On the Analysis and Design of Good Algebraic Interleavers

Kai Xie[1] Wenbo Wang[2] and Jing Li (Tiffany)[1]

[1]Electrical and Computer Engineering Dept, Lehigh University, Bethlehem, 18015
{kax205,jingli}@lehigh.edu
[2]School of Telecom. Engineering, Beijing U of Posts & Telecom., Beijing, P.R.China, 100876
{wbwang}@bupt.edu.cn

Abstract

This paper analyzes, classifies and searches for good coprime interleavers using the recently-developed metric of *cycle correlation sum* (CCS). A subset of good coprime interleavers, referred to as *regular coprime interleavers*, are identified and formulated. It is shown that regular coprime interleavers perform as well as random interleavers, but can be generated on-the-fly using virtually no parameters. Another subset of coprime interleavers, referred to as *linear coprime* interleavers, comprise some of the best (short-length) interleavers including the Golden prime interleaver. Comparison of these coprime interleavers with the Welch-Costas interleavers, the Takeshita-Costello interleavers, random interleavers and S-random interleavers is performed through CCS evaluation, graph presentation and computer simulations.

1 Introduction

The superb performance of turbo and turbo-like codes is achieved not only through concatenating two convolutional codes in a parallel or serial fashion, but also through manipulating sequences with interleavers. In the case of parallel concatenation, for example, the two constituent encoders are working on the same set of information bits but in different bit orders. When a sequence produces a low-weight output on one constituent encoder, its scrambled counter part will, with a high probability, produce a high-weight output on the other. At the decoder, the interleaver helps break up error bursts and de-correlate the reliability information exchanged between the constituent decoders, making the iterative decoding algorithm efficient.

The simplest class of interleavers is row-column interleavers, where interleaving is achieved by reading a sequence into a block of buffers row-wise and reading it out column-wise. Row-column interleavers are cheap to implement, but fail to remove rectangular error patterns that are detrimental to turbo codes.

It is generalized recognized that randomness is essential to the capacity-approaching performance of turbo codes. However, a random or S-random interleaver requires the storage of the entire interleaving pattern, which can be expensive or infeasible for applications that have limited storages, use a very long code or concurrently support several different block lengths. Hence, *algebraic interleavers*, which can be generated on-the-fly using a few parameters and which exhibit reasonable randomness in the interleaving pattern, are preferred in practice. Coarsely speaking, an algebraic interleaver is an interleaver whose scrambling pattern is completely specified by a well-defined mathematical formula with a few seeding parameters[1] [2].

Two classes of algebraic interleavers are particularly worth mentioning in literature. The *Welch-Costas interleavers* make essential use of the Costas array, offer performances comparable to random interleavers, and allow for efficient implementations [3]. One drawback, however, is the high complexity in the design procedure, since searching for a primitive element in the Galois field $GF(N)$ can be nontrivial especially for large N. Further, for many practical interleaver lengths of $N = 2^m$, the Welch-Costas interleavers do not even exist. Another notable class of algebraic interleavers are the *Takeshita-Costello interleavers* [4], which have been proven to possess several desirable properties as random interleavers. However, since its interleaving pattern can not be derived directly from the input indices, an intermediate sequence of length N has to be computed and stored, thus diminishing the storage advantage of a typical algebraic interleaver.

The purpose of this paper is to design and identify good algebraic interleavers which are cheap and easy to generate and which offer performances comparable to or better than random interleavers. Our primary search tool is the *cycle correlation sum* (CCS), an efficient metric developed recently for measuring the goodness of interleavers in turbo codes [1]. Simpler and offering more accurate predictions than the existing metrics including *iterative decoding suitability* (IDS) [7], the cycle correlation sum captures the iterative nature of a turbo decoder by noting the correlation properties between input and output extrinsic messages

The research is supported by the National Science Foundation under Grant No. CCF-0430634, by Army Research Laboratory, by Seagate Technology Inc., and by the Commonwealth of Pennsylvania, Department of Community and Economic Development, through the Pennsylvania Infrastructure Technology Alliance (PITA).

[1]Row-column interleavers can also be treated as a special class of algebraic interleavers.

during a decoding iteration [1]. Since an interleaver that performs well for one turbo code (with specific constituent convolutional codes) will in general also perform well for a class of turbo codes of the same constraint length [1], we will concentrate the search on one sample turbo code, but the search results generalize to the entire class.

For practical considerations such as complexity and storage, we focus our investigation on the rich class of *coprime interleavers* [2], which allow on-the-fly generation using two seeding parameters. The coprime interleavers as proposed in [2] were formulated in recursive (and hence sequential) forms. Here we show how the interleaved position for i, denoted as $\pi(i)$, may be computed directly from i, and thus allowing fast, parallel implementation.

Exploiting the CCS metric, we classify coprime interleavers by their performances, and subsequently formulate the rules for good parameters that will lead to performance on par with or better than random interleavers. To complement the CCS evaluation, we further compare coprime interleavers with Welch-Costas interleavers, Takeshita-Costello interleavers, random interleavers and S-random interleaver through graph representation and computer simulations.

The remainder of the paper is organized as follows. We begin with a brief introduction to several typical classes of algebraic interleavers in Section 2. Section 3 provides a concise discussion of the CCS metric along with simulations to demonstrate its efficiency. Section 4 uses CCS to analyze algebraic interleavers at large and the coprime interleavers, the Welch-Costas interleavers and the Takeshita-Costello interleavers in particular. We then propose a set of rules for generating good coprime interleavers and confirm it using graph representation and simulations in Section 5. Finally, Section 6 concludes this paper.

2 Common Classes of Algebraic Interleavers

A length-N interleaver is a single-input single-output device that provides a one-to-one mapping of an alphabet set $A \equiv \{0, 1, ..., N-1\}$ to itself. Let π and π^{-1} denote interleaving and its reverse operation (known as deinterleaving). We say position i is interleavered to position j if

$$\pi(i) = j, \qquad i, j \in A \qquad (1)$$

$$\text{or} \qquad \pi^{-1}(j) = i, \qquad i, j \in A \qquad (2)$$

An algebraic interleaver permutes the elements of its input vector according to an algebraic formula. Below we review a few typical classes of algebraic interleavers that will be used in the discussion.

2.1 Coprime Interleavers

In general, the interleaving pattern of a coprime interleaver is specified recursively as[2]:

$$\begin{cases} \pi(0) &= 0; \\ \pi(i) &= mod\big(a\pi(i-1)+b\,,\,N\big), \\ & \quad \text{for } i = 1, 2, ...N-1; \end{cases} \qquad (3)$$

where N is the interleaver length and $mod(x, N)$ stands for the modulo N arithmetic. To ensure a maximum-length sequence, the parameters a and b should satisfy the following set of rules [2]:

1) $0 < a < N$, $0 \le b < N$, and b be relatively prime to N;
2) $a-1$ be a multiple of p, for every prime p dividing N;
3) $a-1$ be a multiple of 4, if N is a multiple of 4.

Since the value of the starting point $pi(0)$ is not essential to the definition of coprime interleavers, it is set 0 in (3) for convenience. Any other integer will result in a circular shift of the entire interleaving pattern, but poses little impact on the performance. In other words, the performance of a length-N coprime interleaver is determined by a pair of parameters (a, b). To ease discussion in the below, we call the set of coprime interleavers generated from the same value of a a *subclass*.

Since recursion in (3) imposes the constraint for sequential implementation which may cause a long delay, we re-formulate the interleaving rule $\pi(i)$ as a directly function of the indice i to allow for parallel implementation.

When $a \neq 1$, (3) can be rewritten as:

$$\begin{aligned} \pi(i) &= mod(((\sum_{j=0}^{i-1} a^j) \times b), N) \\ &= mod(\frac{(1-a^i)\times b}{(1-a)}, N), \\ &\qquad where \quad i = 0, 1, ..., N-1. \end{aligned} \qquad (4)$$

For the special subclass of $a = 1$, we have

$$\begin{aligned} \pi(i) &= \begin{cases} 0, & i = 0, \\ mod(\pi(i-1)+b, N), & i = 1, 2, ...N-1 \end{cases} \\ &= mod(\,b\,i\,, N), \quad i = 0, 1, ...N-1. \end{aligned} \qquad (5)$$

Since the only parameter b is relatively prime to N, this subclass of coprime interleavers are referred to as *relatively prime interleavers* in [1]. Since the interleaving function $\pi(i)$ is (circularly) linear in i, they are termed *linear interleavers* in [4]. Since these interleavers are "a nonrandom permutation based on circular shifting" [5], they are also known as *circular shifting interleavers*. For consistency with its mother class, here we refer to them as *linear coprime interleavers*.

Furthermore, when $a = 1$ and b is chosen to be the closest integer, which is relatively prime to N, to the Golden section of N, Equation (5) results in the well-known *Golden prime interleaver* [6].

2.2 The Welch-Costas Interleavers

A Welch-Costas interleaver is generated according to the following rule [3]:

$$\pi(i) = mod((a_1^i), N) - 1, \quad i = 0, 1, ...N-1, \quad (6)$$

where $N+1$ is a prime number and a_1 is a primitive element in $GF(N)$. Note that the constraint on N being a prime number minus 1 excludes the possibility for many interleaver lengths. for example, there does not exist Welch-Costas interleavers at length $N = 32$, 64, 128, 512, 1024, 2048, 4096.

2.3 The Takeshita-Costello Interleavers

The generating rule of the Takeshita-Costello interleavers is [4]:

$$C_i = mod((a_2 \times (i-1) \times i/2), N), \quad (7)$$
$$\pi(C_i) = C_{i+1}, \quad (8)$$

where the interleaver length N should be 2^m (m is an integer), and the parameter a_2 should be an odd number smaller than N. As mentioned before, the intermediate sequence $\{C_i\}$ needs to be generated and stored before performing interleaving or de-interleaving.

3 Cycle Correlation Sum (CCS)

This section briefly reviews the CCS metric proposed in [1], and demonstrate its accuracy through simulations.

From the coding theory, the performance of an iterative decoder will approach that of the optimal decoder when the code graph has asymptotically unbounded girth. For practical codes where finite cycles are unavoidable, the requirement translates to minimal correlation between the outbound and inbound messages along any edge in the graph. For turbo decoders, completing any one round of message exchange between the two component decoders will have inevitably introduced or aggravated message correlation. To see this, consider two bits located at i and j in the first component code being interleaved to positions $\pi(i)$ and $\pi(j)$ in the second component code. Bits located at i and j, as well as bits located at $\pi(i)$ and $\pi(j)$, being part of a convolutional codeword, are inherently correlated with each other. Now interleaving and de-interleaving enable the looping back of the same information by completing an 4-edge cycle, where, for example, reliability information originated from bit position i will pass to j through decoding of the first component code, then to $\pi(j)$ through interleaving, to $\pi(i)$ through decoding of the second component code, and finally back to i through de-interleaving. Since looping information undermines the efficiency of iterative decoding and increases the possibility for error propagation, a valid measure for the goodness of an interleaver is its ability to minimize the average amount of information that gets looped back from one decoding iteration to the next, where the average is taken over all the bits in the sequence.

To quantify this measure, [1] proposes to first evaluate the correlation between the input and output extrinsic information of the BCJR decoder using the standard *correlation coefficients*. It is shown in [7] that the correlation coefficient between two bit position's in a convolutional code is a function of the distance between them and can be approximated by an exponential function. Hence, [1] formulates the correlation between bit positions i and j as $e^{-c|i-j|}$, where c is a parameter related to the code. Since the correlation between bit positions $\pi(i)$ and $\pi(j)$ follows a similar form of $e^{-c|\pi(i)-\pi(j)|}$, the correlations induced by cycle $i \to j \to \pi(j) \to \pi(i) \to i$ becomes $e^{-c(|j-i|+|\pi(i)-\pi(j)|)}$. Gathering all such cycle correlations, we obtain the metric of cycle correlation sum [1]:

$$CCS = \sum_{i,j \in A} e^{-c(|j-i|+|\pi(j)-\pi(i)|)} \quad (9)$$

where $A \equiv \{0, 1, 2, ..., N-1\}$, and N is the interleaver length. Parameter c is a function of the component convolutional code, whose value is largely determined by the code's constraint length and may be computed using either simulations or an approximated analytical formula [1]. A smaller value of CCS indicates less message correlation, a higher decoding efficiency, and therefore a better code performance.

Detailed discussion on CCS can be found in [1]. To demonstrate the accuracy of CCS, Figure 1 compares the CCS predictions and their corresponding performances for linear coprime interleavers ($a=1$) at length $N=100$ and 128 bits. For all the possible values of b, the simulated bit error rate (BER) matches remarkably well with the CCS prediction, with a complete and accurate identification of all the worst choices of b (what we should definitely avoid) and a quite accurate identification of the best choices of b (what we wish to attain).

4 Analysis and Classification of Algebraic Interleavers

In this section, we evaluate and classify several types of algebraic interleavers using the CCS metric. Our main results are illustrated in Figure 2.

Figure 2 evaluates the performance of a host of interleavers with length $N = 128$ bits, including coprime interleavers (and the Golden prime interleaver), the Takeshita-Costello interleaver, several random interleavers and S-random interleavers. (Length-128 Welch-Costas interleaver does not exist). The y-axis represents the CCS value. The x-axis represents the value of b for coprime interleavers and the value of a_2 for the Takeshita-Costello interleavers. We tested all the subclasses of coprime interleavers with $a = 4 \times n + 1$,

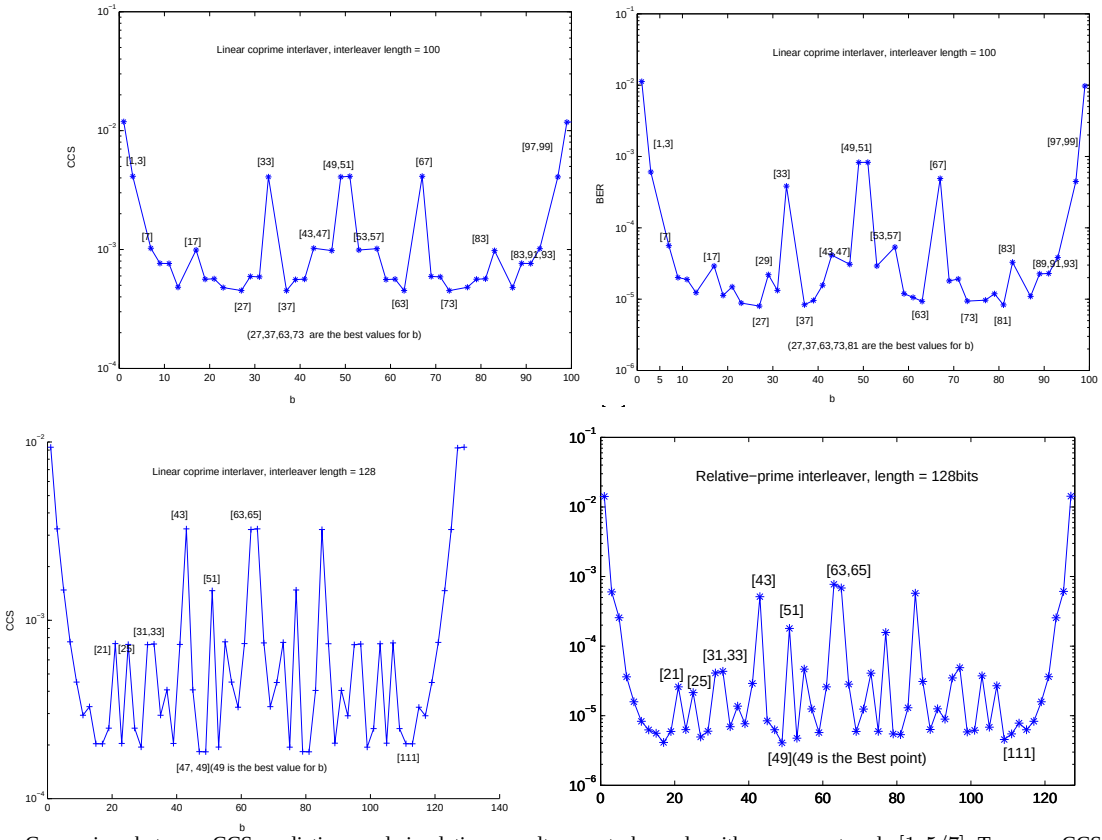

Fig. 1. Comparison between CCS predictions and simulations results on a turbo code with component code $[1, 5/7]$. Top row: CCS prediction and simulated BER of a length 100 linear coprime interleaver; Bottom row: CCS prediction and simulated BER of a length 128 linear coprime interleaver. Evaluating SNR=3.0dB.

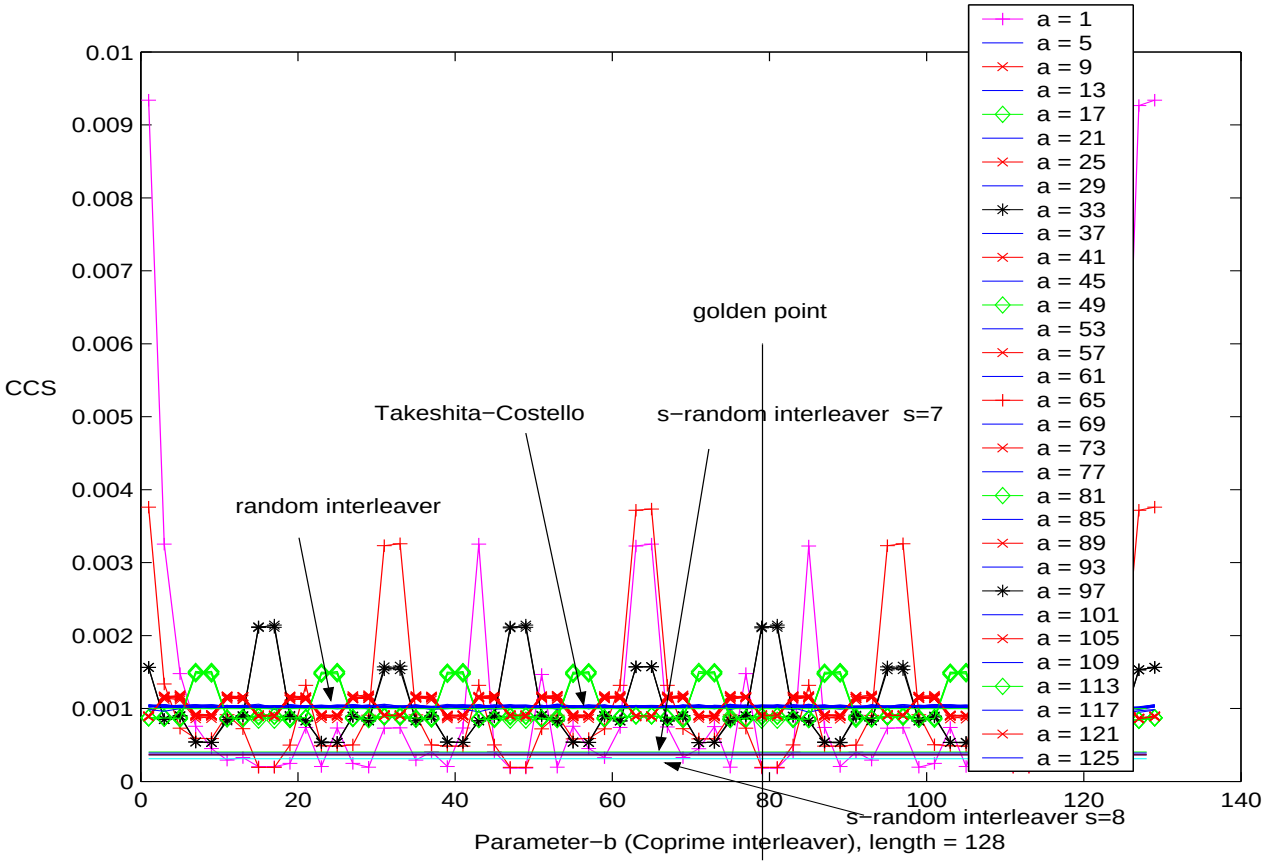

Fig. 2. The CCS values of coprime interleavers, random interleavers, S-random interleavers and the Takeshita-Costello interleavers. $N = 128$.

$0 \leq n < 32$ and all odd values of b. Different values of a are marked with different line type.

Let us start with S-random interleavers whose performances are delineated by the set of straight horizontal lines located at CCS=0.00035 to 0.0004. From the plot, most of these straight lines are hugging around CCS=0.0004 and form one thick line. They correspond to the five S-interleavers we found with spread factor $s = 7$. The thin line slightly below them at CCS=0.00035 is an S-interleaver with $s=8$. Since the spread factor is upper bounded by $\sqrt{2N}$ for a length N S-random interleaver, these interleavers we tested are about the best S-random interleavers of length 128.

Next, look at the bundle of blue horizontal lines at around CCS=0.001 in Figure 2. They correspond to the five random interleavers we tested (generated randomly), the set Takeshita-Costello interleavers generated using (7) and (8) with different values of a_2, and several subclasses of coprime interleavers. First, the performances of the Takeshita-Costello interleavers are not sensitive to the parameter a_2 (denoted by the x-axis) and fall right in the random interleaver region according to CCS. This confirms the claim that they are structured interleavers but behave like random interleavers [4]. Similar results of the Welch-Costas interleavers (i.e. perform similar to random interleavers and insensitive to a_1) are obtained for interleaver length of 100 bits, but the plot is omitted due to the space limitation. Third, the subclasses of coprime interleavers that fall in this performance category have $a=5, 13, 21, ...125$. Unlike other subclasses, the performances of these coprime interleavers are consistently close to that of random interleavers regardless of the value of b. It is remarkable to note that this observation is not unique to length $N = 128$. In general, it appears that for any given length N, *there exists subclasses of coprime interleavers which perform unanimously close to random interleavers.* These subclasses, thereafter referred to as *regular coprime interleavers*, are determined by a single parameter a (provided that b is coprime with N). From extensive tests, when $N = 2^m$, the subclasses having $a=8k-3$ where $k=1, 2, ..., N/8$ form regular prime interleavers.

In addition, we observe that coprime interleavers can be classified in several categories in accordance to their ensemble CCS values. For the case of $N=128$ shown in Figure 2, regular coprime interleavers clearly form one category. The subclasses with $a = 9, 25, ..., 8k+1, ..., 121$ (marked with red cross) form a second category, whose CCS values are either slight above or slight below that of random interleavers depending on b. Then there is the category with $a = 17, 49, 81, 113$ (marked with green diamonds), whose performances deviate more noticeably with b. Finally, the subclasses of $a=1$ and 65 (marked with red plus signs) see the largest performance variation with respect to b. These subclasses consist of a hybrid of "extreme" interleavers, i.e. the worst coprime interleavers that lag far behind the others and the best coprime interleavers that can outperform random and S-random interleavers. We are unable to formulate a rule for the desirable choices of b, but the Golden prime interleaver with $a = 1$ and $b=0.618 \times N=79$ is certainly one good example.

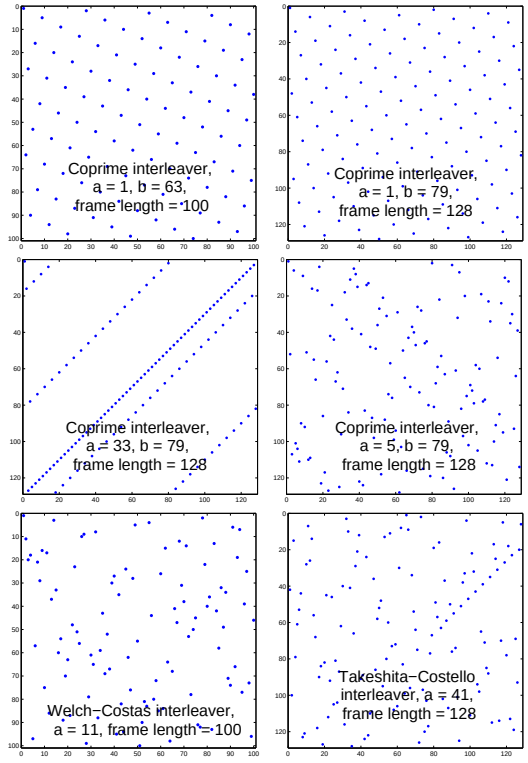

Fig. 3. Scatter-plot representation for interleavers with $N = 100$ and 128.

To summarize, we have the following major results:

1) The ensemble of coprime interleavers comprises different subclasses parameterized by a. In general, the interleaver performances in each subclass are also dependent on b. However, some subclasses exhibit a quite strong dependence while some others appear rather insensitive.
2) One important subclass is the linear coprime interleavers where $a=1$. Despite its simplicity, it consists of some of the best coprime interleavers which can outperform random interleavers and S-random interleavers (for short lengths) [1] (e.g. the Golden prime interleaver). Since it also consists of some of the worst interleavers, the value of b should therefore be chosen with caution.
3) There exist several subclasses of coprime interleavers, referred to as *regular coprime interleavers*, which perform as well as random interleavers. Regular coprime interleavers are attractive for their random-like behavior and cheap implementation. For $N = 2^m$, the following parameters lead to regular coprime interleavers:

$$\begin{cases} a = 8k-3, & k = 1, 2, ..., N/8 \\ b = 2t-1, & t = 1, 2, ..., N/2 \end{cases} \quad (10)$$

5 Graph Representation and Simulations

As a complement to the CCS evaluation, we visualize the randomness of some interleavers using graphs. As shown in Figure 3, a length-N interleaver can be represented using an $N \times N$ grid or lattice where the y-axis represents the original sequence i and the x-axis indicates the interleaved sequence $\pi(i)$.

The coprime interleavers with $N = 100$, $a = 1$, $b = 63$ (top-left) and $N = 128$, $a = 1$, $b = 79$ (top-right) are the Golden prime interleavers. Despite their regularity which may lead to repeated and periodic error patterns, Golden prime interleavers offer quite good performances especially at short lengths.

The coprime interleaver with $N = 128$, $a = 33$, $b = 79$ (mid-left) is an example of a poor interleaver. The undesirable interleaving pattern is obvious from the existence of many repeated (error) patterns and in particular the many vulnerable pairs with very short Euclidean distances [1].

The three other interleavers, the regular coprime interleaver with parameters $N = 100$, $a = 5$, $b = 79$ (mid-right), the Welch-Costas interleaver with $N = 100$, $a_1 = 11$ (bottom-left), and the Takeshita-Costello interleaver with $N = 128$, $a_2 = 41$ (bottom-right), are clearly examples of algebraic interleavers that are constructed using structure yet exhibit random-like behavior.

Further, it is interesting to compare the three interleavers on the top-right, mid-left and mid-right, all of which have $b = 79$, the Golden section. Depending on a, they exhibit very different properties: regular but still good, regular and bad, and random-like and hence good. This points out the importance to understand the classification of coprime interleavers and the impact of the parameters on their behavior, and to subsequently make informed choices.

Finally, we provide the SNR-vs-BER performance of regular coprime interleavers, and compare it with that of the Takeshita-Costello interleavers and random interleavers. Two different interleaver lengths of 128 bits and 2048 bits are simulated for a turbo code with two identical component codes of generator polynomial $[1, 5/7]$. The simulation results confirm that regular coprime interleavers perform as well as random interleavers and the Takeshita-Costello interleavers.

6 Conclusion

Algebraic interleavers are preferable due to practical concerns such as reduction of hardware requirements and interleaving/deinterleaving operations. In this paper, we investigate the behavior of random interleavers and random-like algebraic interleavers using the metric of cycle correlation sum. We found that random interleavers and S-random interleavers fall into the fixed regions in the CCS plane. The Welch-Costas interleavers, the Takeshita-Costello interleavers and certain subclasses of coprime interleavers will also stay in the random interleaver region. Following this observation, we propose a bank of good interleavers, termed *regular coprime interleavers*, and formulate their parameters for interleaver lengths of power of 2. Graph representation and BER simulations further confirm the randomness and the good performance exhibited by regular coprime interleavers. In addition, we found that the subclass of linear coprime interleavers ($a = 1$), although simple, contain some of the best interleavers. However, caution should be taken in choosing parameter b, since the same subclass also contain some of the worst interleavers.

We conclude the paper by proposing the regular coprime interleavers as a strong candidate for practical turbo codes. They offer similar performance as the Welch-Costas interleavers, the Takeshita-Costello interleavers, and random interleavers, but are simpler, more storage efficient and easily parallelizable.

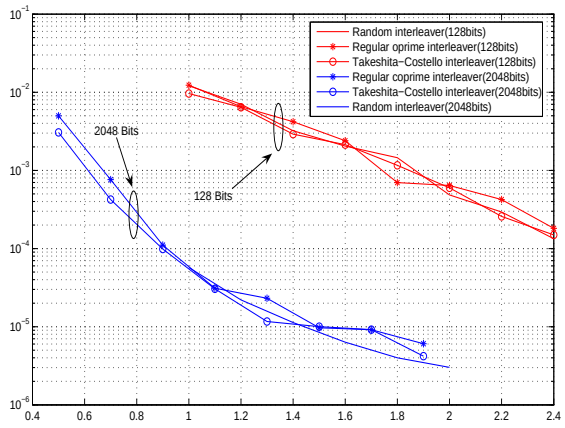

Fig. 4. BER performance of the random-like interleaver.

References

[1] K. Xie, P. Tan, J. Li and W. Wang, *Interleaver Design for Short-Length Turbo Codes*. Proc. of 39th Conference on Information Sciences and Systems, March, 2005.

[2] J. Li, *Low-complexity, capacity-approaching coding schemes: design, analysis and applications*. Dissertation, Dec 2002, pp:58 .

[3] C. Heegard, and S. B. Wicker., *Turbo Coding*. Boston: Kluwer Academic Publishers, 1999. pp:54-55.

[4] O. Y. Takeshita, and D. J. Costello, D.J., Jr., *New deterministic interleaver designs for turbo codes* IEEE Trans. Inf. Theory, vol.46, no.6, pp:1988-2006, Sept. 2000

[5] S.Dolinar,and D. Divsalar, *Weight distribution of turbo codes using random and nonrandom permutations*. JPL, TDA Progress. Rep. 42-122, Aug. 1995.

[6] S. Crozier, J. Lodge, P. Guinand and A. Hunt, *Performances of Turbo-Codes with Relative Prime and Golden Interleaving Strategies*, 6th Inter. M. Satellite Conference, pp. 268-175, Jun 1999.

[7] J. Hokfelt, *On the Design of Turbo Codes*, PhD dissertation, Lund University, 2000.