A Computer Search Algorithm for Optimum Free Distance Binary Convolutional Codes*

Herbert S. Palines, Virgilio P. Sison

Institute of Mathematical Sciences and Physics
University of the Philippines Los Baños
College, Laguna 4031, Philippines

{hspalines, vpsison}@uplb.edu.ph

ABSTRACT

A computer search algorithm for optimum free distance (OFD) binary convolutional codes is presented. The algorithm obtains polynomial encoders for rate-1/2 and rate-2/4 OFD binary convolutional codes. For rate-1/2 OFD binary convolutional codes, the search is done for memory 1, 2, 3, 4 and 5 and having Hamming free distances d_{free} equal to $4, 5, 6, 7$ and 8, respectively. Minimal-basic encoders are also derived. The encoders for rate-2/4 OFD binary convolutional codes have memory 2 and $d_{free} = 8$ and all are minimal-basic. The resulting codes meet the Griesmer's upper bound on the free distance.

Keywords

Convolutional codes, column distance, row distance, free distance, distance profile, Griesmer bound

1. INTRODUCTION

The existing digital communication systems such as digital imaging, radio and mobile communications, and deep space telecommunications, utilize error correcting block codes and convolutional codes to achieve efficient and reliable data transmission. Most of the time, these codes have optimum distances. For convolutional codes, it is well known that the free distance is the main parameter for the decoding error especially when Viterbi decoding [13] and sequential decoding [2] are used. In constructing optimum free distance (OFD) or optimum distance property (ODP) convolutional codes, a computer search is usually employed (see for example [3], [4], [6], [7], [10], [1], [9], and [8]).

The main objective in a computer search is to find for relatively superior convolutional codes of various rates based on the specified criteria, usually, the free distance and the distance profile of the code. In searching for OFD convolutional codes, the goal is to find for codes with maximum free distance among the codes with the same rate and memory. However, for ODP codes, the distance profile is the major concern of the search. It has been reported in [4] that an exhaustive search becomes practically impossible even for small constraint lengths. Therefore, methods are needed to limit the search for good codes. For instance in [4], rate-2/3 and 3/4 OFD minimal codes were obtained by utilizing the critical terminated code, the parity check matrix of the code and the minimality of the code. In [10], the search for rate-1/2 binary convolutional codes was simplified by considering a smaller ensemble of systematic ODP encoders. The result in [1] uses a fast algorithm in searching a tree (FAST) to compute for the distance spectrum parameters and to find for ODP or OFD rate-1/2 binary convolutional codes. A different strategy was used in [9] where an OFD rate-1/2 convolutional code over the ring of integers modulo 4 where constructed initially then Gray mapped it to a binary trellis code which was found to be an OFD code. In the same paper, OFD rate-2/4 binary convolutional codes were also found.

It must be noted that the proposed algorithm in this study is different from the papers mentioned above. Mainly because we focus only on searching OFD convolutional codes. Moreover, the proposed search exclusively uses the most common distance measures for convolutional codes, the column distance and row distance introduced by Costello [3]. In this manner, this algorithm has a potential to be more algebraic in nature.

The material is organized as follows. Section 2 introduces the definition of binary linear block codes and binary convolutional codes and its distance measures. The reader is referred to [11] for a more detailed discussion. Section 3 presents the algorithm used to construct the OFD rate-1/2 and rate-2/4 binary convolutional codes. Examples of OFD rate-1/2 and rate-2/4 binary convolutional codes are given in Section 4. The computer program is created using the computer algebra MAGMA® on a personal computer (*Acer Veriton 7200, Intel® 1.6GHz, SDRAM 2GB*). In Section 5, the summary and some remarks are stated.

2. PRELIMINARIES AND DEFINITIONS

*This study is supported by the University of the Philippines Los Baños through the UPLB Basic Research Grant.

A *rate-k/n binary linear block code* $\mathcal{B}$ of length n is a k-dimensional subspace of $\mathbb{F}_2^n$, where $\mathbb{F}_2$ is the Galois field of order 2. The code $\mathcal{B}$ is completely determined by a full rank $k \times n$ matrix G also known as the generator matrix or encoder of $\mathcal{B}$. The rows of G constitute a set of linearly independent vectors in $\mathbb{F}_2^n$ that span $\mathcal{B}$. Consequently, the encoding map η given by

$$\begin{aligned} \eta : \mathbb{F}_2^k &\to \mathbb{F}_2^n \\ u &\mapsto v = uG \end{aligned}$$

is injective, where u is an *information word* encoded by G as the *codeword* v. The code $\mathcal{B}$ is said to be systematic if it is encoded by a systematic encoder $G = \begin{pmatrix} I_k & A \end{pmatrix}$ where I_k is the $k \times k$ identity matrix and A is a $k \times (n-k)$ matrix over $\mathbb{F}_2$.

A binary linear code can be equipped by the Hamming weight function w_H to determine its minimum distance. It is known that the minimum distance of a code is the main parameter that dictates its error detecting and error correcting capability. The Hamming weight of a codeword $v \in \mathcal{B}$, denoted by $w_H(v)$, is the number of nonzero symbols in v. The distance between any two codewords $v', v'' \in \mathcal{B}$ is computed as $w_H(v'-v'')$. Equivalently, it is the number of coordinates in which v' and v'' differ. The minimum Hamming distance d of the code $\mathcal{B}$ is the smallest possible nonzero distance between any two codewords in $\mathcal{B}$. Since $\mathcal{B}$ is linear, d is also given by

$$d = \min_{v \in B, v \neq 0} \{w_H(v)\}$$

.

A *rate-k/n binary convolutional encoder* can be regarded as a linear mapping whose inputs (*information sequence*) are of the form

$$u = \cdots u_{-2}u_{-1}u_0u_1u_2 \cdots,$$

where each block u_j has k symbols, that is,

$$u_j = \begin{pmatrix} u_j^{(1)} & u_j^{(2)} & \cdots & u_j^{(k)} \end{pmatrix},$$

and $u_j^{(i)} \in \mathbb{F}_2$, for $i = 1, 2, \cdots, k$. Each k-ary information block u_j is accepted by the encoder at a given time instant j and an n-ary code block v_j is produced. Each code block v_j is given by

$$v_j = \begin{pmatrix} v_j^{(1)} & v_j^{(2)} & \cdots & v_j^{(n)} \end{pmatrix},$$

and $v_j^{(i)} \in \mathbb{F}_2$, for $i = 1, 2, \cdots, n$. Thus, the corresponding output, called as the *code sequence* or simply *codeword*, for an information sequence u is given by

$$v = \cdots v_{-2}v_{-1}v_0v_1v_2 \cdots .$$

The sequences u and v must start at some finite time, usually at $j = 0$, and may or may not end.

The main difference of a convolutional encoder from a block encoder is that the j-th code block v_j depends not only on the current information block u_j, but also on the earlier, say m, fixed number of information blocks $u_{j-1}, u_{j-2}, \cdots, u_{j-m}$. It is often convenient to express v_j as

$$v_j = u_jG_0 + u_{j-1}G_1 + \cdots + u_{j-m}G_m,$$

where G_i(s) are $k \times n$ binary *generator submatrices*. Moreover, we can derive the code sequence v via

$$\cdots v_{-2}v_{-1}v_0v_1v_2 \cdots = (\cdots u_{-2}u_{-1}u_0u_1u_2 \cdots)\, G$$

or

$$v = uG$$

where

$$G = \begin{pmatrix} G_0 & G_1 & \dots & G_m & & \\ & G_0 & G_1 & \dots & G_m & \\ & & \ddots & \ddots & & \ddots \end{pmatrix} \tag{1}$$

is a semi-infinite matrix and the blank spaces in G are assumed to be filled with zeros.

In this paper, we consider a feedback-free convolutional encoder. That is, a causal information sequence, .i.e. $u_j = 0$ for all $j < 0$, given by

$$u = u_0u_1u_2 \cdots,$$

is encoded as a causal code sequence

$$v = v_0v_1v_2 \cdots .$$

Another convenient way of writing these sequences is through the delay operator D or D-transforms:

$$u(D) = u_0 + u_1D + u_2D^2 \cdots$$

and

$$v(D) = v_0 + v_1D + v_2D^2 \cdots .$$

Note that $u(D)$ and $v(D)$ belong to $\mathbb{F}_2[[D]]$, the ring of formal power series over $\mathbb{F}_2$. Moreover, if we limit the given sequences to be causal and finite, $u(D)$ and $v(D)$ become polynomials of some degree. Clearly, the ring of polynomials $\mathbb{F}_2[D]$ is contained in $\mathbb{F}_2[[D]]$. The polynomial generator matrix (encoder) $G(D)$ for the code is given by

$$G(D) = G_0 + G_1D + \cdots + G_mD^m.$$

In general, $u(D)$ and $v(D)$ are elements of the field of Laurent series $\mathbb{F}_2((D))$. In [5], a rate-k/n binary convolutional code $\mathcal{C}$ is a k-dimensional $\mathbb{F}_2((D))$-subspace of $\mathbb{F}_2((D))^n$. In this paper, we consider a *rate-k/n binary convolutional code* as a $\mathbb{F}_2(D)$-subspace of $\mathbb{F}_2(D)^n$ with dimension k, where $\mathbb{F}_2(D)$ is the field of rational functions over $\mathbb{F}_2$. Thus, the encoder $G(D)$ is a matrix with entries coming from $\mathbb{F}_2(D)$. Now, the encoding can be summarized as

$$v(D) = u(D)G(D)$$

where $G(D)$ is a matrix over $\mathbb{F}_2(D)$. Henceforth, it should be understood that a codeword $v(D)$ is also treated as a code sequence.

We now consider some structural properties of an encoder $G(D)$ of a rate-k/n binary convolutional code. We say that $G(D)$ is *basic* if it is polynomial and has a polynomial right inverse. Equivalently, $G(D)$ is basic if all of its $k \times k$ minors are relatively prime [12]. The *i-th constraint length* of a polynomial encoder $G(D)$, denoted by ν_i, is defined to be

the maximum among the degrees of the component polynomials of the i-th row of $G(D)$. The *overall constraint length* of $G(D)$ is given by $\nu = \sum_{i=1}^{k} \nu_i$. An encoder $G(D)$ is *minimal-basic* if it is basic and the overall constraint length ν is minimal over all equivalent basic encoders. A binary convolutional code is *systematic* if it is encoded by a systematic encoder $G(D) = \begin{pmatrix} I_k & A(D) \end{pmatrix}$ where I_k is the $k \times k$ identity matrix and $A(D)$ is a $k \times (n-k)$ matrix over $\mathbb{F}_2(D)$. Two encoders are *equivalent* if they generate the same code.

Before we discuss distance measures of a convolutional code, we introduce first the Hamming weight of a causal and finite code sequence. The Hamming weight of the sequence $v = v_0 v_1 \dots v_j$, denoted by $w_H(v)$, is the number of nonzero symbols in v. Similarly, if $v(D) = \begin{pmatrix} v_1(D) & v_2(D) & \cdots & v_n(D) \end{pmatrix}$ is a polynomial codeword, the Hamming weight of $v(D)$ is the sum of the numbers of nonzero coefficients of polynomials $v_i(D)$, for $i = 1, 2, \dots, n$. Because of the linearity of convolutional codes, the free distance d_{free} of a code $\mathcal{C}$ is given by

$$d_{free} = \min_{v(D) \in C, v(D) \neq 0} \{w_H(v(D))\}.$$

An important upper bound on the *free distance* of a rate-k/n binary convolutional code of memory m is the Griesmer's bound which states that the inequality

$$\sum_{j=0}^{ki-1} \left\lceil \frac{d_{free}}{2^j} \right\rceil \leq (m+i)n$$

is satisfied for $i = 1, 2, \cdots$. Table 1 gives the upper bounds on free distances of rate-1/2 and rate-2/4 binary convolutional codes of memory $m = 0$ up to 14.

Table 1: Griesmer's bounds for rate-$1/2$ and rate-$2/4$ binary convolutional codes.

m	0	1	2	3	4	5	6	7
rate-1/2	2	4	5	6	7	8	10	10
rate-2/4	2	5	8	10	12	14	16	18
m	8	9	10	11	12	13	14	
rate-1/2	12	12	14	15	16	16	18	

We are now ready to discuss the fundamental distance measures of convolutional codes. The main reference of the following discussion is [11]. Let $\mathcal{C}$ be a rate-k/n binary convolutional code with a polynomial encoder $G(D)$ of memory m. Moreover, we let $u(D)$ and $v(D)$ be the polynomial information word and the encoded codeword, respectively. Note that we can express $u(D)$ and $v(D)$ as $u = u_0 u_1 u_2 \dots u_{t'}$ and $v = v_0 v_1 v_2 \dots v_{t''}$, respectively, for some positive integers t' and t''. We further consider the following notation for a finite and causal sequence x given by

$$x_{[0,t]} = x_0 x_1 x_2 \dots x_t,$$

for some positive integer t.

The *j-th order column distance* d_j^c of the generator matrix $G(D)$ is the minimum Hamming distance between two encoded sequences $v_{[0,j]}$ resulting from the causal information sequences $u_{[0,j]}$ with differing u_0. Since $\mathcal{C}$ is linear, it follows that d_j^c is also the minimum of the Hamming weights of the paths $v_{[0,j]}$ resulted from causal information sequences with $u_0 \neq 0$. Thus, we have

$$d_j^c = \min_{u_0 \neq 0} \{w_H\left(v_{[0,j]}\right)\}. \tag{2}$$

Consider a polynomial encoder $G(D)$ with a semi-infinite matrix G given in (1), we truncate G after $j+1$ columns and we denote the resulting matrix by G_j^c, that is

$$G_j^c = \begin{pmatrix} G_0 & G_1 & G_2 & \dots & G_j \\ & G_0 & G_1 & & G_{j-1} \\ & & G_0 & & G_{j-2} \\ & & & \ddots & \vdots \\ & & & & G_0 \end{pmatrix} \tag{3}$$

where $G_i = \mathbf{0}$ when $i > m$. Consequently, (2) can be rewritten as

$$d_j^c = \min_{u_0 \neq 0} \{w_H\left(u_{[0,j]} G_j^c\right)\}.$$

The column distance is an encoder property and not a code property. To this effect, we define an encoding matrix $G(D)$ such that $G(0)$ is of full rank. By doing this, we can say that the column distance is invariant over the class of equivalent encoding matrices. Thus, the j-th order column distance of $\mathcal{C}$ is the j-th column distance of any encoding matrix of C.

It is reported in [11] that the column distances of an encoding matrix satisfy the following conditions:

(i) $d_j^c \leq d_{j+1}^c, j = 0, 1, \dots$;

(ii) the sequence $d_0^c, d_1^c, d_2^c, \dots$ is bounded above; and

(iii) d_j^c becomes stationary as j increases.

In other words, d_j^c is a non-decreasing function of j. Thus, we have

$$d_\infty^c = \lim_{j \to \infty} d_j^c.$$

Using the above result, it can be shown that

$$d_{free} = d_\infty^c.$$

Therefore, the column distances given by

$$d_0^c \leq d_1^c \leq d_2^c \leq \dots \leq d_\infty^c$$

will eventually give the value of d_{free}. However it is much practical, even though possibly hard, to determine at what particular value of j this sequence becomes stationary.

We now introduce the row distances of a code. From a computational point of view, the *j-th order row distance* d_j^r of a polynomial encoder $G(D)$ is given by

$$d_j^r = \min_{u_{[0,j]} \neq 0} \left\{w_H\left(u_{[0,j]} G_j^r\right)\right\}, \tag{4}$$

where

$$G_j^r = \begin{pmatrix} G_0 & G_1 & \dots & G_m & & & \\ & G_0 & G_1 & \dots & G_m & & \\ & & \ddots & & & \ddots & \\ & & & G_0 & G_1 & \dots & G_m \end{pmatrix} \tag{5}$$

derived in (1) by truncating G at its first $j+1$ rows.

The row distances of a generator matrix $G(D)$ satisfy the following conditions:

(i) $d_{j+1}^r \leq d_j^r, j = 0, 1, \ldots$;

(ii) $d_j^r > 0, j = 0, 1, \ldots$; and

(iii) d_j^r becomes stationary as j increases.

It follows that the row distances of a code is a non-increasing function of j. The j-th row distance of $G(D)$ is also given by

$$d_j^r = d_H(G_j^r),$$

where $d_H\left(G_j^r\right)$ is the minimum Hamming distance of the linear block code generated by G_j^r.

The following are given in [11]:

$$0 < d_0^c \leq d_1^c \leq d_2^c \leq \ldots \leq d_\infty^c \leq d_\infty^r \leq \ldots \leq d_2^r \leq d_1^r \leq d_0^r \tag{6}$$

and if $G(D)$ is basic (or non-catastrophic (see [11] or [12])),

$$d_\infty^c = d_{free} = d_\infty^r. \tag{7}$$

The inequalities in (6) and the equation in (7) play the main role in creating the computer search algorithm presented in the next section.

3. THE COMPUTER SEARCH ALGORITHM

The main objective of the search is to find for polynomial encoders $G(D)$ of rate-k/n OFD binary convolutional codes whose free distances meet the Griesmer's upper bound. The idea is to search through the set of all possible $k \times n(m+1)$ binary matrices $\begin{pmatrix} G_0 & G_1 & \cdots & G_m \end{pmatrix}$, G_is are $k \times n$ matrices and G_0 is of full rank, such that the convolutional code generated by the polynomial encoder $G(D) = G_0 + G_1 D + \cdots + G_m D^m$ is an OFD code. For convenience, we let $d_H(M)$ be the minimum Hamming distance of the linear block code generated by a matrix M. Moreover, we let $d_{opt}^{k,n,m}$ be the Griesmer's upper bound on the free distance for the rate-k/n binary convolutional code of memory m.

In general, the algorithm is given by the following steps:

1. Collect in S_0 all $k \times n(m+1)$ binary matrices given by

   $$M_0 = \begin{pmatrix} G_0 & G_1 & \cdots & G_m \end{pmatrix},$$

   where G_0 is of full rank, satisfying $d_H(M_0) = d_{opt}^{k,n,m}$.

2. Each matrix $M_0 \in S_0$ is used to form the corresponding $k(j+1) \times n(m+1)$ matrix

   $$M_j = \begin{pmatrix} G_0 & G_1 & \ldots & G_m & & \\ & G_0 & G_1 & \ldots & G_m & \\ & & \ddots & & & \ddots \\ & & & G_0 & G_1 & \ldots & G_m \end{pmatrix}$$

   given in (5), for a chosen value of j.

3. All matrices $M_0 \in S_0$ with corresponding matrices M_j in Step 2 satisfying $d_H(M_j) = d_{opt}^{k,n,m}$ are collected in S_1. Note that $|S_1| \leq |S_0|$. Consequently, the j-th order row distances (from 0 up to j) of the polynomial encoder $G(D) = G_0 + G_1 D + \cdots + G_m D^m$ determined by a given $M_0 \in S_1$ are all equal to $d_{opt}^{k,n,m}$. That is, $d_0^r = d_1^r = \cdots = d_j^r = d_{opt}^{k,n,m}$ because of the results given in (6) and (7).

4. Each matrix $M_0 \in S_1$ in Step 3 is used to construct the corresponding $k(i+1) \times n(i+1)$ binary matrix

   $$N_i = \begin{pmatrix} G_0 & G_1 & G_2 & \ldots & G_i \\ & G_0 & G_1 & & G_{i-1} \\ & & G_0 & & G_{i-2} \\ & & & \ddots & \vdots \\ & & & & G_0 \end{pmatrix},$$

   given in (3). For all $M_0 \in S_1$, save the corresponding matrices N_i in the following manner: N_0 in T_0, N_1 in T_1, N_2 in T_2 ... N_i in T_i, where the stopping value of i is obtained in the last step.

5. Simultaneously with Step 4, increment i starting from $i = 0$ and obtain the i-th order column distances d_i^c using the matrices $N_i \in T_i$. If there exists L such that

   $$d_L^c = d_{opt}^{k,n,m}, \tag{8}$$

   for some $N_L \in T_L$, the search ends. All matrices M_0 that corresponds to N_L satisfying (8) are collected in S_2.

As a result, the set of polynomial encoders $G(D) = G_0 + G_1 D + \cdots + G_m D^m$ determined by matrices $M_0 \in S_2$ are all encoders of rate-k/n OFD binary convolutional codes of memory m with $d_{free} = d_{opt}^{k,n,m}$ since $d_j^r = d_{opt}^{k,n,m} = d_L^c$ for some positive integers j and L.

4. THE RATE-1/2 AND RATE-2/4 OFD BINARY CONVOLUTIONAL CODES

The algorithm given in Section 3 is utilized first to search for rate-1/2 OFD binary convolutional codes of memory $1 \leq m \leq 5$. The results are summarized in Table 2.

Table 2: The number of rate-1/2 OFD convolutional codes of memory $1 \leq m \leq 5$ obtained from the search.

m	$\lvert S_0 \rvert$	$\lvert S_1 \rvert$	L	$\lvert S_2 \rvert$	Minimal-basic	d_{free}
1	12	1	0	1	0	4
2	48	2	5	2	2	5
3	192	16	8	2	2	6
4	768	156	7	2	0	6
5	3072	168	13	4	4	8

Table 3 gives the minimal-basic encoders obtained in the search.

It must be noted that the given encoders described in Tables 2 and 3 are not necessarily new. In fact, sufficient studies on ODP and OFD rate-1/2 convolutional codes of memory $m \geq 20$ have been done already. See for instance [6], [7], [1] and [8]. However, the encoders obtained using the proposed

Table 3: The 1×2 minimal-basic encoders obtained from the search.

m	Minimal-basic	d_{free}
2	$\begin{pmatrix} 1+D^2 & 1+D+D^2 \end{pmatrix}$	5
	$\begin{pmatrix} 1+D+D^2 & 1+D^2 \end{pmatrix}$	5
3	$\begin{pmatrix} 1+D^2+D^3 & 1+D+D^2 \end{pmatrix}$	6
	$\begin{pmatrix} 1+D+D^2 & 1+D^2+D^3 \end{pmatrix}$	6
5	$\begin{pmatrix} 1+D+D^2+D^3+D^5 & 1+D^2+D^3 \end{pmatrix}$	8
	$\begin{pmatrix} 1+D+D^2+D^3+D^5 & 1+D^3+D^4 \end{pmatrix}$	8
	$\begin{pmatrix} 1+D+D^2+D^3 & 1+D^2+D^3+D^5 \end{pmatrix}$	8
	$\begin{pmatrix} 1+D+D^3+D^4 & 1+D+D^2+D^3+D^5 \end{pmatrix}$	8

algorithm are used to validate the MAGMA® routines that were created to materialize this search. Thus, the authors are focusing instead on finding OFD codes of rate 2/4.

In searching for rate-2/4 OFD binary convolutional codes of memory $m = 2$, Step 1 of the general algorithm has been modified. It is basically because of the very large ensemble of 2×12 binary matrices given by

$$M_0 = \begin{pmatrix} G_0 & G_1 & G_2 \end{pmatrix},$$

G_i(s) are 2×4 binary matrices, that were used supposedly as the basis for the 2×4 polynomial encoders $G(D) = G_0 + G_1 D + G_2 D^2$. Instead, the search starts from the set of 2×4 binary matrices that will be used to form M_0. Whereas, we have the following.

1. Consider the set of all distinct full-rank 2×4 binary matrices G_i.
2. From Step 1, save in S all matrices G_i such that $d_H(G_i) = 2$.
3. Form the set given by
$$S_0 = \left\{ M_0 = \begin{pmatrix} G_0 & G_1 & G_2 \end{pmatrix} | G_0, G_1, G_2 \in S \right\}.$$
4. Proceed using the general algorithm discussed in Section 3 starting from Step 2.

Table 4 summarizes the results for the search of rate-2/4 OFD binary convolutional codes of memory $m = 2$. Each polynomial encoder has the column distances given by $2, 3, 4, 5, 6, 6, 7, 8, 8, \ldots$.

Table 4: The number of rate-2/4 OFD binary convolutional codes of memory $m = 2$ obtained from the search.

$\lvert S_0 \rvert$	$\lvert S_1 \rvert$	L	$\lvert S_2 \rvert$	Minimal-basic
1560	186	7	52	52

The 52 distinct minimal-basic polynomial encoders are found in Appendix A.

5. SUMMARY AND REMARKS

Preliminaries and definitions that are relevant in this paper have been discussed. A computer search algorithm that employs column distances and row distances of rate-k/n binary convolutional codes was presented. The algorithm was used to find for rate-1/2 and rate-2/4 OFD binary convolutional codes. Rate-1/2 OFD binary convolutional codes of memory $1 \leq m \leq 5$ were obtained and minimal-basic encoders were also derived. For rate-2/4 OFD binary convolutional codes of memory $m = 2$, the general algorithm has been modified due to very large ensemble of 2×12 binary matrices. There were 52 distinct minimal-basic 2×4 polynomial encoders of memory $m = 2$ obtained in the search. The computer program was developed using MAGMA® on a personal computer (*Acer Veriton 7200, Intel® 1.6GHz, SDRAM 2GB*).

Because of the limited computing capacity of personal computers, the authors aim to do some more refinements on the algorithm to improve the results. The plan is twofold: (1) obtain a more efficient algorithm, hopefully an algebraic one, by limiting the search to a much smaller ensemble of binary matrices that will serve as the building blocks for the encoders of OFD and ODP binary convolutional codes, and (2) compare the result of the search to the existing OFD and ODP codes to determine which of these codes are superior in terms of a given criteria.

6. REFERENCES

[1] M. Cedervall and R. Johannesson. A fast algorithm for computing distance spectrum of convolutional codes. *IEEE Trans. Inform. Theory*, 35(6):1146 - 1159, November 1989.

[2] P. Chevillat and D.J. Costello, Jr. Distance and computation in sequential decoding. *IEEE Trans. Inform. Theory*, COM-24:440 - 447, April 1976.

[3] D. Costello,Jr. A construction technique for random-error-correcting convolutional codes. *IEEE Trans. Inform. Theory*, IT-15:631 - 636, September 1969.

[4] E. Paaske. Short binary convolutional codes with maximal free distance for rates 2/3 and 3/4. *IEEE Trans. Inform. Theory*, IT-20:683 - 689, September 1974.

[5] G. D. Forney, Jr. Convolutional codes I: Algebraic structure. *IEEE Trans. Inform. Theory*, IT-16(6):720 - 738, November 1970.

[6] R. Johannesson. Robustly optimal rate one-half binary convolutional codes. *IEEE Trans. Inform. Theory*, IT-21:464 - 468, July 1975.

[7] R. Johannesson. Some rate 1/3 and 1/4 binary convolutional codes with an optimum distance profile. *IEEE Trans. Inform. Theory*, Correspondence:281 - 283, March 1977.

[8] R. Johannesson. New rate 1/2, 1/3, and 1/4 binary convolutional encoders with an optimum distance profile. *IEEE Trans. Inform. Theory*, 45(5):281 - 283, July 1999.

[9] R. Johannesson and Emma Wittenmark. Two 16-state, rate r=2/4 trellis codes whose free distances meet the heller bound. *IEEE Trans. Inform. Theory*, 44(4):1602 Ú 1604, July 1998.

[10] R. Johannesson and Erik Paaske. Further results on

binary convolutional codes with an optimum distance profile. *IEEE Trans. Inform. Theory*, IT-24(2):264 - 268, March 1978.

[11] R. Johannesson and K. Sh. Zigangirov. *Fundamentals of Convolutional Coding*. New York: IEEE Press, 1999.

[12] R. J. McEliece. The algebraic theory of convolutional codes. In V. S. Pless and W. C. Huffman, editors, *Handbook of Coding Theory*, chapter 12, pages 1065 - 1138. Amsterdam, The Netherlands: North-Holland, Elsevier, 1998.

[13] A. J. Viterbi. Convolutional codes and their performance in communication systems. *IEEE Trans. Inform. Theory*, COM-19:751 - 772, October 1971.

APPENDIX

A. THE 2×4 POLYNOMIAL ENCODERS FOR RATE-$2/4$ OFD BINARY CONVOLUTIONAL CODES.

$$\begin{pmatrix} D^2 & D^2+D & D+1 & D^2+D+1 \\ D+1 & 1 & D^2+D+1 & D^2+D \end{pmatrix}$$

$$\begin{pmatrix} D^2+D & D^2 & D+1 & D^2+D+1 \\ 1 & D+1 & D^2+D+1 & D^2+D \end{pmatrix}$$

$$\begin{pmatrix} D^2 & D+1 & D^2+D & D^2+D+1 \\ D+1 & D^2+D+1 & 1 & D^2+D \end{pmatrix}$$

$$\begin{pmatrix} D^2 & D+1 & D^2+D+1 & D^2+D \\ D+1 & D^2+D+1 & D^2+D & 1 \end{pmatrix}$$

$$\begin{pmatrix} 0 & D^2+1 & D^2+D+1 & D^2+D+1 \\ D^2+D+1 & D & D & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} 0 & D^2+D+1 & D^2+1 & D^2+D+1 \\ D^2+D+1 & D & D & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} 0 & D^2+1 & D^2+D+1 & D^2+D+1 \\ D^2+D+1 & D & D^2+D+1 & D \end{pmatrix}$$

$$\begin{pmatrix} 0 & D^2+D+1 & D^2+D+1 & D^2+1 \\ D^2+D+1 & D & D^2+D+1 & D \end{pmatrix}$$

$$\begin{pmatrix} 0 & D^2+D+1 & D^2+1 & D^2+D+1 \\ D^2+D+1 & 0 & D^2+D+1 & D^2+1 \end{pmatrix}$$

$$\begin{pmatrix} D^2 & D+1 & D^2+1 & D^2+D+1 \\ D+1 & D^2 & D^2+D+1 & D^2+1 \end{pmatrix}$$

$$\begin{pmatrix} 0 & D^2+D+1 & D^2+D+1 & D^2+1 \\ D^2+D+1 & 0 & D^2+1 & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} D^2 & D+1 & D^2+D+1 & D^2+1 \\ D+1 & D^2 & D^2+1 & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} D & D^2+1 & D^2+1 & D^2+D+1 \\ D^2+1 & D & D^2+D+1 & D^2+1 \end{pmatrix}$$

$$\begin{pmatrix} D^2+D & 1 & D^2+1 & D^2+D+1 \\ 1 & D^2+D & D^2+D+1 & D^2+1 \end{pmatrix}$$

$$\begin{pmatrix} D & D^2+1 & D^2+D+1 & D^2+1 \\ D^2+1 & D & D^2+D+1 & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} D^2 & 1 & D^2+D+1 & D^2+D \\ D+1 & D^2+D+1 & D^2 & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} D^2 & 1 & D^2+D+1 & D^2+D \\ D+1 & D^2+D+1 & D^2+D & D^2 \end{pmatrix}$$

$$\begin{pmatrix} D^2 & 1 & D^2+D+1 & D^2+D+1 \\ D+1 & D^2+D+1 & D^2+D & D^2 \end{pmatrix}$$

$$\begin{pmatrix} 0 & D^2+D+1 & D^2+1 & D^2+D+1 \\ D^2+D+1 & D^2+D+1 & D & D \end{pmatrix}$$

$$\begin{pmatrix} 0 & D^2+D+1 & D^2+D+1 & D^2+1 \\ D^2+D+1 & D^2+D+1 & D & D \end{pmatrix}$$

$$\begin{pmatrix} 0 & D^2+1 & D^2+D+1 & D^2+D+1 \\ D^2+D+1 & D^2+D+1 & 0 & D^2+1 \end{pmatrix}$$

$$\begin{pmatrix} 0 & D^2+D+1 & D^2+D+1 & D^2+1 \\ D^2+D+1 & D^2+1 & 0 & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} D & D^2+1 & D^2+1 & D^2+D+1 \\ D^2+1 & D^2+D+1 & D & D^2+1 \end{pmatrix}$$

$$\begin{pmatrix} D^2+D & 1 & D^2+D+1 & D+1 \\ 1 & D^2+D+1 & D^2 & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} D & D^2+D+1 & D^2+1 & D^2+1 \\ D^2+1 & D^2+1 & D & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} 0 & D^2+1 & D^2+D+1 & D^2+D+1 \\ D^2+D+1 & D^2+1 & 0 & D^2+1 \end{pmatrix}$$

$$\begin{pmatrix} 0 & D^2+D+1 & D^2+D+1 & D^2+1 \\ D^2+D+1 & D^2+1 & D^2+D+1 & 0 \end{pmatrix}$$

$$\begin{pmatrix} D^2+D+1 & D^2+D+1 & D^2+1 & 0 \\ D^2+1 & D^2+D+1 & D^2+1 & D \end{pmatrix}$$

$$\begin{pmatrix} D^2+D & 1 & D+1 & D^2+D+1 \\ 1 & D^2+D+1 & D^2+D+1 & D^2 \end{pmatrix}$$

$$\begin{pmatrix} D & D^2+D+1 & D^2+1 & D^2+1 \\ D^2+1 & D^2+1 & D^2+D+1 & D \end{pmatrix}$$

$$\begin{pmatrix} D+1 & D^2 & D^2+D & D^2+D+1 \\ D^2 & D^2+D+1 & D^2+D+1 & 1 \end{pmatrix}$$

$$\begin{pmatrix} D+1 & D^2 & D^2+D+1 & D^2+D \\ D^2 & D^2+D+1 & 1 & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} D^2+1 & 0 & D^2+D+1 & D^2+D+1 \\ D & D^2+D+1 & D & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} D^2+1 & D & D^2+D+1 & D^2+1 \\ D & D^2+D+1 & D & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & D^2+D & D+1 & D^2+D+1 \\ D^2+D & D^2+D+1 & D^2 & D+1 \end{pmatrix}$$

$$\begin{pmatrix} D^2+1 & 0 & D^2+D+1 & D^2+D+1 \\ D & D^2+D+1 & D^2+D+1 & D \end{pmatrix}$$

$$\begin{pmatrix} D^2+1 & D & D^2+1 & D^2+D+1 \\ D & D^2+D+1 & D^2+D+1 & D \end{pmatrix}$$

$$\begin{pmatrix} 1 & D^2+D & D^2+D+1 & D+1 \\ D^2+D & D^2+D+1 & D+1 & D^2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & D^2 & D^2+D+1 & D^2+D+1 \\ D^2+D & D^2+D+1 & 1 & D+1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & D^2 & D^2+D+1 & D^2+D+1 \\ D^2+D & D^2+D+1 & D+1 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & D^2+D & D^2+1 & D^2+D+1 \\ D^2+D & 1 & D^2+D+1 & D^2+1 \end{pmatrix}$$

$$\begin{pmatrix} D^2+1 & D & D^2+1 & D^2+D+1 \\ D & D^2+1 & D^2+D+1 & D^2+1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & D^2+D & D^2+D+1 & D^2+1 \\ D^2+D & 1 & D^2+1 & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} D^2+1 & D & D^2+D+1 & D^2+1 \\ D & D^2+1 & D^2+1 & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} D+1 & D^2 & D^2+1 & D^2+D+1 \\ D^2 & D+1 & D^2+D+1 & D^2+1 \end{pmatrix}$$

$$\begin{pmatrix} D^2+D+1 & 0 & D^2+1 & D^2+D+1 \\ 0 & D^2+D+1 & D^2+D+1 & D^2+1 \end{pmatrix}$$

$$\begin{pmatrix} D+1 & D^2 & D^2+D+1 & D^2+1 \\ D^2 & D+1 & D^2+1 & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} D^2+D+1 & 0 & D^2+D+1 & D^2+1 \\ 0 & D^2+D+1 & D^2+1 & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} D^2+1 & D^2+D+1 & 0 & D^2+D+1 \\ D & D & D^2+D+1 & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} D^2+1 & D^2+D+1 & D & D^2+1 \\ D & D & D^2+D+1 & D^2+D+1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & D+1 & D^2+D & D^2+D+1 \\ D^2+D & D^2 & D^2+D+1 & D+1 \end{pmatrix}$$

$$\begin{pmatrix} D+1 & 1 & D^2+D & D^2+D+1 \\ D^2 & D^2+D & D^2+D+1 & D+1 \end{pmatrix}$$