Proceedings of the National Conference
On Undergraduate Research (NCUR) 2007
Dominican University of California
San Rafael, California
April 12-14, 2007

Permutations of Z_{p^r} as Interleavers for Turbo Codes

Joyce M. Fernandez
Mathematics Department
University of Puerto Rico at Humacao
Humacao, Puerto Rico

Faculty Advisor: Dr Ivelisse Rubio

Abstract

Interleavers for error correcting codes are permutations of Z_n. Permutations of Z_{p^r} constructed from permutations of finite fields F_{p^r} using monomials x^i and the performance of Turbo Codes using them as interleavers have been studied by C. Corrada and I. Rubio[1]. We construct permutations of Z_{p^r} from permutations of finite fields F_{p^r} that decompose in cycles of length 2 using monomials cx^i. We study the dispersion and the spreading of these permutations and the performance of Turbo Codes using them as interleavers.

Keywords: permutation, interleaver, turbo code.

1. Introduction

Error control codes are used in digital communication systems to correct errors that might occur during the transmission of messages. Some examples of systems that use error correcting codes are satellite communication, cellular phones, storage of information in compact discs (CD), computer memory and others. Figure 1 shows a message passing thru a channel that could have noise. The received message could have errors. On Figure 2, the message passes thru an encoder, the encoder adds redundancy to the message and we obtain a codeword. At the receiver, the decoder detects and corrects the errors.

On the compact disc example the channel is the disc and the noise can be dirt. The information on the CD is encoded, so that when the CD is played, the player decodes to detect and correct the errors. On the cellular phone example the digital signal is transmitted over the air and an antenna receives it. But while the signal is traveling, interruptions could occur, for example, if we are near to a mountainous area. These are some examples of why error correcting codes are necessary in digital communication systems.

$$\underbrace{(m_1, m_2, ..., m_k)}_{message} \xrightarrow{channel} \underbrace{(m_1+e_1, m_2+e_2, ..., m_k+e_k)}_{received\ message}$$

Figure 1. Message transmission without codifying.

$$\underbrace{(m_1, m_2, \ldots, m_k)}_{message} \xrightarrow{encoder} \underbrace{(c_1, c_2, \ldots, c_n)}_{codeword} \xrightarrow{channel} \underbrace{(c_1 + e_1, \ldots, c_n + e_n)}_{received\ message} \xrightarrow{decoder} \underbrace{(\hat{m}_1, \hat{m}_2, \ldots, \hat{m}_k)}_{message\,?}$$

Figure 2. Digital communication.

Turbo Codes are appropriated for wireless communication systems because they have an effective performance in the correction of errors and provide a reduction to the transmitter power levels. The interleaver is an important component of Turbo Codes and its function is to permute the information symbols. One of its advantages is that consecutive information symbols might not be affected if consecutive errors occur during the message transmission. We study some properties like the dispersion and the spreading of the permutations and the performance of Turbo Codes using them as interleavers. We also have interest in permutations that decompose in cycles of length 2 because these permutations are their own inverse and this has an implementation advantage because the same technology that is constructed to encode the information, could be used to decode it.

The interleavers that we study are permutations of Z_{p^r} constructed with monomials cx^i over F_{p^r} and the performance of Turbo Codes using them as interleavers. This would generalize our study[4] of permutations of Z_p obtained with monomials cx^i.

2. Permutations and Interleavers

A permutation is a reordering of the elements of a set. This is, a permutation of a set A is a bijective function $\pi : A \to A$. An interleaver is an important component of some codes that permutes the information symbols. This means that an interleaver is a permutation. The following figure shows the encoding process of a Turbo Code. The codeword c is the concatenation of the original message m with the encoded message x_1 and with the message permuted by the interleaver I and then encoded with encoder $\mathrm{e}_2 : \mathrm{c} = (\mathrm{m}, \mathrm{x}_1, \mathrm{x}_2)$.

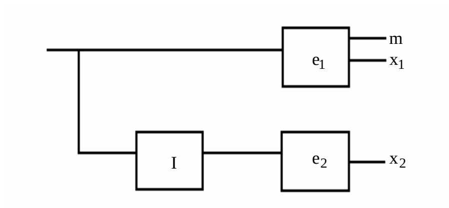

Figure 3. Turbo Code.

Choosing interleavers randomly is one way to construct them. Turbo Codes with interleavers constructed in this way have good performance, but the performance has to be analyzed by simulations and the permutations have to be stored in memory. Another method to construct interleavers is algebraically. Interleavers constructed in this way have the advantage that they could be analyzed in advance and can be generated in the moment. In this way memory space is saved and good constructions could be characterized. We want interleavers that result in codes with good performance. Some properties that have been associated to the performance of the codes are the dispersion and the spreading.

2.1. permutation monomials

Now we introduce the function that we use to construct permutations of F_q and hence permutations of Z_q.

Definition 1. A monomial $x^i \in F_{p^r}[x]$ is a ***permutation monomial*** if and only if the polynomial function $f : F_q \to F_q$; $f(x) = x^i$ is a permutation of the finite field F_q.

Example 2. The function $\pi(x): F_7 \to F_7$, $\pi(x) = x^5$ is a permutation monomial of F_7 and it can be represented as

$$\pi = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ 0 & 1 & 4 & 5 & 2 & 3 & 6 \end{pmatrix}$$

The cyclic decomposition is (2,4) (3,5).

On this representation, the elements of the first row are the elements of F_7 and the elements of the second row are their images under π. The following is a well known characterization of permutation monomial.

Theorem 1. The monomial $x^i \in F_q[x]$ is a permutation monomial of F_q if and only if $\gcd(i, q-1) = 1$.

The next theorem tells us that we can use any permutation of F_q to obtain a permutation of Z_q it was proved in our previous work[2].

Theorem 2. Let $q = p^r$, p a prime, $\{\xi_0, \xi_1, \ldots \xi_{q-1}\} = F_q$ and $f : F_q \to F_q$ any function. The function $\pi : Z_q \to Z_q$ defined as $\pi(n) = m$, where $f(\xi_n) = \xi_m$, is a permutation of Z_q if and only if f is a permutation of F_q.

When we write $\xi_0, \xi_1, \ldots, \xi_{q-1}$ we are assuming that we ordered the elements of the field. This associates the elements of the field with integers of 0 to q-1. After applying the permutation to the elements of F_q, the π function "follows" these elements and we obtain a permutation of Z_q.

L. Cruz[3] characterized the monomials ax^i such that the permutations of F_q given by them decomposed in cycles of length 2. These results together with Theorem 2 guaranty that we can obtain permutations $\pi : Z_q \to Z_q$ that decompose in cycles of length 2 from permutations $f : F_q \to F_q$ that decompose in cycles of length 2 using certain $f(x) = ax^i$.

2.2. interleaver properties

The dispersion and the spreading are properties of a permutation that have been associated to the performance of Turbo Codes. We want to study these properties for permutations of F_q given by monomials cx^i.

Let π be a permutation of Z_n. The dispersion is a factor that measures the interleaver randomness. The dispersion is given by the number of elements in the set $D(\pi) = \{j - i, \pi(j) - \pi(i)) \mid 0 \le i < j < n\}$. To be able to compare permutations of different lengths, we calculate the normalized dispersion $\gamma = \dfrac{2 \mid D(\pi)}{n(n-1)}$. The closer the normalized dispersion is to 1 the best it is.

The spreading measures how separate are the elements that originally were near. An interleaver has spreading factor s, if s is the largest integer such that $\mid i - j \mid \le s \Rightarrow \mid \pi(i) - \pi(j) \mid \ge s$. The closer the spreading is to $\sqrt{\dfrac{n}{2}}$, the best it is.

3. Permutations of Z_{p^r} obtained from permutations of F_{p^r}

We need to construct permutations of Z_n. Our constructions use monomials $cx^i \in F_{p^r}[x]$ that give permutations of F_{p^r}. Theorem 2 guaranties that we can do this. We construct permutations of Z_{p^r} in the following manner: first we associate the elements of F_{p^r}, to *r*-tuples of integers between 0 and p-1. Then we order the *r*-tuples using vector orderings. In this way we create a correspondence between the elements of F_{p^r} and the integers between 0 and $p^r - 1$, the elements of Z_{p^r}. Finally we apply the monomial cx^i to the elements of F_{p^r} and taking the indices *n* from the ξ_n we obtain a permutation of Z_{p^r}. We will illustrate this process with an example.

3.1. representations of finite fields

There are several ways in which one can represent the elements of a finite field. One of them is to write the non-zero elements as powers of a primitive root.

Definition 2. *Let* $\alpha \in F_q$, α *is a* ***primitive root*** *of* F_q *if and only if* α *generates all the elements of* $F_q^* = F_q \setminus \{0\}$. *This is,* $F_{p^r} = \{\alpha^0, \alpha^1, ..., \alpha^{q-2}\}$.

Definition 3. *A polynomial* $f \in F_q[x]$ *of degree* $m \geq 1$ *is called a* ***primitive polynomial*** *over* F_q *if it is the minimal polynomial over* F_q *of a primitive element of* F_{q^m}.

It is known that a finite field F_{p^r} is isomorphic to a quotient $Z_p[x] / \langle p(x) \rangle$ where *p(x)*, is an irreducible primitive polynomial over $Z_p[x]$ that has degree *r*. The elements in this quotient can be identified with polynomials with coefficients in Z_p and degree less than r. These polynomials can be associated to vectors of length r and entries in Z_p. This gives a representation of F_{p^r} as a vector space over Z_p. If the polynomial *p(x)* is primitive, this will give us a correspondence between the primitive root representation and the polynomial and the *r*-tuple representation. The following example illustrates how to get the different representations.

Example 3. Consider $F_{3^2} \approx Z_3[x]/\langle x^2 + x + 2 \rangle$ and let α be a primitive root of F_{3^2}.

The following table shows the different representations of the finite field $F_{3^2} \approx Z_3[x]/\langle x^2 + x + 2 \rangle$. In the first column is the primitive root representation, in the second is the polynomial representation that is obtained in the following way: The primitive root α of F_{3^2} is a zero of the primitive polynomial $x^2 + x + 2$. Therefore, $\alpha^2 + \alpha + 2 = 0$ and from here we can obtain that $\alpha^2 = -\alpha - 2 = 2\alpha + 1$ because the coefficients are in Z_3. In this way we can reduce all the powers of α that are greater or equal to 2 and write them as polynomials of degree less than 2. Finally, in the third column is the *r*-tuple representation that it is obtained taking the coefficients of the polynomials.

Table 1. representations of the finite field F_{3^2}

α^i	polynomial	r-tuple
α^0	1	(0,1)
α^1	α	(1,0)
α^2	$2\alpha+1$	(2,1)
α^3	$2\alpha+2$	(2,2)
α^4	2	(0,2)
α^5	2α	(2,0)
α^6	$\alpha+2$	(1,2)
α^7	$\alpha+1$	(1,1)
α^8	1	(0,1)

3.2. vector orderings

We can order a field using their vector representation and vector orderings. For this, it is necessary to decide how to order the vectors so that we always can decide when an element is greater than another one. Some examples of vector orderings are the Lexicographic Order and the Graded Lexicographic Order.

3.2.1. *lexicographic order*

The first type of vector orderings that we will utilize is the Lexicographic Ordering. Intuitively, ordering the vectors in this way is similar to the method used to order the words in a dictionary.

Definition 3. Let $\gamma=(\gamma_1,\gamma_2,...,\gamma_r)$ and $\delta=(\delta_1,\delta_2,...,\delta_r)$. We have $\gamma >_{lex} \delta$ if and only if the left-most nonzero entry of $\gamma-\delta$ is positive.

Example 4. $\gamma=(1,0,0) >_{lex} (0,1,1)=\delta$ since $\gamma-\delta=(1,-1,-1)$.

3.2.2. *graded lexicographic order*

The second type of vector orderings that we will utilize is the Graded Lexicographic Order. This order takes in consideration the total degree of the vectors.

Definition 4. Let $\gamma=(\gamma_1,\gamma_2,...,\gamma_r)$ and $\delta=(\delta_1,\delta_2,...,\delta_r)$, be such that $|\gamma|=\sum_{i=1}^{r}\gamma_i$, $|\delta|=\sum_{i=1}^{r}\delta_i$. We have $\gamma >_{grlex} \delta$ if and only if $|\gamma|>|\delta|$ or , $|\gamma|=|\delta|$ and $\gamma >_{lex} \delta$.

Example 5. Let $\gamma=(1,0,0)$ and $\delta=(0,1,1)$. Then $\delta >_{grlex} \gamma$. Note that $\gamma >_{lex} \delta$.

Continuing with Example 3, we will now order the elements of F_{3^2} using the Lexicographic Order.

Table 2. orderings for F_{3^2}

Natural number	r –tuple Lexicographic Order	r -tuple Graded Lexicographic Order
0	(0,0)	(0,0)
1	(0,1)	(0,1)
2	(0,2)	(1,0)
3	(1,0)	(0,2)
4	(1,1)	(1,1)
5	(1,2)	(2,0)
6	(2,0)	(1,2)
7	(2,1)	(2,1)
8	(2,2)	(2,2)

3.3 permutations of Z_{p^r}

C. Corrada and I. Rubio[1] studied permutations of Z_p given by monomials x^i. They obtained bounds for the dispersion and good simulation results for Turbo Codes with interleavers for Z_p constructed with x^{p-2}. Y. Luis and L. Pérez[2] studied permutations of Z_{p^r} constructed from permutations of F_{p^r} that were given by monomials x^i. We studied[4] the dispersion and spreading of permutations of Z_p given by cx^{p-2}. Here we study permutations of $Z_q, q = p^r$ given by monomials cx^{q-2}. We are interested in permutations that decompose in cycles of length 2. These types of permutations were characterized by L. Cruz and I. Rubio[3].

One obtains permutations of Z_{p^r}, by ordering the elements of the finite field F_{p^r} and then associating them to Z_{p^r}. We construct a permutation of Z_{p^r} using the permutation monomial $f(x) = cx^i$, and order the elements of F_{p^r} using their vector representation and vector orderings. In this way we create a correspondence between the elements of F_{p^r} and the elements of Z_{p^r}. Finally we apply the monomial cx^i to the elements of F_{p^r}, follow their position in the ordering, and obtain a permutation of Z_{p^r}. We now conclude our example of the construction of a permutation of Z_{p^r} from a permutation of F_{p^r}.

Example 6. Let $F_{3^2} = Z_3[x]/\langle x^2 + x + 2\rangle$. We can construct a permutation of Z_{3^2} using the permutation of F_{3^2} obtained with the monomial $\pi(x) = 2x^7$. We order the elements of F_{3^2} with the Lexicographic Order. Then we evaluate each element α^i in $2x^7 \in F_{3^2}[x]$ to obtain a permutation of F_{3^2} and assign to the result the correspondent natural number, from the results obtained by L. Cruz[3] we know that this is a permutation of F_{3^2} that decompose in cycles of length 2. We "follow" the position in the ordering and this gives us the permutation.

$$\begin{array}{ccccccccccc}
Z_{p^r} & & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\
 & & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow \\
 & \text{Lexicographic Order} & (0,0) & (0,1) & (0,2) & (1,0) & (1,1) & (1,2) & (2,0) & (2,1) & (2,2) \\
 & & || & || & || & || & || & || & || & || & || \\
F_{p^r} & \text{Primitive Root} & 0 & \alpha^0 & \alpha^4 & \alpha^1 & \alpha^7 & \alpha^6 & \alpha^5 & \alpha^2 & \alpha^3 \\
\pi(x)=2x^7 & & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\
F_{p^r} & \text{Primitive Root} & 0 & \alpha^4 & \alpha^0 & \alpha^3 & \alpha^5 & \alpha^6 & \alpha^7 & \alpha^2 & \alpha^1 \\
\updownarrow & & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow & \updownarrow \\
Z_{p^r} & & 0 & 2 & 1 & 8 & 6 & 5 & 4 & 7 & 3
\end{array}$$

The permutation is $\begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 0 & 2 & 1 & 8 & 6 & 5 & 4 & 7 & 3 \end{pmatrix}$.

The cyclic decomposition is (1,2) (3,8)(4,6).

4. Performance of Turbo Codes

We wrote programs in Maple to construct permutations monomials and compute their spreading and dispersion. Jose Lugo, from the UPR-High Performance Computing Facilities ran simulations of Turbo Codes that use these permutations as interleavers. Our goal was to compare the performance of the Turbo Codes with the dispersion and the spreading to see if there is any relation. As an illustration, the following table shows the dispersion and the spreading results for permutations of F_q given by cx^{q-2} where q=625. The columns are divided in Graded Lexicographic Ordering and Lexicographic Ordering; each one contains the exponent l of the primitive root α , the dispersion and the spreading respectively. Note that in this table all the permutations computed have dispersion close to .8; this is true for all the permutations cx^{q-2} .

Table 3. dispersion and spreading of permutations of F_{625}

q=625 $Z_5 / < x^4 + x^3 + x + 3) >$					
Graded Lexicographic Ordering			Lexicographic Ordering		
l	dispersion	spreading	l	dispersion	spreading
0	.813246	1	0	.806277	1
574	.817277	1	303	.817538	1
534	.817031	1	300	.817538	2
13	.812872	1	124	.812313	1
80	.812867	1	102	.812308	1
528	.807062	1	409	.802462	1
212	.814549	3	372	.814323	3
325	.810892	3	193	.811995	3

As an illustration, the following graph shows the performance of a Turbo Code constructed with interleavers given by the permutations described above for the Graded Lexicographic Order and are compare with random and s-random interleavers. The random interleavers are the ones that are used actually in the application. The s-random interleavers are known to be the best in terms of performance. The SNR means the signal to noise ratio and the BER measures the probability of errors in the message. The interleaver that more down is in the graph has the best performance. Note that, for example, the interleavers a_528, a_13 and a_0 have better performance than the random interleaver.

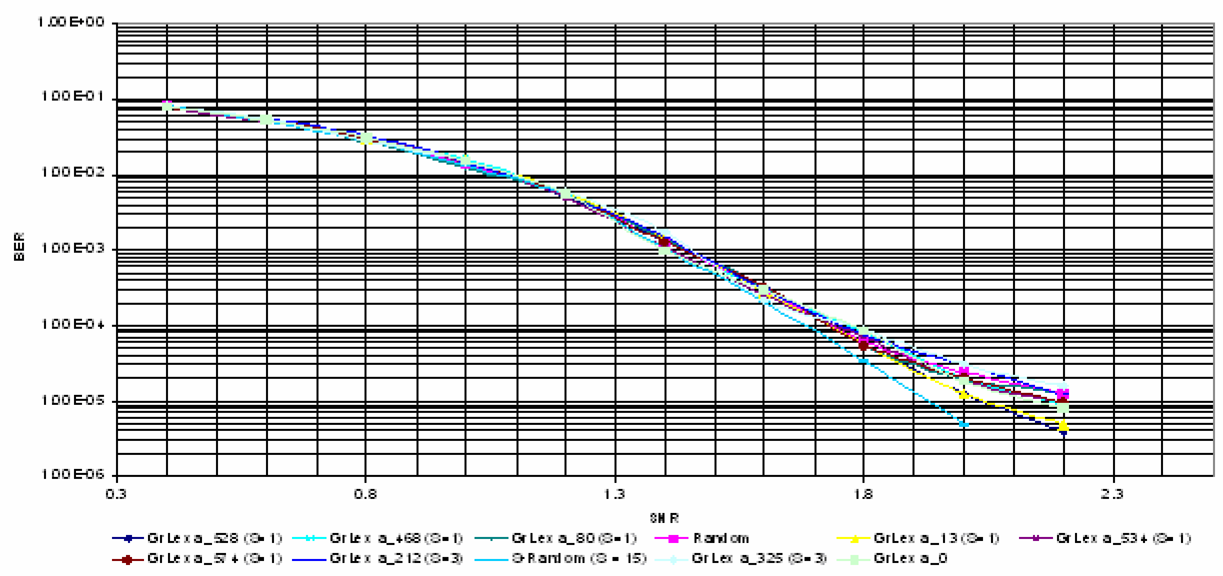

Figure 4. BER for random, s-random and graded lexicographic interleavers of length 625.

6. Conclusions and Work in Progress

The results presented here are partial results. We computed all the permutations $\alpha^{l} x^{q-2}$ and obtained dispersion close to .8. All but two of the permutations that were tried perform better than the random permutations, which is the one that is used in actual applications. It seems that small differences in spreading do not affect the performance of the code. It still has to be investigated from which value on, the spreading does makes a difference in performance. It seems that there is not too much difference in performance between finite fields ordered with Lexicographic Order and Graded Lexicographic Order. We still want to determinate other parameters for the interleavers that could be related to the performance of the code. We also want to study other permutations with cycles of length 2 to see if we could characterize permutations with good performance.

7. Acknowledgements

This work has been funded in part by the National Security Agency, Grant Num. H98230-04-C-0486; and by the National Science Foundation CSEMS program at the UPRH, Grant Num. 0123169. Simulations were done at the UPR-High Performance Computing Facilities.

8. References

Journals

1. C. Corrada, I. Rubio, “Algebraic Construction of Interleavers Using Permutation Monomials”, IEEE International Conference on Communications, 2004.
2. Y. Luis, L. Pérez, “Properties of a Class of Permutations Over Finite Fields and Applications on Turbo Codes”, Proceedings of The National Conference On Undergraduate Research (NCUR) 2005.
3. L. Cruz, “Characterization of Monomials over Finite Fields that Give Permutations that Decompose in Cycles of Length 2”, Proceedings of The National Conference On Undergraduate Research (NCUR) 2007.
4. J. Fernández, “Properties of Permutations with Cycles of Length 2 and Obtained with Monomials”, Proceedings of The National Conference On Undergraduate Research (NCUR) 2006, April 2006.