Extension of the BCH Decoding Algorithm to Decode Binary Cyclic Codes up to Their Maximum Error Correction Capacities

PATRICK STEVENS

Abstract—**The BCH algorithm can be extended to correct more errors than indicated by the BCH bound. In the first step of the decoding procedure, we correct a number of errors, corresponding to a particular case of the Hartmann–Tzeng bound. In the second step we aim at full error correction. A measure for the worst-case number of field elements of an extension field GF(2^m) that must be tested for this purpose is given for binary cyclic linear unequal error protection codes as well as for conventional binary cyclic codes.**

I. Introduction

Binary cyclic codes can be decoded by the well-known BCH algorithm up to the BCH bound and even up to a particular case of the Hartmann–Tzeng (HT) bound. We denote by t_δ the number of errors that can be straightforwardly decoded in the so-called first step of the decoding procedure. In Section II we summarize the algorithm and examine what may happen when more than t_δ errors occur at the input of the decoder.

Our main interest is directed towards correcting more than t_δ errors in the second step of the decoding algorithm. When binary cyclic linear unequal error protection (LUEP) codes—briefly described in Section III—are involved, this touches upon the problem of reaching the higher error protection level of such a code. In the case of conventional cyclic codes, we actually search for a method of correcting a number of errors lying between $(t_\delta+1)$ and $t=\lfloor(d-1)/2\rfloor$, where d denotes the minimum distance of the code. In Section IV we describe how the BCH algorithm may be extended by a second step and at what cost. We define a code parameter μ^* that measures the price we must pay in exchange for full error-correcting capacity.

In Section V we apply the extended algorithm to the class of binary cyclic LUEP codes. We begin by discussing briefly a subclass of these codes that allows an immediate implementation of the second step of the decoding procedure. We compute the parameter μ^* for the class of all binary cyclic LUEP codes of length $n\le 39$, and we list the results. In Section VI we proceed in a similar way for all conventional binary cyclic codes of length $n\le 57$. Our main conclusion is that decoding via the extended BCH algorithm is of considerable practical interest for codes with a small value of μ^*.

Manuscript received June 1, 1987; revised November 1, 1987.

The author is with the Faculty of Applied Sciences, Department of Mathematical Analysis, Brussels Free University, Pleinlaan 2, 1050 Brussels, Belgium.

IEEE Log Number 8824497.

II. Generalities about Cyclic Codes

A binary cyclic code of odd length n is generated by a generator polynomial $g(X)$ which is a divisor of X^n+1. When $\deg g(X)=n-k$, $g(X)$ generates an (n,k) code C, we write $C=\langle g(X)\rangle$.

As $\gcd(n,2)=1$, a smallest positive integer m exists such that $2^m\equiv 1 \pmod n$; m is the modulo m multiplicative order of 2. GF(2^m) is the splitting field of X^n+1; its nonzero elements are generated by a primitive element β.

A primitive nth root of unity exists, say α, such that

$$X^n+1=\prod_{i=0}^{n-1}(X+\alpha^i).$$

The minimal polynomial of α^i, denoted $m_i(X)$, is the lowest degree polynomial over $\mathbb{F}=\mathrm{GF}(2)$ having α^i as a zero. All the zeros of $m_i(X)$ are α^i and its conjugates; their exponents form the cyclotomic coset C_i modulo n. We denote by K_n the set of all minimal representative indices of cyclotomic cosets modulo n.

K_n is partitioned into the subsets K and $\bar{K}$ such that

- $i\in K$ if and only if α^i is a nonzero of the code C,
- $j\in\bar{K}$ if and only if α^j is a zero of the code C.

Hence

$$g(X)=\prod_{i\in\bar{K}}m_i(X).$$

The defining set R of the code is

$$R=\bigcup_{i\in\bar{K}}C_i.$$

Example: $n=15$, $\alpha\in\mathrm{GF}(2^4)$; C is the (15,7) code having $g(X)=m_1(X)\cdot m_3(X)$ and

$$K_n=\{0,1,3,5,7\}\qquad K=\{0,5,7\}$$
$$\bar{K}=\{1,3\}\qquad R=\{1,2,4,8,3,6,12,9\}.$$

BCH Bound

A cyclic code of length n with zeros $\alpha^b,\alpha^{b+r},\cdots,\alpha^{b+(\delta-2)r}$ has minimum distance at least δ. The integers b, δ, and r are such that $b \geq 0$, $\delta \geq 1$, and $\gcd(r, n) = 1$. In other words, when the defining set R of a cyclic code contains a string of $\delta - 1$ integers $b + \{0, r, 2r, \cdots, (\delta - 2)r\} \pmod{n}$ as a subset, then $d \geq \delta$. The parameter $\delta_{\mathrm{BCH}} = \delta$ is called the BCH bound or the designed distance of the code.

HT Bound

A cyclic code of length n with zeros $\alpha^{b+i_1r_1+i_2r_2}$ for all $i_1 = 0, 1, \cdots, \delta - 2$ and all $i_2 = 0, 1, \cdots, s$ has minimum distance at least $\delta + s$. The integers b, δ, r_1, and r_2 are such that $b \geq 0$, $\delta \geq 1$, and $\gcd(r_1, n) = \gcd(r_2, n) = 1$. Otherwise stated, when the defining set contains $s+1$ strings of $\delta - 1$ integers satisfying the abovementioned conditions, then $d \geq \delta + s$.

The parameter $\delta_{\mathrm{HT}} = \delta + s$ is called the HT bound of the code. A case of special interest is $s + 1 = t_\delta$ and $\delta - 1 = t_\delta + 1$; then $\delta_{\mathrm{HT}} = 2t_\delta + 1$ and thus the code is at least t_δ-error-correcting. Its defining set contains t_δ strings of $t_\delta + 1$ integers as a subset, which we can arrange in a (t_δ by $t_\delta + 1$) array. As this case of the HT bound will henceforth be of special importance, we introduce the concept of the HT array, corresponding to an error-capacity of at least τ.

Definition: $\mathrm{HT}(\tau) = (w_{ij})$, $1 \leq i \leq \tau$, $1 \leq j \leq \tau + 1$; the entries of the array are integers such that $w_{11} \geq 0$ and $w_{i+\lambda, j+\mu} = w_{ij} + \mu r_1 + \lambda r_2 \pmod{n}$, where $\gcd(r_1, n) = \gcd(r_2, n) = 1$ and n is the length of the code.

The largest value τ for which $\mathrm{HT}(\tau) \subset R$ is defined as t_δ; the HT bound of the code is then at least $2t_\delta + 1$.

Note that the string of $\delta - 1$ integers appearing in the construction of the BCH bound may also be regarded as an array $\mathrm{HT}(\tau)$, for $\tau = \lfloor (\delta - 1)/2 \rfloor$, $w_{11} = b$, $r_1 = r_2 = r$ (when δ is even, the last integer in the string is omitted). Consequently, $\lfloor (\delta_{\mathrm{BCH}} - 1)/2 \rfloor \leq t_\delta$.

Although the BCH algorithm was originally conceived to correct $\lfloor (\delta_{\mathrm{BCH}} - 1)/2 \rfloor$ or fewer errors, in the next paragraph we show that it is actually able to correct t_δ or fewer errors. For this reason, we argue that t_δ is the number of errors we can straightforwardly correct in the first step of what we shall call the extended BCH algorithm.

BCH Algorithm

The BCH algorithm is described in [1]–[3]; we will briefly outline its main features when $r = 1$. Consider a vector $\bar{e} = (e_0, e_1, \cdots, e_{n-1})$ with ν nonzero components $e_{i_1}, e_{i_2}, \cdots, e_{i_\nu}$. The associated polynomial is $e(X) = \sum_{i=0}^{n-1} e_i X^i$.

The locators of the vector $\bar{e}$ are the following ν elements of $\mathrm{GF}(2^m)$:

$$\zeta_1 = \alpha^{i_1},\ \zeta_2 = \alpha^{i_2}, \cdots, \zeta_\nu = \alpha^{i_\nu}.$$

We define the power sums

$$s_l = e(\alpha^l) = \sum_{\gamma=1}^{\nu} \zeta_\gamma^l, \qquad l = 0, 1, \cdots .$$

The locator polynomial $\sigma(z)$ of the vector $\bar{e}$ is

$$\sigma(z) = \prod_{i=1}^{\nu} (1 - \zeta_i z) = \sum_{i=0}^{\nu} \sigma_i z^i, \qquad \sigma_0 = 1.$$

The zeros of $\sigma(z)$ are the inverses of the locators.

The Newton identities are a set of linear equations that relate the power sums S_l to the coefficients σ_i of the locator polynomial. The recurrence relation is

$$S_{j+\nu} + \sigma_1 \cdot S_{j+\nu-1} + \cdots + \sigma_\nu \cdot S_j = 0, \qquad \forall\, j.$$

When an error pattern of weight ν, $\nu \leq \lfloor (\delta_{\mathrm{BCH}} - 1)/2 \rfloor$, occurs, it is corrected by the following steps.

- For all $j \in R$, compute the power sum $S_j = e(\alpha^j) = r(\alpha^j)$, where $r(X)$ denotes the polynomial corresponding to the received word $\bar{r}$.
- Solve the Newton identities for $\sigma_1, \sigma_2, \cdots, \sigma_\nu$.
- Find the zeros of the locator polynomial $\sigma(z)$.
- Compute their inverses $\zeta_1, \zeta_2, \cdots, \zeta_\nu$; this determines $e(X)$.

This algorithm can, in fact, be applied to correct t_δ errors, as we shall presently explain. We assume that an array $\mathrm{HT}(t_\delta)$ is known and that ν ($\nu \leq t_\delta$) errors occur. We do not restrict ourselves to $r_1 = 1$; this fact induces the following modifications.

- Define

$$\sigma(z) = \prod_{i=1}^{\nu} (1 - \zeta_i^{r_1} z) = \sum_{i=0}^{\nu} \sigma_i z^i, \qquad \sigma_0 = 1.$$

- The Newton identities turn out to be

$$S_{j+\nu r_1} + \sigma_1 \cdot S_{j+(\nu-1)r_1} + \cdots + \sigma_\nu \cdot S_j = 0.$$

Consecutively setting $j = w_{11}, w_{21}, \cdots, w_{\nu 1}$ yields

$$S_{w_{i1}+\nu r_1} + \sigma_1 \cdot S_{w_{i1}+(\nu-1)r_1} + \cdots + \sigma_\nu \cdot S_{w_{i1}} = 0, \qquad 1 \leq i \leq \nu$$

or

$$S_{w_{i,\nu+1}} + \sigma_1 \cdot S_{w_{i\nu}} + \cdots + \sigma_\nu \cdot S_{w_{i1}} = 0, \qquad 1 \leq i \leq \nu.$$

The indices of the power sums involved in this set of linear equations correspond to the entries of $\mathrm{HT}(\nu) \subseteq \mathrm{HT}(t_\delta)$. As we can prove that the (ν by ν) matrix $(S_{w_{ij}})$, $1 \leq i, j \leq \nu$, is nonsingular, we find a unique solution $(\sigma_1, \sigma_2, \cdots, \sigma_\nu)$ that determines $\sigma(z)$.

- After the zeros of $\sigma(z)$ have been computed and inverted, we end up with $\zeta_i^{r_1}$, $1 \leq i \leq \nu$.

The locators ζ_i themselves can be easily obtained by recalling that integers u, v exist such that $ur_1 + vn = 1$, because $\gcd(r_1, n) = 1$. Hence $(\zeta_i^{r_1})^u = \zeta_i$.

Output Modes of the BCH Decoder when $wt(\bar{e}) > t_\delta$

We now describe in more detail what may happen when $wt(\bar{e}) > t_\delta$, i.e., the weight of the error pattern exceeds the error-correcting capacity of the BCH decoder. Little attention has been directed to this question in the literature. Usually, the assumption $wt(\bar{e}) \leq t_\delta$ is made *a priori*.

1) It is possible that the coset of $\bar{e}$, denoted as Coset

$(\bar{e})$, contains a vector $\bar{e}'$ of weight t_δ or less. The decoder outputs the codeword $\bar{c}' = \bar{r} + \bar{e}' \neq \bar{c}$. This is an erroneous decoding decision as $\bar{c}' \neq \bar{c}$. We will later see that, with respect to LUEP codes, this decoding is nevertheless adequate, as it allows the correct retrieval of the high-order information bits.

When Coset $(\bar{e})$ does not contain a vector $\bar{e}'$ of weight t_δ or less, the decoder outputs a failure. The failure mode can have different causes, which are stated below.

2) The Newton identities do not yield appropriate values for the coefficients σ_i.

3) The number of distinct zeros of $\sigma(z)$ belonging to $GF(2^m)$ is less then $\deg \sigma(z)$. This is the case, for instance, when $\sigma(z)$ has no zeros at all in $GF(2^m)$.

4) The decoder computes a potential error pattern $\bar{e}''$ of weight t_δ or less, but $\bar{c}'' = \bar{r} + \bar{e}''$ turns out to be not a codeword (easy to verify by ascertaining that $c''(\alpha^j) \neq 0$ for some $j \in R$). The decoder then outputs a failure.

III. Binary Cyclic LUEP Codes

The separation vector $\bar{s}(G)$ of a binary LUEP code is defined as follows:

$$s(G)_i = \min\left\{ wt(\bar{u}G) | \bar{u} \in \mathbb{F}^k,\ u_i \neq 0 \right\}, \qquad 0 \leq i \leq k-1.$$

$\bar{u} = (u_0, u_1, \cdots, u_{k-1})$ denotes the information word which is encoded into the codeword $\bar{c} = \bar{u}G$ for G, the generator matrix of the code.

The ith information bit is protected against $\tau(G)_i = \lfloor (s(G)_i - 1)/2 \rfloor$ errors $(0 \leq i \leq k-1)$. We define $\bar{\tau}(G) = (\tau(G)_0, \cdots, \tau(G)_{k-1})$ as the protection vector of the code, which depends on the encoding chosen.

We restrict ourselves to the study of binary cyclic LUEP codes whose protection vector has only two distinct values for its components. We write

$$\bar{\tau}(G) = \underbrace{(\underbrace{t', t', \cdots, t'}_{k'}, t, t, \cdots, t)}_{k}, \qquad 1 \leq t < t'.$$

This means that all information bits are protected against t errors and, in addition, that k' high-order information bits are protected against t' errors.

We assume that t or fewer errors can be corrected by the BCH algorithm, i.e., $t = t_\delta$. This assumption is satisfied by a considerable number of codes, as can be seen in the table of all binary cyclic LUEP codes of length $n \leq 39$ [4]. Later on, we will examine what can be done when $t_\delta < t$.

We will now describe the application of the BCH algorithm as the first step in the decoding procedure of a binary cyclic LUEP code.

a) When an error pattern of weight t or less occurs, it is corrected by the decoder.

b) When an error pattern of weight $t < wt(\bar{e}) \leq t'$ occurs and the decoder outputs a codeword $\bar{c}'$, it is possible that $\bar{c}'$ is not identical to the original codeword $\bar{c}$. However, the corresponding information words $\bar{u}'$ and $\bar{u}$, whose retrieval requires a transformation of the codeword as the code is not systematic (see [5]), do have the same high-order bits $u_i' = u_i$ for $0 \leq i \leq k'-1$; thus with respect to LUEP codes, this is quite an adequate decoding algorithm.

Proof: $\bar{r} = \bar{c} + \bar{e}$, $\bar{c}' = \bar{r} + \bar{e}'$, $wt(\bar{e}') \leq t < t'$ (see Section II), $wt(\bar{e}) \leq t'$. Hence

$$wt(\bar{e} + \bar{e}') < 2t' \quad t' = \lfloor (s(G)_i - 1)/2 \rfloor, \quad 0 \leq i \leq k'-1$$

or

$$wt(\bar{c} + \bar{c}') < s(G)_i, \qquad 0 \leq i \leq k'-1;$$

thus $u_i = u_i'$ for $0 \leq i \leq k'-1$.

From the receiver's point of view, the two cases mentioned are indistinguishable. When a codeword is output by the decoder, it may either be the one transmitted or any other one that yields the same high-order information bits. However, the former possibility is the more likely for two reasons:

- $wt(\bar{e}) \leq t$ is more probable than $t < wt(\bar{e}) \leq t'$;
- the latter case only occurs when $t < wt(\bar{e}) \leq t'$ and Coset $(\bar{e})$ contains a vector $\bar{e}'$ of weight t or less. This implies that $(\bar{e} + \bar{e}')$ must be a codeword of weight $(t + t')$ or less, i.e., small weight. As codewords of small weight are relatively rare, the latter possibility occurs far less frequently than the former.

c) When the decoder outputs a failure, we known with certainty that more than t errors have occurred. We take for granted that the number of errors is smaller than t' because this is the ultimate error correction capacity.

We now proceed with the second step of the decoding procedure, which only attempts to determine the high-order information bits. After a general description of the extended BCH algorithm, its application to binary cyclic LUEP codes is given (Section V).

IV. Description of the Extended BCH Algorithm

In Section II we observed that t_δ or fewer errors can be corrected in the first step of the decoding procedure. If the algorithm outputs a failure in its first step, that should imply that more than t_δ, say τ, errors have occurred. Yet we would still like to apply the BCH algorithm to obtain the transmitted codeword (in the case of conventional codes) or at least one that contains the same high-order information bits (in the case of LUEP codes).

We still use the Newton identities in relation to the entries of some array $HT(\tau)$ (see Section II) which is, however, no longer a subset of the defining set of the code. We denote by N' $(N' \geq 1)$ the number of power sums S_{q_j} that now appear as unknown quantities in the Newton identities. As $S_{2k} = S_k^2$, these S_{q_j} need not all be independent. To obtain a set of independent parameters, we only consider power sums whose indices belong to distinct cyclotomic cosets modulo n. In doing so, we end up with an N-tuple $(S_{p_1}, S_{p_2}, \cdots, S_{p_N})$ of power sums that appear as independent unknown quantities in the Newton identities. In accordance with the notation in Section II, $p_i \in K$ $(1 \leq i \leq N)$ implies α^{p_i} is a nonzero of the code.

Property: Let $p \in K$; the power sum $S_p = e(\alpha^p)$ takes 2^{b_p} different values in $\mathrm{GF}(2^m)$ (recall that $b_p = \deg m_p(X)$). Indeed, as $\alpha^p \in \mathrm{GF}(2^{b_p})$, it follows that

$$\{e(\alpha^p) \colon e(X) \in \mathbb{F}[X]/X^n+1\} = \mathrm{GF}(2^{b_p}).$$

We conclude that there exist 2^{μ} different N-tuples $(S_{p_1}, S_{p_2}, \cdots, S_{p_N})$ or N'-tuples $(S_{q_1}, S_{q_2}, \ldots, S_{q_{N'}})$ over $\mathrm{GF}(2^m)$, where μ is defined as $\mu = \sum_{i=1}^{N} \deg m_{p_i}(X)$.

If in the Newton identities we insert all these 2^{μ} different values for the parameters, we will find an error pattern that meets our purpose. The second step of the decoding procedure apparently requires at most 2^{μ} tests to correct τ errors.

It is useful to present the fundamental aspect of our decoding method. A cyclic code C is considered the direct sum of a τ-error-correcting subcode C' and some code C''. The received word $\bar{r} = \bar{c} + \bar{e}$ is decoded as follows. For each $\bar{c}'' \in C''$, decode $\bar{r} + \bar{c}''$ using the BCH algorithm that corrects τ errors in C'. When this decoding results in a codeword $\bar{c}' \in C'$, the final output is $\bar{c}' + \bar{c}'' \in C$. We are concerned with finding subcodes of C such that C'' is of minimal cardinality. To anticipate the notation of the next section, we state that

$$C'' = \bigoplus_{i=1}^{N} M_{p_i}.$$

It is clear that we aim at constructing an array $\mathrm{HT}(\tau)$ such that the value of μ is minimized. We call this an optimal array $\mathrm{HT}^*(\tau)$; it is determined by letting a computer program run exhaustively through the defining set of the code and perform the following operations.

1) $\forall\ b \in \{0, 1, \cdots, n-1\}$; $\forall\ r_1, r_2 \in \{1, 2, \cdots, n-1\}$, $\gcd(r_1, n) = \gcd(r_2, n) = 1$: construct the array $\mathrm{HT}(\tau)(b, r_1, r_2)$, according to the definition in Section II, where $w_{11} = b$.
2) For each array $\mathrm{HT}(\tau)(b, r_1, r_2)$,
   a) consider the distinct $q_j \in (\mathrm{HT}(\tau) \setminus R)$, and denote their cardinality by N';
   b) select those integers that belong to distinct cyclotomic cosets modulo n, always choosing the smaller if two integers are in the same cyclotomic coset; denote them by $p_1, \cdots, p_N$;
   c) determine $\deg m_{p_i}(X) = \mathrm{card}\, C_{p_i}$ (number of integers in the cyclotomic coset C_{p_i} modulo n), $1 \le i \le N$;
   d) compute the sum $\mu(\tau, b, r_1, r_2) = \sum_{i=1}^{N} \deg m_{p_i}(X)$.
3) Select the minimum value μ^* among all μ corresponding to the optimal array $\mathrm{HT}^*(\tau)(b^*, r_1^*, r_2^*)$. This array is not unique; any one will do—for instance, the one with the smallest values of r_1^*, r_2^* and b^*.

Note: It is not difficult to see that in the first operation the ranges of r_1 and r_2 may be limited by the additional restrictions r_1 odd, $r_2 < n/2$.

We conclude that, to correct τ errors in the second step of the BCH decoding procedure, the Newton identities have to be filled in by the power sums $S_{w_{ij}}$, $w_{ij} \in \mathrm{HT}^*(\tau)$. Among these power sums, there is an unknown N-tuple $(S_{p_1}, \cdots, S_{p_N})$ over $\mathrm{GF}(2^m)$, having 2^{μ^*} distinct possible values which must all be tried until the decoder finds an error pattern of weight τ. If these possibilities are tried sequentially, 2^{μ^*} is an upper bound on the number of tests. If they are performed simultaneously, the computations may be stopped when any parallel circuit outputs an error pattern of weight τ. The results of the computer program searching for $\mathrm{HT}^*(\tau)$ are presented in the following sections.

V. Application to Cyclic LUEP Codes

We briefly describe some more concepts and notation in relation to cyclic codes, which can be found in more detail in [1], [4], and [5]. This allows us to deal immediately with the particular case $k' = 1$, $t' = t + 1$ (this means that one information bit is protected against one more error in comparison with the other information bits). The residue class ring of polynomials over $\mathbb{F}$ modulo $X^n + 1$ is $R_n = \mathbb{F}[X]/X^n + 1$.

A cyclic code C is an ideal in R_n; it is the direct sum of minimal ideals in R_n. One such minimal ideal M is generated by a primitive idempotent $\theta(X)$.

The nonzeros of the minimal ideal M_s are $\{\alpha^i | i \in C_s\}$. The generator polynomial of M_s is

$$g_s(X) = X^n + 1/m_s(X) = \prod_{i \in K_n \setminus \{s\}} m_i(X).$$

The dimension of the minimal ideal M_s is $\dim M_s = \deg m_s(X) = b_s$. A basis for M_s is $\theta_s(X), X\theta_s(X), \cdots, X^{b_s - 1}\theta_s(X)$ in R_n.

Let C be the direct sum of the minimal ideals $M_{k_1}, M_{k_2}, \cdots, M_{k_\kappa}$; $\dim M_{k_j} = b_{k_j}$. An information polynomial $u(X) = \sum_{i=0}^{k-1} u_i X^i$ is the concatenation of the polynomials $U_{k_1}(X) | U_{k_2}(X) | \cdots | U_{k_\kappa}(X)$, where

$$U_{k_j}(X) = \sum_{\lambda=0}^{b_{k_j}-1} u_{\gamma_j + \lambda} X^{\lambda} \qquad \gamma_j = \sum_{\mu=0}^{j-1} b_{k_\mu}, \quad 2 \le j \le \kappa, \ \gamma_1 = 0.$$

The encoding rule is

$$c(X) = \sum_{j=1}^{\kappa} U_{k_j}(X) \cdot \theta_{k_j}(X) = \sum_{i \in K} U_i(X) \cdot \theta_i(X) \quad \text{in } R_n.$$

In the particular case we are considering, the one high-order bit u_0 corresponds to the minimal ideal M_0, generated by

$$\theta_0(X) = (X^n + 1)/(X + 1) = \sum_{l=0}^{n-1} X^l.$$

As $0 \in K$, the encoding rule becomes

$$c(X) = u_0 \cdot \theta_0(X) + \sum_{i \in K \setminus \{0\}} U_i(X) \cdot \theta_i(X) \quad \text{in } R_n.$$

When an error pattern $e(X)$ of weight $(t+1)$ occurs, then

$$r(X) = u_0 \cdot \theta_0(X) + \sum_{i \in K \setminus \{0\}} U_i(X) \cdot \theta_i(X) + e(X) \text{ in } R_n.$$

$\theta_0(X)$ has odd weight; all other primitive idempotents have even weight, as they have $\alpha_0 = 0$ as a zero. Consequently, $\sum_{i \in K \setminus \{0\}} U_i(X) \cdot \theta_i(X) \bmod (X^n + 1)$ has even weight. We may conclude that

- when t is odd

$$\begin{cases} u_0 = 0 & \Rightarrow wt\, r(X) \text{ is even} \\ u_0 = 1 & \Rightarrow wt\, r(X) \text{ is odd} \end{cases},$$

- when t is even

$$\begin{cases} u_0 = 0 & \Rightarrow wt\, r(X) \text{ is odd} \\ u_0 = 1 & \Rightarrow wt\, r(X) \text{ is even} \end{cases}.$$

Thus

$$(t \text{ odd AND } wt(\bar{r}) \text{ even}) \text{ OR } (t \text{ even AND } wt(\bar{r}) \text{ odd}) \Leftrightarrow u_0 = 0,$$

$$(t \text{ odd AND } wt(\bar{r}) \text{ odd}) \text{ OR } (t \text{ even AND } wt(\bar{r}) \text{ even}) \Leftrightarrow u_0 = 1.$$

When $(t+1)$ errors occur, we immediately determine u_0 without estimating the $(k-1)$ low-order bits; the decoding is incomplete.

General Case

The binary cyclic LUEP codes of length $n \leq 39$ are considered, except for the codes having $t = 0$, a protection vector with k identical components, or $k' = 1$ and $t' = t + 1$ (see previous paragraph). In Table I each code is indicated by a reference number, the length n, the dimension k, the number of high-order bits k', the larger and the smaller components s_1 and s_k of the separation vector, the designed distance δ, the error correcting capacity t_δ of the first step of the decoding procedure, the set $\bar{K}$ of zeros of the code, and the set K_{s_1} representing the nonzeros associated with s_1. Furthermore, an optimal array $\mathrm{HT}^*(\tau)$ is described by $\tau(\lfloor(\delta-1)/2\rfloor < \tau \leq t')$, r_1^*, r_2^* (r_2^* is only printed when different from r_1^*), and b^*. The elements $p_i \in \mathrm{HT}^*(\tau) \setminus R$ are also listed; if they do not all belong to distinct cyclotomic cosets modulo n, the p_i that really do appear boldfaced. Finally, the value of μ^* is given.

TABLE I

Number	n	k	k'	s_1	s_k	δ	t_δ	$\bar{K}$	K_{s_1}	τ	r_1^*	r_2^*	b^*	$\{p_i\}$	μ^*
1	15	7	2	5	3	3	1	1,7	5	2	1		13	0	1
2	21	6	3	9	7	7	3	1,5,9	3	4	1		15	0	1
3		7	1	9	8	5	2	3,5,7,9	0	3	1		5	8	6
4		8	2	8	6	6	2	0,1,5	7	3	1		0	3	3
5		9	2	7	3	3	1	1,5	7	2	1		19	0	1
										3	1		1	**3**,6	3
6		10	1	9	4	4	1	3,5,7	0	2	1		12	15	3
										3	1		1	**1**,2,4	6
										4	1		1	**1**,2,4,8	6
7		11	2	6	4	4	1	0,3,5	7	2	1		17	18	3
8		12	3	6	4	4	1	0,5,7	3	2	1		17	18	3
9		13	1	7	4	3	1	3,7,9	0	2	1		6	8	6
										3	1		2	**2**,4,**5**	12
10	27	7	1	9	6	6	2	1,9	0	3	1		4	6	6
										4	1		4	6	6
11	31	16	1	9	6	6	2	5,7,11	0	3	1		9	12	5
										4	3		7	16	5
12	33	12	2	12	6	6	2	0,1,5	11	3	1		0	3	10
										4	1		28	30	10
										5	1		25	**27**,30	10
13		13	1	11	10	5	3	3,5	0	3	5	4	5		0
										4	5		13	0	1
14		13	2	11	3	3	1	1,5	11	2	1		31	0	1
										3	1		1	**3**,6	10
										4	1		1	**3**,6	10
										5	1		1	**3**,6,9	10
15		21	1	11	4	3	1	3,11	0	2	1		9	10	10
										3	1		9	**10**,13,14	10
										4	1		29	**29**,31,32,**0**,1,2	11
										5	1		3	**4**,**5**,7,8,10	20
16		23	12	5	3	3	1	5	1,11	2	5		23	0	1
17	35	7	3	16	14	12	5	0,1,3,15	5	6	1		29	5	3
										7	1		22	28	4
18		8	3	15	7	7	3	1,3,15	5	4	1		29	0	1
										5	1		29	0	1
										6	1		22	28	4
										7	1		21	**21**,28	4
19		11	4	7	5	5	2	1,3	7	3	1		31	0	1
20		13	1	15	8	6	2	3,5,7,15	0	3	1		10	11	12
										4	1		24	29	12
										5	1		12	**16**,18	12
										6	1		12	**16**,18,22,23	12
										7	1		12	**16**,18,22,23	12

TABLE I
(*continued*)

Number	n	k	k'	s_1	s_k	δ	t_δ	$\bar{K}$	K_{s_1}	τ	r_1^*	r_2^*	b^*	$\{p_i\}$	μ^*
21	35	15	3	12	8	6	2	0,3,7,15	5	3	1		12	16	12
										4	1		24	29	12
										5	1		24	**29**,32	12
22		16	1	15	4	4	1	3,5,7	0	2	1		12	15	3
										3	1		1	**1**,2,4	12
										4	1		1	**1**,2,4,8	12
										5	1		1	**1**,2,4,8,9	12
										6	1		1	**1**,2,4,8,9,11	12
										7	1		1	**1**,2,4,8,9,11	12
23		18	3	8	4	4	1	0,3,7	5	2	1		12	15	3
										3	1		31	**32**,1	12
24		19	4	5	4	4	1	3,7	0,5	2	1		12	15	3
25		19	3	8	6	5	2	0,3,15	5	3	1		30	32	12
26		19	4	6	4	4	1	0,3,5	7	2	1		24	25	3
27		20	8	7	6	5	2	3,15	0,5,7	3	1		22	**22**,23	12
28		22	7	6	4	4	1	0,3	5,7	2	1		24	25	3
29		25	1	7	4	3	1	5,7,15	0	2	1		7	**8**,9	12
										3	1		5	**6**,**8**,9	24
30	39	13	1	15	12	7	3	3,7,13	0	4	1		12	16	12
										5	1		26	32	12
										6	1		26	32	12
										7	1		1	**1**,2,4,5,8,10,11	12
31		14	2	14	6	6	2	0,1,7	13	3	1		0	3	12
										4	1		34	36	12
										5	1		31	**33**,36	12
										6	1		0	**3**,6,9	12
32		15	1	13	10	7	3	3,7	0	4	1		27	32	12
										5	1		27	32	12
										6	1		27	32	12
33		15	2	13	3	3	1	1,7	13	2	1		37	0	1
										3	1		1	**3**,6	12
										4	1		1	**3**,6	12
										5	1		1	**3**,6,9	12
										6	1		1	**3**,6,9,12	12
34		25	12	6	3	3	1	7,13	1	2	1		26	27	12
35		25	1	13	4	3	1	3,13	0	2	1		12	14	12
										3	1		8	**8**,10,11	12
										4	1		6	**7**,**8**,10,11	24
										5	1		6	**7**,**8**,10,11,14	24
										6	1		2	**2**,4,5,**7**,8,10,11	24
36		27	14	6	3	3	1	7	1,13	2	1		28	30	12

TABLE II

Number	n	k	d	δ	$\bar{K}$	τ	r_1^*	r_2^*	b^*	$\{p_i\}$	cfr code	μ^*
1	17	9	5	4	1	2	3	2	13			0
2	21	9	8	6	0,1,3,7	3	1		0	5		6
3		7	8	5	1,3,7,9	3	1		1	5		6
4		6	8	6	0,1,3,7,9	3	1		0	5		6
5	23	12	7	5	1	3	1		1	5		11
6		11	8	6	0,1	3	1		0	5		11
7	31	21	5	4	1,5	2	1	8	0	0		1
8		21	5	4	1,7	2	3	7	25	0		1
9		20	6	4	0,1,5	2	1	8	0			0
10		20	6	4	0,1,7	2	3	7	25			0
11		16	7	5	1,5,7	3	1		4	6		5
12		15	8	5	0,1,5,7	3	1		0	3		5
13		11	11	7	1,3,5,11	4	5		1	0		1
						5	1		1	7		5
14		11	11	7	1,3,7,11	4	3		13	0		1
						5	3		1	10		5
15		10	12	10	0,1,3,5,11	5	1		0	7		5
16		10	12	10	0,1,3,7,11	5	3		1	10		5
17	33	13	10	5	1,3	3	1	14	1			0
						4	1		29	0		1
18		11	11	8	1,3,11	4	1		29	0		1
						5	5		24	26		10
19		10	12	10	0,1,3,11	5	1		27	28		10
20	35	16	7	6	1,5,7	3	1		4	6		12

TABLE II
(*continued*)

Number	n	k	d	δ	$\overline{K}$	τ	r_1^*	r_2^*	b^*	$\{p_i\}$	cfr code	μ^*
21	35	15	8	6	0,1,5,7	3	1		0	3		12
22		13	8	6	1,5,7,15	3	1		4	6		12
23		12	8	6	0,1,5,7,15	3	1		0	3		12
24		7	14	12	0,1,3,5	6	1		29	30		3
25	39	26	6	4	0,1	2	1		0	3		12
26		24	6	4	0,1,13	2	1		0	3		12
27		15	10	7	1,3	4	1		1	7		12
28		14	10	8	0,1,3	4	1		0	7		12
29		13	12	7	1,3,13	4	1		1	7		12
						5	1		1	7		12
30		12	12	8	0,1,3,13						29	
31	41	21	9	6	1	3	3		30	30		20
						4	1		1	3,6,7		20
32		20	10	6	0,1	3	3		30	30		20
						4	1		36	38		20
33	43	29	6	4	1	2	3	2	39			0
34		15	13	7	1,3	4	1		37	0		1
						5	1		37	0		1
						6	1		37	0		1
35	45	23	7	6	1,5,21	3	1		30	30		2
36		22	8	6	0,1,5,21						35	
37		16	10	8	0,1,3,7	4	1		4	42		4
38		15	9	8	1,3,7,15	4	1		11	18		4
39		15	10	8	1,7,9,15	4	1		11	12		4
40		14	10	8	0,1,7,9,15						39	
41		14	10	8	0,1,3,7,15						38	
42		12	10	8	0,1,3,7,9	4	1		11	15		2
43		11	9	8	1,3,7,15,21	4	1		41	0		1
44		9	12	9	1,5,7,9,15	5	1		7	12		4
45		8	12	9	0,1,5,7,9,15						44	
46	47	24	11	5	1	3	1		1	5		23
						4	1		1	5		23
						5	1		1	5,10		23
47		23	12	6	0,1	3	1		0	5		23
						4	1		0	5		23
						5	1		0	5		23
48	51	35	5	4	1,9	2	5	8	8			0
49		34	6	4	0,1,9						48	
50		34	6	4	0,1,5	2	1		0	3		8
51		32	6	4	0,1,5,17	2	1		0	3		8
52		27	5	4	1,9,19						48	
53		27	8	5	1,3,9	3	1		1	5		8
54		27	9	5	1,5,9	3	1		13	17		2
						4	1		1	3,6		8
55		27	9	5	1,3,19	3	1	23	1			0
						4	1		47	0		1
56		26	8	6	0,1,3,9	3	1		0	5		8
57		26	10	6	0,1,5,9						54	
58		25	8	5	1,3,9,17						53	
59		25	10	7	1,5,9,17	4	1		13	19		8
60		25	10	5	1,3,17,19	3	1	23	1			0
						4	1		47	0		1
61		24	8	6	0,1,3,9,17						53	
62		24	10	7	0,1,5,9,17						59	
63		19	10	6	1,3,9,19						55	
64		19	14	11	1,3,5,9	6	1		1	11		8
65		18	14	12	0,1,3,5,9	6	1		0	11		8
66		17	12	6	1,3,9,17,19	3	1	8	16			0
						4	1		47	0		1
						5	1		1	5,7,10		8
67		17	14	9	1,3,5,17,19	5	1		47	0		1
						6	1		47	0		1
68		17	14	10	1,5,9,17,19	5	1		12	12		8
						6	5		5	45		8
69		17	16	11	1,3,5,9,17	6	1		1	11		8
						7	1		1	11		8
70		16	12	10	0,1,3,9,17,19	5	1		45	46		8
71		16	14	10	0,1,5,9,17,19						68	
72		16	16	12	0,1,3,5,9,17						69	
73		11	15	9	1,3,5,11,19	5	1		43	0		1
						6	1		43	0		1
						7	1		43	0		1

TABLE II
(continued)

Number	n	k	d	δ	$\bar{K}$	τ	r_1^*	r_2^*	b^*	$\{p_i\}$	cfr code	μ^*
74	51	11	15	9	1,5,9,11,19	5	5		11	0		1
						6	5		11	0		1
						7	5		11	0		1
75		9	15	12	1,3,5,11,17,19						73	
76		9	15	12	1,5,9,11,17,19						74	
77		8	24	20	0,1,3,5,9,17,19	10	1		0	11		8
						11	1		0	11		8
78	55	35	5	4	1	2	1	9	7			0
79		34	8	4	0,1	2	1	7	0			0
						3	1		13	15		10
80		30	10	5	0,1,11	3	1		13	15		10
						4	1		51	**51**,53,54,3		20
81		25	11	7	1,5	4	1		13	19		20
						5	1		1	**3**,6		20
82		24	12	7	0,1,5						81	
83		21	15	8	1,5,11	4	1		4	6		20
						5	1		7	12		20
						6	1		7	12		20
						7	1		4	**6**,12		20
84		20	16	8	0,1,5,11						83	
85	57	21	14	6	1,3	3	1	5	48			0
						4	1		53	0		1
						5	1		1	**5**,10		18
						6	1		1	**5**,10,11		18
86		20	14	10	0,1,3	5	1		0	5		18
						6	1		48	52		18
87		19	16	9	1,3,19	5	5		18	23		18
						6	1		1	**5**,10,11		18
						7	1		1	**5**,10,11,13		18
88		18	16	10	0,1,3,19	5	5		18	23		18
						6	1		48	52		18
						7	1		48	52		18

VI. Application to Conventional Cyclic Codes

In Table II we list the binary cyclic codes of length $n \leq 57$, having $\lfloor(\delta-1)/2\rfloor < \lfloor(d-1)/2\rfloor$. A complete list of binary cyclic codes of length $n \leq 57$ and $\delta < d$ is given in [6]. In Table II, each code is indicated by a reference number, the length n, the dimension k, the minimum distance d, the designed distance δ, and the set $\bar{K}$ of zeros.

Furthermore, we describe an optimal array $\mathrm{HT}^*(\tau)$ by $\tau(\lfloor(\delta-1)/2\rfloor < \tau \leq \lfloor(d-1)/2\rfloor)$ r_1^*, r_2^* (r_2^* is only printed when different from r_1^*), and b^*. We list elements $p_i \in \mathrm{HT}^*(\tau)\setminus R$ and we give the value of μ^*.

VII. Conclusion and Remarks

When $\mu^* = 0$ appears in the tables, the corresponding value of τ is such that $\lfloor(\delta_{\mathrm{BCH}}-1)/2\rfloor < \tau \leq t_\delta$. An error pattern of weight τ can be corrected in the first step of the decoding procedure by means of $\mathrm{HT}(\tau) \subset R$. Consequently, the set $\{p_i\}$ is empty.

It often happens that $\mu^*(\tau_1) \ll \mu^*(\tau_2)$, where $\tau_1 < \tau_2$. We can apply the extended BCH algorithm to correct τ_1 or fewer errors, but the full error-correcting capacity is out of reach.

Some codes have $\mu^* = 1$, S_0 being the unknown power sum. The corresponding number of errors τ can be corrected by applying the BCH algorithm twice. Actually, once is sufficient as we do in fact know the value of S_0. Indeed, when τ is even, $S_0 = 0$; when τ is odd, $S_0 = 1$. These codes are certainly of practical interest.

Up to which value of μ^* a code may be called practical depends on the number of achievable parallel computations. We conjecture that the future is in favor of our results. We finish by giving an illustrative example.

Example: Consider the $(31,11)_{d=11,\,\delta=7}$ code (13, Table II). Three or fewer errors can be corrected in the first step of the decoding procedure.

To correct four errors, we use $\mathrm{HT}^*_{(4)}$ ($r_1^* = r_2^* = 5$, $b^* = 1$). According to the modified algorithm described in Section II, the Newton identities are

$$S_{21} + S_{16}\sigma_1 + S_{11}\sigma_2 + S_6\sigma_3 + S_1\sigma_4 = 0$$

$$S_{26} + S_{21}\sigma_1 + S_{16}\sigma_2 + S_{11}\sigma_3 + S_6\sigma_4 = 0$$

$$S_0 + S_{26}\sigma_1 + S_{21}\sigma_2 + S_{16}\sigma_3 + S_{11}\sigma_4 = 0$$

$$S_5 + S_0\sigma_1 + S_{26}\sigma_2 + S_{21}\sigma_3 + S_{16}\sigma_4 = 0.$$

S_0 is unknown, but we conclude that it must be 0, as we are looking for an even number of errors.

Suppose that the all-zero codeword was sent and that $r(X) = 1 + X^3 + X^4 + X^{23}$. With the aid of a table of

$GF(2^5)$, which can be found in [1], we compute $S_1 = \alpha^{28}$, $S_6 = 1$, $S_{11} = \alpha^{13}$, $S_{16} = \alpha^{14}$, $S_{21} = \alpha^{22}$, $S_{26} = \alpha^{11}$, $S_5 = \alpha^{11}$.

Solving the Newton identities yields the error-locator polynomial $\sigma(z) = 1 + \alpha^{11}z + \alpha^{16}z^2 + \alpha^6 z^3 + \alpha^{26}z^4$, having zeros $1, \alpha^9, \alpha^{11}, \alpha^{16}$ with inverses $1, \alpha^{22}, \alpha^{20}, \alpha^{15}$. As $r_1 = 5$ and $5u + 31v = 1$ for $u = -6$, $v = 1$, we have to raise these results to (-6), thus obtaining the error locators 1, $\alpha^{23}, \alpha^4, \alpha^3$.

To correct five errors, we use $HT^*_{(5)}$ $(r_1^* = r_2^* = 1,\ b^* = 1)$. The Newton identities are $S_{5+j} + \sigma_1 \cdot S_{4+j} + \sigma_2 \cdot S_{3+j} + \cdots + \sigma_5 \cdot S_j = 0$, $1 \le j \le 5$. Here S_7 is unknown; we have to test all the field elements of $GF(2^5)$, as $\mu = m = 5$.

Suppose $r(X) = 1 + X + X^2 + X^9 + X^{25}$. We compute $S_1, S_2, \cdots, S_{10} = \alpha^2, \alpha^4, \alpha^7, \alpha^8, \alpha^6, \alpha^{14}, S_7, \alpha^{16}, \alpha^{17}, \alpha^{12}$. Setting $S_7 = 0$, we calculate $\sigma(z) = 1 + \alpha^2 z + \alpha^{23}z^2 + \alpha^{11}z^3 + \alpha^4 z^4 + \alpha^{13}z^5 = (\alpha^2 z + 1)$. $\sigma'(z)$, where $\sigma'(z) = 1 + \alpha^{23}z^2 + \alpha^{24}z^3 + \alpha^{11}z^4$ has no solutions in $GF(2^5)$; thus a failure of type 3 has occurred (see Section II).

Only when we set $S_7 = \alpha^{17}$, do we obtain $\sigma(z) = 1 + \alpha^2 z + \alpha^{23}z^2 + \alpha^{11}z^3 + \alpha^{11}z^4 + \alpha^6 z^5$, having zeros $1, \alpha^6, \alpha^{22}, \alpha^{29}, \alpha^{30}$. The inverses $1, \alpha^{25}, \alpha^9, \alpha^2, \alpha$ are the locators of the error pattern $e(X) = 1 + X + X^2 + X^9 + X^{25}$.

Acknowledgment

I wish to thank the whole team of Discrete Mathematics of the Eindhoven University of Technology for their encouragement during my three months' stay with them. Special thanks are given to Mr. H. J. Tiersma and Mr. F. C. Bussemaker for their practical assistance and to Mr. W. J. Van Gils of Philips Research Laboratories for his interest and comments on this paper.

References

[1] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes.* Amsterdam, The Netherlands: North-Holland, 1977.

[2] W. W. Peterson and E. J. Weldon, *Error Correcting Codes.* Cambridge, MA: MIT press, 1972.

[3] E. R. Berlekamp, *Algebraic Coding Theory.* New York: McGraw-Hill, 1968.

[4] W. J. van Gils, "Two topics on linear unequal error protection codes: Bounds on their length and cyclic code classes," *IEEE Trans. Inform. Theory*, vol. IT-29, no. 6, Nov. 1983.

[5] P. Stevens, "About the transformation of information word into codeword, and vice versa, for binary cyclic LUEP codes," Free Univ. of Brussels, Internal Rep., Dec. 1986.

[6] J. H. van Lint and R. M. Wilson, "On the minimum distance of cyclic codes," *IEEE Trans. Inform. Theory*, vol. IT-32, Jan. 1986.