledge about the LW input patterns allow us to obtain the weight expression of the corresponding PC sequences. Thus, we can design an interleaver by taking into account the weight of the LWCWs for the TC.

For the $(37/21)_8$ RSC code, the PC sequence for W2LW inputs is given by

$$h_2(x) = a_2(x)f(x) = \sum_{i=0}^{p-1} x^{i\tau}\left(1 + x + x^2 + x^3 + x^4\right), \tag{31}$$

and its weight can be expressed by

$$w_H(h_2(x)) = 3p + 2. \tag{32}$$

whereas the PC sequence weight for the W4LW inputs in (25), without proof due to limited space, is given as

$$w_H(h_4(x)) \geq \begin{cases} 3(p + q) + 4 & \text{separable} \\ 2\max\{p, q\} & \text{unseparable} \end{cases}, \tag{33}$$

where the W4LW input is called *unseparable* if $b_2(x)$ and $b_2'(x)$ in (27) overlap, *i.e.*, $v < p$ and *seperable* otherwise.

Having obtained the weight expression of PC sequences, we proceed to design an interleaver considering the possible catastrophic weight reduction in (33).

IV. Interleaver Design for TC

Interleaver design, which is the final step in our novel design framework, aims to maximize d_f for the TC. We briefly introduce the *coset interleaver*, and for the $(37/21)_8$ RSC code with $\tau = 4$, we design it to break as many W2LW and W4LW inputs as possible, yielding a rate $r = 1/3$ TC with $d_f = 42$.

A. Coset Interleaver Overview

For a length-N bit sequence $\boldsymbol{b}$, we introduce the *integer array* (IA) representation of an interleaver. In the IA representation, a length-N interleaver is simply specified by an indexed set $\mathbb{N}$, where the ith entry of $\mathbb{N}$, n_i specifies the interleaving position of the ith bit, *i.e.*, $d_{n_i} = b_i$, to generate the interleaved bit sequence $\boldsymbol{d} = (d_{n_i})_{i=0}^{N-1}, n_i \in \mathbb{N}$.

To break as many LW inputs as possible for a RSC encoder with period τ, we assume that N is a multiple of τ and let

$$\mathcal{M}^c = \tau\mathcal{M} + c \tag{34}$$

for $M := N/\tau$, where $\tau = 4$ throughout this paper unless otherwise stated. Then $\mathcal{N}$ can be expressed as a union of disjoint cosets as $\mathcal{N} = \cup_{c=0}^{\tau-1}\mathcal{M}^c$, and thus, we can generate an interleaver by specifying the union and ordering rules for the entries in $\mathcal{M}^c$. To this end, we let $\mathbb{M}^c$ be an indexed set consisting of the entries of $\mathcal{M}^c$ and introduce a *permutation matrix* (PM) $\boldsymbol{\Pi}_{\tau\times\tau}$, $\pi_i^j \in \{0, 1, \cdots \tau - 1\}$ for $i, j \in \{0, 1, \cdots \tau - 1\}$. The resulting interleaver is called the *coset interleaver*.

The PM determines the order of the cosets $\mathbb{M}^t$ from which to pick elements to form $\mathbb{N}$ and it is used in a cyclic manner while the aim of coset ordering design is to maximize the LWCW weight for the W2LW and W4LW inputs. For sake of simplicity, we assume M, $T := M/\tau$, and $S = T/(\tau + 1)$ are integers, and let $m = m_1\tau + m_0$ for $0 \le m_1 < T$ and $0 \le m_0 < \tau$. Then, the value of the mth entry is specified by

$$e_m^{(c)} = \tau\varphi^{(c)}(m) + c. \tag{35}$$

Example 7. For a bit sequence of length $N = 32$, $\mathcal{N} = \{0, 1, 2, \cdots, 31\}$, $\tau = 4$ and $\mathcal{M} = \{0, 1, \cdots, 7\}$. Then

$$\mathcal{M}^0 = \{0, 4, .8, \cdots, 28\},\ \mathcal{M}^1 = \{1, 5, 9, \cdots, 29\},$$
$$\mathcal{M}^2 = \{2, 6, 10, \cdots, 30\},\ \mathcal{M}^3 = \{3, 7, 11, \cdots, 31\}. \tag{36}$$

Let

$$\tilde{\boldsymbol{\Pi}} = \begin{bmatrix}\boldsymbol{\Pi}\\ \boldsymbol{\Pi}\end{bmatrix},\ \boldsymbol{\Pi} = \begin{bmatrix}0 & 1 & 2 & 3\\ 1 & 2 & 3 & 0\\ 2 & 3 & 0 & 1\\ 3 & 0 & 1 & 2\end{bmatrix}, \tag{37}$$

and $e_m^{(c)} = \tau\,[3m]_M + c,\ m = \{0, 1, \cdots, M-1\}$.
Then

$$\begin{aligned}\mathbb{M}^0 &= \{0, 12, 24, 4, 16, 28, 8, 20\},\\ \mathbb{M}^1 &= \{1, 13, 25, 5, 17, 29, 9, 21\},\\ \mathbb{M}^2 &= \{2, 14, 26, 6, 19, 30, 10, 22\},\\ \mathbb{M}^3 &= \{3, 15, 27, 7, 20, 31, 11, 23\},\end{aligned} \tag{38}$$

and

$$\begin{aligned}\mathbb{N} = \{&0, 1, 2, 3, 13, 14, 15, 12, 26, 27, 24, 24, 7, 4, 5, 6, 16,\\ &17, 18, 19, 29, 30, 31, 28, 10, 11, 8, 9, 23, 20, 21, 22\}.\end{aligned} \tag{39}$$

B. PM Design

$$\tilde{\boldsymbol{\Pi}} = \begin{bmatrix}\boldsymbol{\Pi}\\ \vdots\\ \boldsymbol{\Pi}\end{bmatrix} = \left[\begin{array}{ccccc} \overbrace{\pi_0^0 \quad \cdots \quad \pi_{u_0}^0}^{u_0} & & & \cdots & \pi_{\tau-1}^0\\ \vdots & \cdots & \vdots & \cdots & \vdots\\ \pi_0^{u_1} & \cdots & \boxed{\pi_{u_0}^{u_1}} & \cdots & \pi_{\tau-1}^{u_1}\\ \vdots & \cdots & \vdots & \cdots & \vdots\\ \pi_0^{\tau-1} & \cdots & \pi_{u_0}^{\tau-1} & \cdots & \pi_{\tau-1}^{\tau-1}\\ \hline \vdots & \cdots & \vdots & \cdots & \vdots\\ \hline \pi_0^{p_2\tau} & \cdots & \pi_{u_0}^{p_2\tau} & \cdots & \pi_{\tau-1}^{p_2\tau}\\ \vdots & \cdots & \vdots & \cdots & \vdots\\ \pi_0^{p_2\tau+p_1+u_1} & \cdots & \boxed{\pi_{u_0}^{p_2\tau+p_1+u_1}} & \cdots & \pi_{\tau-1}^{p_2\tau+p_1+u_1}\\ \vdots & \cdots & \vdots & \cdots & \vdots \end{array}\right] \begin{array}{l} \left.\vphantom{\begin{array}{c}a\\a\\a\end{array}}\right\} u_1\\ \left.\vphantom{\begin{array}{c}a\\a\\a\\a\\a\\a\end{array}}\right\} p \end{array}$$

Fig. 1: General Permutation Matrix

The PM is designed by considering the breaking of W2LW inputs. For (20), let $u = u_2\tau^2 + u_1\tau + u_0$ for $0 \le u_0, u_1 < \tau$ and $p = p_2\tau + p_1$, $0 \le p_1 < \tau$. The post-interleaving cosets can then be found in the manner shown in Fig. 1 and a W2LW input is mapped to another if and only if the same elements appear in the same column. Therefore, the distance between the elements in the same column should be made as large as possible. This means that each column should be a permutation of $\{0, 1, \cdots, \tau - 1\}$ and we select the simplest PM, which is

$$\boldsymbol{\Pi} = \begin{bmatrix}0 & 0 & 0 & 0\\ 1 & 1 & 1 & 1\\ 2 & 2 & 2 & 2\\ 3 & 3 & 3 & 3\end{bmatrix}. \tag{40}$$

C. Coset Ordering Design

Consider

$$\varphi^{(c)}(m) = \left[mD + \psi^{(c)}(m_0)\right]_M, \tag{41}$$

where D and

$$\psi^{(c)}(m_0) := [(c+1)(m_0+1)]_{\tau+1}R, \tag{42}$$

are called the *depth* and *bias* of the cth coset, respectively, and we let $R = \tau S$. Thus, the ordering rule is expressed as follows:

$$e_m^{(c)} = \tau\left[mD + [(c+1)(m_0+1)]_{\tau+1}R\right]_M + c. \tag{43}$$

Based on the PM, the condition for a W2LW input to be mapped to another is given by

$$\begin{aligned} b_2(x) &= 1 + x^{p_2\tau^2}\\ b_2'(x) &= x^{e_0^{(0)}} + x^{e_{p_2\tau}^{(0)}}, \end{aligned} \tag{44}$$

TABLE III: Mapping of W4 Inputs

pattern	$b_4(x)$	$b'_4(x)$
1	$\left(1+x^{p_2\tau^2}\right)+x^{v_2\tau^2+v_1\tau+v_0}\left(1+x^{q_2\tau^2}\right)$	$\left(x^{e_0^{(0)}}+x^{e_{p_2\tau}^{(0)}}\right)+\left(x^{e_{v_2\tau+v_0}^{(v_1)}}+x^{e_{(v_2+q_2)\tau+v_0}^{(v_1)}}\right)$
2	$\left(1+x^{p_2\tau^2+p_1\tau}\right)+x^{v_2\tau^2+v_0}\left(1+x^{q_2\tau^2+p_1\tau}\right)$	$\left(x^{e_0^{(0)}}+x^{e_{v_2\tau+v_0}^{(0)}}\right)+\left(x^{e_{p_2\tau}^{(p_1)}}+x^{e_{v'_2\tau+v_0}^{(p_1)}}\right)$
3	$\left(1+x^{p_2\tau^2+p_1\tau}\right)+x^{v_2\tau^2+p_1\tau+v_0}\left(1+x^{(q_2+1)\tau^2-p_1\tau}\right)$	$\left(x^{e_0^{(0)}}+x^{e_{(v'_2+1)\tau+v_0}^{(0)}}\right)+\left(x^{e_{p_2\tau}^{(p_1)}}+x^{e_{v_2\tau+v_0}^{(p_1)}}\right)$

where $p_2 \geq 1$. Similarly, the conditions for the W4LW inputs are given in Table III, where $v'_1 := v_1 + q_1 - \nu\tau$ and $v'_2 := (v_2 + q_2 + \nu)$ for $\nu := \lfloor (v_1 + q_1)/\tau \rfloor$ and

$$\begin{cases} p_2, q_2 \geq 1,\ p_1 = q_1 = 0 & \text{Pattern 1} \\ p_2\tau + p_1,\ q_2\tau + p_1 > 0,\ q_1 = p_1,\ v_1 = 0 & \text{Pattern 2}\ . \\ p_2\tau + p_1,\ q_2 > 0,\ q_1 = \tau - p_1 \text{ and } v_1 = p_1 & \text{Pattern 3} \end{cases} \tag{45}$$

Based on the mapping relationships, we make the following assertions about the attainability of d_f without proof.

1) For W2LW inputs, $w_H(c_2(x))$ can be made as large as possible via the selection of D.
2) For W4LW inputs, $w_H(c_4(x))$ is also dependent on D and peaks at 42.

Our aim is to select D such that $\min\{w_H(c_2(x)),\ w_H(c_4(x))\}$ is maximized. The simplest way to achieve this is via a computer search, by fixing D and solving (44) and the equations in Table III for $0 \leq p_2,\ q2 \leq \tau,\ 0 < p_1 \leq \tau$ and $0 < v \leq M$.

V. Simulation Results

In this section, we compare the simulation results for a TC using the coset interleaver with that of the S-random interleaver and the QPP interleaver in Figure. 2. For each TC, the interleaver size is $N = 1024$ and the $(37/21)_8$ RSC code is chosen as the component code. The *extended tail-biting with state mapping* method [9] is used to somewhat preserve the tail-biting assumption whereas at the decoder, the end states for each MAP decoder are known to simplify decoding and improve the simulation results. For the coset interleaver, D is found via a computer search, such that $\min\{w_H(c_2(x)),\ w_H(c_4(x))\} \geq 42$, and from a list of suitable candidates, we selected $D = 59$ for use in the computer simulation.

It can be observed that our interleaver significantly outperforms both the S-random and QPP interleavers, and suppress the error floor level to approximately 10^{-2} and 3×10^{-3}, respectively, at $E_b/N_0 = 2$dB. Simulation data confirms that $d_f = 42$ for W4LW and W6LW inputs.

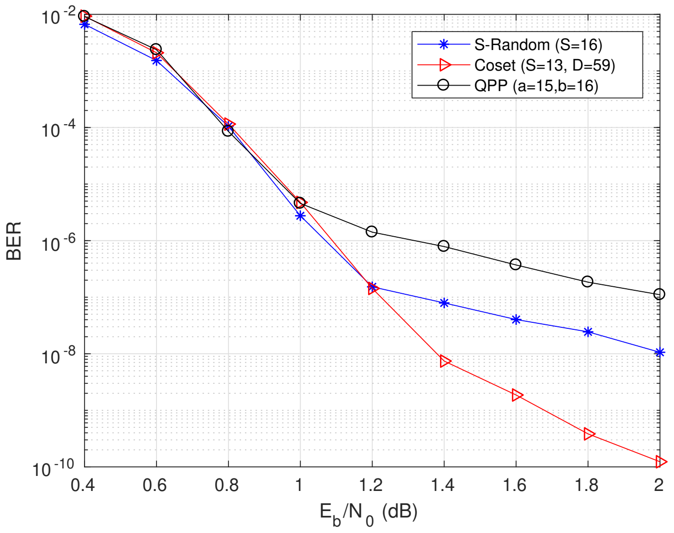

Fig. 2: Simulation Results for S-random, QPP and Coset Interleaver for $N = 1024$

VI. Conclusion

In this paper, we introduced a low-complexity method to obtain detailed patterns of LW inputs of Hamming weight up to 4 for a rate $r = 1/2$ RSC code with a tail-biting scheme. The novel method is applied to RSC component code selection for TC and the coset interleaver is introduced, which yields a TC with $d_f = 42$ and outperforms both the QPP and S-random Interleavers for $N = 1024$.

References

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo-codes." in *Proc. IEEE Int. Conf. Commun. (ICC'93)*, vol. 2, Geneva, Switzerland, May 1993, pp. 1064–1070.

[2] *IEEE Standard for Local and Metropolitan Area Networks. Part 16: Air Interface for Fixed Broadband Wireless Access Systems*, IEEE std 802.16-2004 Std., Rev. of IEEE 802.16-2001, Aug. 2004.

[3] *LTE Envolved Universal Terrestrial Radio Access (E-UTRA): Multiplexing and Channel Coding*, Third Generation Partnership Project (3GPP) Std., Jan. 2011.

[4] T. K. Moon, *Error Correcting Codes*. Hoboken, NJ, USA: Wiley, 2005.

[5] M. Cedervall and R. Johannesson, "A fast algorithm for computing distance spectrum of convolutional codes," *IEEE Trans. Inf. Theory*, vol. 35, no. 6, pp. 1146–1159, 1989.

[6] I. Bocharova, M. Handlery, R. Johannesson, and B. Kudryashov, "A beast for prowling in trees," *IEEE Trans. Inf. Theory*, vol. 50, no. 6, pp. 1295–1302, 2004.

[7] S. Dolinar and D. Divsalar, "Weight distributions for turbo codes using random and nonrandom permutations," *Technical report, JPL TDA Progress Report*, vol. 42-122, no. 8, pp. 56–65, 1995.

[8] J. Sun and O. Y. Takeshita, "Interleavers for turbo codes using permutation polynomials over integer rings," *IEEE Trans. Inf. Theory*, vol. 51, no. 1, pp. 101–119, Jan. 2005.

[9] ——, "Extended tail-biting schemes for turbo codes," *IEEE Commun. Lett.*, vol. 9, no. 3, pp. 252–254, Mar. 2005.