

**Theorem 7:** Let  $h$  and  $g$  be Boolean functions on, respectively,  $t$  and  $n-t$  variables and let

$$f(x_1, x_2, \dots, x_n) = g(x_1, x_2, \dots, x_{n-t}) + \sum_{i \leq t, j > t} x_i x_j + h(x_{n-t+1}, \dots, x_n).$$

If  $X(h)$  is a  $H(2, r, t)$ , then  $X(f)$  is an  $H(2, n-t+r, n)$ . Moreover, if  $X(g)$  is a  $H(2, s, n-t)$  and  $u = \max(t+s, n-t+r)$ , then  $X(f)$  is an  $H(2, u, n)$ .

**Corollary 8:** Let  $n \geq r \geq 2$ ; then

$$|C(2, r, n)| \geq \sum_{i=n-r+2}^n \sum_{i=0}^{n-i} {}^n C_i (-1)^i \binom{n-i}{i} 2^{t+i+2^{n-t-i}}.$$

*Proof:* For all  $i = 1, 2, \dots, n$ , let  $h_i(x) = x_i \sum_{j=1}^n x_j$ . We will say  $f$  has attribute  $i$  if  $f(x) = h_i(x) + g(x) + ax_i$ , where  $a = 0$  or  $1$ , and  $g(x)$  is independent of  $x_i$ . Now let  $N = \{1, 2, \dots, n\}$ , and, for all  $T \subset N$ , let  $S(T)$  denote the set of  $H(2, n, n)$  of the form  $X(f)$  where, for all  $i \in T$ ,  $f$  has property  $i$ . The typical element of  $S(T)$  is  $X(f)$  where

$$f(x) = g(x) + \sum_{i \in T, j \in N-T} x_i x_j + \sum_{i \in T} a_i x_i + \sum_{i < j, i, j \in T} x_i x_j,$$

where  $g(x)$  is independent of  $x_i$  for all  $i \in T$ . Hence, if  $t = |T|$ ,  $|S(T)| = 2^t 2^{2^{n-t}}$ . By Theorem 7, every element of  $S(T)$  is an  $H(2, n-t+2, n)$ . Now let  $E(T)$  be the set of matrices of the form  $X(f)$  where  $f$  has property  $i$  precisely when  $i \in T$ ; then, by the Principle of Inclusion and Exclusion,

$$|E(T)| = \sum_{i=0}^{n-t} (-1)^i \binom{n-t}{i} 2^{t+i+2^{n-t-i}}. \quad \square$$

**V.  $C(2^t, r, n)$ : SOME SUBCODES OF  $C(2, n(t-1)+r, n)$**

Consider a Boolean function  $f(x)$  where  $m = tn$ . We may construct an  $n$ -dimensional matrix of order  $2^t$  by using  $x_{it+1}, x_{it+2}, \dots, x_{(i+1)t}$  to index the  $(i+1)$ th dimension. We let  ${}^t X(f)$  denote this matrix. (So  $X(f) = {}^1 X(f)$ .)

**Theorem 9:** Let  $s, t \geq 1$  and  $n \geq r \geq 2$  be integers. If  ${}^s X(f)$  is an  $H(2^s, r, n)$ , then  ${}^s X(f)$  is an  $H(2^s, n(t-1)+r, nt)$ . Hence, for  $t \geq 2$ ,  $C(2^t, r, n)$  is a subcode of  $C(2, n(t-1)+r, nt)$  and, hence, of  $R(n(t-1)+r-1, nt)$ .

*Proof:* Let  $T$  be an  $r$ -subset of  $\{1, 2, \dots, n\}$ , and let  $j \in T$ . Suppose that, for all  $i = 1, 2, \dots, j-1, j+1, \dots, n$ ,  $y_i = x_i$  and that  $y_j \neq x_j$ ; then

$$w \left( \sum_{i \in T - \{j\}} \sum_{x_i \in V^{st}} f(x_1, \dots, x_n) + f(y_1, \dots, y_n) \right) = 2^{st(r-1)-1}.$$

Since any  $n(t-1)+r$  dimensions of  ${}^s X(f)$  must contain all the components of some  $r$  dimensions of  ${}^{st} X(f)$ , any  $(n(t-1)+r)$ -section of  ${}^s X(f)$  must be Hadamard. The final remark follows from Theorem 3.  $\square$

**VI. CONCLUSION**

Hammer and Seberry [4] and Shlichhta [5] suggest that higher dimensional Hadamard matrices may have applications as error-correcting codes. This seems to be particularly probable when errors occur in bursts (which could be assumed to be confined to sections of certain dimensions). This correspondence establishes lower bounds to the minimum distance of

some codes obtained from families of higher dimensional matrices, and shows how to correct certain numbers of errors quickly even when the errors are homogeneously spread throughout the codeword. It would be interesting to see whether there are any fast algorithms for correcting larger numbers of errors which are assumed to occur in bursts. It is not certain that orthogonality properties are needed in all directions.

This correspondence and the work of Yi Xian [1]–[3] demonstrates a strong connection between the Boolean functions and higher dimensional designs; any Boolean function can be thought to have a spectrum of orthogonality properties depending on how its truth table is used to fill the entries of a higher dimensional design. Suggesting that higher dimensional Hadamard matrices may be of use in certain circumstances is an instance of proposing that sets of Boolean functions with certain prescribed orthogonality properties may have certain desirable properties.

**ACKNOWLEDGMENT**

The author would like to thank Kathy J. Horadam and especially Alan Blenkin (AZB) for their help and interest during the preparation of this correspondence. The author would also like to thank a referee for a careful reading of the original manuscript.

**REFERENCES**

- [1] Yang Yi Xian, "The proofs of some conjectures on higher dimensional Hadamard matrices," *Kexue Tongbao*, vol. 21, no. 24, pp. 1662–1667, 1986.
- [2] —, "On the classification of four-dimensional 2nd-order Hadamard matrices," preprint, 1986.
- [3] —, "On  $n$ -dimensional 2nd-order Hadamard matrices," preprint, 1986.
- [4] J. Hammer and J. Seberry, "Higher dimensional orthogonal designs and applications," *IEEE Trans. Inform. Theory*, vol. IT-27, no. 6, pp. 772–779, Nov. 1981.
- [5] P. J. Shlichhta, "Higher dimensional Hadamard matrices," *IEEE Trans. Inform. Theory*, vol. IT-25, no. 5, pp. 566–572, Sept. 1979.

**More Efficient Soft Decoding of the Golay Codes**

Alexander Vardy and Yair Be'ery

**Abstract**—An algorithm for maximum-likelihood soft-decision decoding of the binary (24, 12, 8) Golay code is presented. The algorithm involves projecting the codewords of the binary Golay code onto the codewords of the (6, 3, 4) code over  $GF(4)$ —the hexacode. The complexity of the proposed algorithm is at most 651 real operations that is, to the best of our knowledge, less than the complexity of any algorithm ever published. Along similar lines the tetracode may be employed for decoding the ternary (12, 6, 6) Golay code with only 530 real operations. The proposed algorithm also implies a reduction in the number of computations required for decoding the Leech lattice.

**Index Terms**—Golay codes, soft-decision decoding, hexacode, Leech lattice.

Manuscript received December 5, 1989; revised June 5, 1990. The authors are with the Department of Electrical Engineering-Systems, Tel-Aviv University, Tel-Aviv 69978, Israel. IEEE Log Number 9142958.

## I. INTRODUCTION

The (24, 12, 8) binary extended Golay code  $C$  is certainly one of the most interesting codes known. It is an extended perfect extremal double-even self-dual code. Codewords of  $C$  of weight eight, called *octads*, hold the Steiner system  $S(5, 8, 24)$ . The Golay code may be used to construct (via construction  $B$  of [8, p. 142]) the Leech lattice,—an extremely dense 24-dimensional sphere packing, which has been recently utilized for implementation of block-coded modulation techniques for band-limited channels [5], [10]. The binary Golay code itself has been successfully employed in several existing communication systems, e.g., satellite control channels. Thus the availability of an efficient Golay decoder, especially a soft-decision decoder, apparently has some practical importance. In fact such a decoder was even implemented in a special purpose VLSI hardware in [1].

For all these reasons the problem of maximum-likelihood soft-decision decoding of the binary Golay code was intensively investigated in the last few years. In 1986 Conway and Sloane [7] published a decoding algorithm which requires 1614 operations and in the same year Be'ery and Snyders [3] have proposed an algorithm with a worst case complexity of 1551 operations. The decoding algorithm of Forney [10] requires only 1351 operations, while the recent algorithm of Be'ery and Snyders [4], [18] requires 827 operations at most. In this correspondence we claim a more efficient algorithm with a worst case complexity of 651 operations.

To be precise we note that the complexity of decoding is measured herein in terms of the total number of real additions and comparisons. In compliance with the convention of [2]–[4], [7], [10]–[12], [18] such operations as memory addressing, negation, taking the absolute value, and multiplication by 2 are neglected though none of these is allowed to be excessive. An effort has been made to evaluate all the algorithms in a uniform manner. The complexity figures just cited follow those of [18]. As Conway and Sloane [7] say: Use these figures for comparison only.

In [9] Curtis proposed the miracle octad generator (MOG) as an efficient means for locating the octad of  $S(5, 8, 24)$  which contains 5 given points. Conway [6] has developed this idea further by introducing the *hexacode*. Conway and Sloane [8] and Pless [15] have shown how the hexacode may be used to enable hard-decision decoding of the binary Golay code by hand. In the sequel we shall employ the hexacode for the soft-decision decoding of the binary Golay code. In Section II we briefly outline the relation between the hexacode and the binary Golay code. For a detailed treatment see [8, Chapter 11]. The decoding algorithm is presented in Section III. More efficient decoding of the ternary Golay code and of the Leech lattice is discussed in Section IV.

## II. PRELIMINARIES

The hexacode  $H$  is the unique, up to a monomial permutation of coordinates, (6, 3, 4) linear code over  $GF(4)$ . Recall that the elements of  $GF(4)$ ,  $-0, 1, \omega, \bar{\omega}$ , hereafter called *characters*, satisfy  $\bar{\omega} = \omega^2$ ,  $\bar{\omega}^2 = \omega$ , and  $\bar{\omega} = 1 + \omega$ . Following [15] we take

$$\begin{bmatrix} 1 & 0 & 0 & 1 & \bar{\omega} & \omega \\ 0 & 1 & 0 & 1 & \omega & \bar{\omega} \\ 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

as a generator matrix of  $H$ . The relation between  $H$  and  $C$  is as follows. We shall represent binary vectors of length 24 by  $4 \times 6$  arrays with entries from  $GF(2)$ . A row or a column of such array is called *odd* or *even* according as it contains an odd or even

number of nonzeros. Let now  $u$  be the 4-tuple  $(0, 1, \omega, \bar{\omega})$  over  $GF(4)$ . Any column 4-tuple  $a = (a_1, a_2, a_3, a_4)^t$  over  $GF(2)$  satisfying  $u \cdot a = \alpha$ , where  $\alpha \in GF(4)$ , is said to be then *interpretation* of the character  $\alpha$ . Conversely,  $\alpha$  is said to be the *projection* of  $a$ . Obviously, any  $\alpha \in GF(4)$  has exactly two odd and two even interpretations. The 16 possible interpretations of the four characters are as follows:

$$\begin{array}{cccccc} 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ \hline 0 & 1 & \omega & \bar{\omega} & & & 0 & 1 & \omega & \bar{\omega} & & & \\ \text{odd interpretations} & & & & & & \text{even interpretations.} & & & & & & \end{array}$$

Thus by taking the projection of each of the six columns of the  $4 \times 6$  array we may project binary vectors of length 24 onto quaternary vectors of length 6.

The equivalence of the following definition to other definitions of the (24, 12, 8) Golay code is proved in detail in [15]. The main idea of the proof is to show that the parameters of  $C$ , as defined below, are indeed (24, 12, 8) and then employ the uniqueness of the binary Golay code [14].

*Definition:* The code  $C$  is the set of all the  $4 \times 6$  arrays with elements from  $GF(2)$ , which satisfy the following conditions.

- The parity of all the columns is the same, i.e., all the columns are either even or odd.
- The parity of the columns equals the parity of the top row.
- The projection is in the hexacode.

For instance the following arrays are codewords of  $C$ :

$$\begin{array}{c} 0 \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ \omega & 1 & 1 & 0 & 0 & 0 \\ \bar{\omega} & 1 & 1 & 0 & 0 & 1 \end{bmatrix} \\ 0 \quad 1 \quad 0 \quad 1 \quad \omega \quad \bar{\omega} \\ 0 \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ \omega & 1 & 1 & 1 & 0 & 1 \\ \bar{\omega} & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \\ \bar{\omega} \quad \omega \quad 1 \quad 0 \quad 0 \quad 1 \end{array} \quad \begin{array}{c} 0 \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ \omega & 0 & 1 & 0 & 0 & 1 \\ \bar{\omega} & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \\ 1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \end{array}$$

## III. THE DECODING ALGORITHM

Assume that a codeword of  $C$  is transmitted through a binary input channel with output alphabet  $\mathbb{R}$ , characterized by the transition probability densities (or in case of a discrete channel, transition probabilities)  $f_j(v) = f(v/j)$ , where  $j \in GF(2)$  and  $v \in \mathbb{R}$ . Let the vector  $v = (v_1, v_2, \dots, v_{24})$  be observed at the output. Maximum likelihood soft decision decoding consists of finding a codeword  $c = (c_1, c_2, \dots, c_{24}) \in C$ , which maximizes  $P(v/c)$ , that is, maximizes the *a posteriori* probability  $P(c/v)$ , provided that  $P(c) = 2^{-12}$  for all codewords of  $C$ . As shown in [3] on a memoryless channel, one may as well search a codeword that maximizes the *metric*  $M(c)$  given by

$$M(c) = \sum_{i=1}^{24} (-1)^{c_i} \mu_i, \quad (1)$$

where  $\mu_i = \log[f_0(v_i)/f_1(v_i)]$  is the *confidence value* of the  $i$ th bit. Alternatively, the maximization of (1) may be regarded as finding a codeword  $c \in C$  that maximizes the inner product  $c \cdot \mu$ , where  $\mu = (\mu_1, \mu_2, \dots, \mu_{24})$  and the components of  $c$  are represented in the  $\pm 1$  notation of [7]. In the special case of an

additive white Gaussian noise (AWGN) channel this is equivalent to minimum Euclidean distance decoding and one may set  $\mu_i = v_i$  (cf. [3]).

The 24 real values  $\mu_1, \mu_2, \dots, \mu_{24}$  are the input to our decoder. The output of the decoder is the codeword  $\hat{c} \in C$  that maximizes (1). The decoding algorithm will be described in five steps.

1) *Computing the Confidence Values of the Characters:* For each of the six coordinates of  $H$ , i.e.,  $j = 1, \dots, 6$  and for each character  $x$  we shall compute the *confidence value of the even interpretation*  $\mu_j^e(x)$  and the *confidence value of the odd interpretation*  $\mu_j^o(x)$ . The confidence values  $\mu_j^e(x)$  and  $\mu_j^o(x)$  are defined as follows:

$$\begin{aligned} \mu_1^o(0) &= |\mu_1 - \mu_2 - \mu_3 - \mu_4| & \mu_1^e(0) &= |\mu_1 + \mu_2 + \mu_3 + \mu_4| \\ \mu_1^o(1) &= |\mu_1 - \mu_2 + \mu_3 + \mu_4| & \mu_1^e(1) &= |\mu_1 + \mu_2 - \mu_3 - \mu_4| \\ \mu_1^o(\omega) &= |\mu_1 + \mu_2 - \mu_3 + \mu_4| & \mu_1^e(\omega) &= |\mu_1 - \mu_2 + \mu_3 - \mu_4| \\ \mu_1^o(\bar{\omega}) &= |\mu_1 + \mu_2 + \mu_3 - \mu_4| & \mu_1^e(\bar{\omega}) &= |\mu_1 - \mu_2 - \mu_3 + \mu_4| \end{aligned}$$

and the confidence values of the remaining 5 coordinates are defined similarly. Note that interpretations of the same character that have the same parity are complements of each other. For instance if  $\mu_1 + \mu_2 + \mu_3 + \mu_4 \geq 0$  then the confidence value of the interpretation  $(0000)^j$  for the character 0 is  $\mu_1^e(0)$ , and the confidence value of the interpretation  $(1111)^j$  is  $-\mu_1^o(0)$ . Thus the sign of  $\mu_1 + \mu_2 + \mu_3 + \mu_4$  determines which of the two even interpretations of the character 0 is the *preferable interpretation*.

*Complexity:* Using the appropriate Gray code [16], such as

$$\begin{aligned} 1. & + & + & + & + \\ 2. & - & + & + & + \\ 3. & - & - & + & + \\ 4. & + & - & + & + \\ 5. & + & - & - & + \\ 6. & - & - & - & + \\ 7. & - & + & - & + \\ 8. & + & + & - & + \end{aligned} \tag{2}$$

This step requires 10 real additions for each coordinate and altogether 60 real additions. It should be pointed out that the difference between two adjacent rows in (2) is equal to  $\pm 2\mu_i$ , for some  $i$ . Thus the computation involves either a multiplication by 2 of each  $\mu_i$  or a multiplication of the first outcome by 0.5. However, in conformity with the convention established in the earlier works of Conway and Sloane [7], Forney [10], and Be'ery and Synders [4], [18] (which also use exactly the same form of Gray code) these operations, as well as the operation of taking the absolute value at each row of (2), are neglected.

2) *Sorting the Confidence Values of the Characters:* For each of the six coordinates  $j = 1, \dots, 6$  we shall sort the confidence values of even and odd interpretations. For example, if  $\mu_1 = 0.32$ ,  $\mu_2 = -0.25$ ,  $\mu_3 = 0.67$ , and  $\mu_4 = 0.11$ , we will create the following two ordered lists for the first coordinate,

$$\begin{aligned} \mu_1^e(\omega) &\geq \mu_1^e(0) \geq \mu_1^e(1) \geq \mu_1^e(\bar{\omega}); \\ \mu_1^o(1) &\geq \mu_1^o(\bar{\omega}) \geq \mu_1^o(\omega) \geq \mu_1^o(0). \end{aligned}$$

The confidence values of the other 5 coordinates are sorted similarly.

*Complexity:* The difference between the confidence values of any two characters may be expressed in the form  $\pm 2\mu_{i_1} \pm 2\mu_{i_2}$ , where  $i_1, i_2$  belong to  $\{1, 2, 3, 4\}$  for the first coordinate and to the corresponding 4-set for the other coordinates. Thus all we have to do for the first coordinate, for instance, is to sort the

absolute values  $|\mu_1|, |\mu_2|, |\mu_3|, |\mu_4|$ . It is well known that the number of comparisons required to sort four real values is 5 [13]. However, if the Gray code at step 1 is chosen appropriately this may be achieved with only three comparisons. Note that the Gray code of (2) is cyclic, i.e., we can start the computation at any row. In the previous example, if we start either at row 6 or at row 8, we can conclude that  $|\mu_1| > |\mu_2|$  and  $|\mu_3| > |\mu_4|$ . In general, in order to save these two comparisons, it is sufficient to require that the Gray code at step 1 would be cyclic; for instance (2) is always appropriate. The only decision that the decoder has to make according to the received data is at which row to start the computation, and this decision may be arrived at with the help of logic operations on the signs of  $\mu_1, \mu_2, \mu_3, \mu_4$ . It therefore follows that the complexity of the second step is at most 3 operations for each coordinate and altogether at most 18 operations (at least 12).

3) *Computing the Confidence Values of the Blocks:* If  $x = (x_1, x_2, x_3, x_4, x_5, x_6)$  is any vector in  $GF(4)^6$  the sets  $\{x_1, x_2\}$ ,  $\{x_3, x_4\}$ , and  $\{x_5, x_6\}$  are said to be the *blocks* of  $x$ . For each of the 4096 vectors of  $GF(4)^6$  and for each of the three blocks we compute the *confidence value of the sum* in even interpretation  $S_j^e(x_1, x_2)$  and in odd interpretation  $S_j^o(x_1, x_2)$ , as well as the *confidence value of the difference* in even interpretation  $D_j^e(x_1, x_2)$  and in odd interpretation  $D_j^o(x_1, x_2)$ , where  $j = 1, 2, 3$ . For the first block the confidence values of the sum and the difference are defined by

$$\begin{aligned} S_1^e(x_1, x_2) &= \mu_1^e(x_1) + \mu_2^e(x_2) \\ S_1^o(x_2, x_2) &= \mu_1^o(x_1) + \mu_2^o(x_2) \\ D_1^e(x_1, x_2) &= |\mu_1^e(x_1) - \mu_2^e(x_2)| \\ D_1^o(x_1, x_2) &= |\mu_1^o(x_1) - \mu_2^o(x_2)| \end{aligned} \tag{3}$$

while for the other two blocks the corresponding confidence values are defined similarly.

*Complexity:* Computing each of the confidence values defined in (3) requires 16 operations according to the 16 possibilities of choosing  $\{x_1, x_2\}$ . Therefore the complexity of this step is 64 operations for each block and altogether 192 operations.

The next two steps require some explanation. Consider for instance the following hexacodeword  $(0101\omega\bar{\omega})$ . We shall construct a  $4 \times 6$  array with entries from  $GF(2)$  by taking the preferable even interpretation for each of the six characters of  $(0101\omega\bar{\omega})$ . Now, if the top row of this array has even parity, condition b) is satisfied and the array corresponds to a codeword  $c \in C$  with a metric

$$M(c) = S_1^e(0, 1) + S_2^e(0, 1) + S_3^e(\omega, \bar{\omega}).$$

However if the top row of the array is odd, condition b) is violated and to obtain a codeword  $c \in C$  we must *complement* one of the characters, i.e., instead of the preferable even interpretation for this character we have to take its complement. Indeed, to achieve the highest possible metric, we should complement the least reliable character, namely the one with the lowest confidence value (this is exactly the Wagner rule of [17]). For instance if  $\mu_5^e(\omega) \leq \mu_2^e(1) \leq \dots \leq \mu_3^e(0)$ , the character in the fifth coordinate should be complemented and the metric  $M(c)$  is given by

$$M(c) = S_1^e(0, 1) + S_2^e(0, 1) + D_3^e(\omega, \bar{\omega}).$$

The next step enables us to determine which of the six characters is to be complemented in each codeword of  $H$  if condition b) is violated.

4) *Merging the Confidence Values of the Characters:* In each of the 64 hexacodewords we shall find the least reliable character in even interpretation and in odd interpretation, by means of the appropriate sorting of

$$\{\mu_1^e(0), \mu_1^e(1), \mu_1^e(\omega), \mu_1^e(\bar{\omega}); \mu_2^e(0), \mu_2^e(1), \mu_2^e(\omega), \mu_2^e(\bar{\omega}); \dots; \mu_6^e(0), \mu_6^e(1), \mu_6^e(\omega), \mu_6^e(\bar{\omega})\} \quad (4a)$$

$$\{\mu_1^o(0), \mu_1^o(1), \mu_1^o(\omega), \mu_1^o(\bar{\omega}); \mu_2^o(0), \mu_2^o(1), \mu_2^o(\omega), \mu_2^o(\bar{\omega}); \dots; \mu_6^o(0), \mu_6^o(1), \mu_6^o(\omega), \mu_6^o(\bar{\omega})\}. \quad (4b)$$

*Complexity:* From the second step we have the sorting of the confidence values in each coordinate. Furthermore, computation of  $D_j^e(x_1, x_2)$  at the third step is equivalent to comparison between  $\mu_1^e(x_1)$  and  $\mu_2^e(x_2)$ . Hence after the third step we have the sorting of the eight confidence values in each block. Evidently the character that has the highest confidence value in its block will never be complemented. Now let  $a, b, c$  be the characters that have the second large confidence value in the three blocks, and assume that the confidence value of  $c$  is lower than that of  $a$  and  $b$ . Then the characters  $a$  and  $b$  will never be complemented as well. Thus the complexity of the required sort of (4a) is two comparisons to determine the minimum among the confidence values of  $a, b, c$  plus another  $(6+6-1)+(12+7-1)=29$  comparisons required for merging of the  $6+6+7$  presorted confidence values of the characters in the three blocks, using the technique of two-way merge [13]. It should be pointed out that the foregoing procedure is not necessarily optimal. However, it allows us to provide an upper bound on the complexity of the fourth step:  $2(2+29)=62$  real comparisons.

5) *Maximizing the Metric with Respect to the Hexacode:* For each of the 64 hexacodewords  $x=(x_1, x_2, x_3, x_4, x_5, x_6) \in H$ , we shall compute the metric of  $x$  in even interpretation and in odd interpretation, according to

$$M^e(x) = S_1^e(x_1, x_2) + S_2^e(x_3, x_4) + S_3^e(x_5, x_6),$$

$$M^o(x) = S_1^o(x_1, x_2) + S_2^o(x_3, x_4) + S_3^o(x_5, x_6),$$

provided that the preferable interpretations of the characters of  $x$  satisfy condition b). If condition b) is violated and the least reliable character of  $x$  belongs to the  $j$ th block, we shall replace  $S_j^e(\cdot, \cdot)$  by  $D_j^e(\cdot, \cdot)$  in the computation of  $M^e(x)$  and/or  $S_j^o(\cdot, \cdot)$  by  $D_j^o(\cdot, \cdot)$  in the computation of  $M^o(x)$ . Among all the codewords of  $H$  we choose the codeword  $\hat{x}$  that has the highest metric in either interpretation, and decode to the codeword  $\hat{c} \in C$  whose projection is  $\hat{x}$ , using the preferable interpretation for all but possibly one of the six characters.

*Complexity:* In a straightforward manner this step requires  $2 \cdot 64 \cdot 2 = 256$  additions and 127 comparisons. There exists, however, a more efficient technique which will enable us to save at least 64 operations. We partition  $H$  into 16 disjoint sets of four codewords each, so that all the codewords in the same set share a common block. For instance consider the following partition:

$$\left\{ \begin{array}{cccc} 00 & 00 & 00 & 0 \\ 00 & 11 & 11 & 1 \\ 00 & \omega & \omega & \omega \\ 00 & \bar{\omega} & \bar{\omega} & \bar{\omega} \end{array} \right\}, \quad \left\{ \begin{array}{cccc} 01 & 01 & \omega & \bar{\omega} \\ 01 & 10 & \bar{\omega} & \omega \\ 01 & \omega & \bar{\omega} & 01 \\ 01 & \bar{\omega} & \omega & 10 \end{array} \right\},$$

$$\dots, \quad \left\{ \begin{array}{cccc} \bar{\omega} & \bar{\omega} & 00 & \bar{\omega} \\ \bar{\omega} & \bar{\omega} & 11 & \omega \\ \bar{\omega} & \bar{\omega} & \omega & \omega \\ \bar{\omega} & \bar{\omega} & \bar{\omega} & \bar{\omega} \end{array} \right\}.$$

Finding the codeword with the highest metric in each of the sets above requires at most 9 operations (at least 8 operations) for

each interpretation. Hence the complexity of the fifth step is at most  $2 \cdot 16 \cdot 9 + 31 = 319$  real operations.

Thus the total number of real operations required in our algorithm is  $60 + 18 + 192 + 62 + 319 = 651$  in the worst case, as compared to the best decoding algorithm presently known [18] which requires 827 operations. Note that the five steps are essentially independent and therefore enable pipelining. Furthermore, each of the five steps offers an appreciable amount of inherent concurrence. These two facts may be employed for an efficient hardware implementation of the proposed decoder. A block diagram of such decoder is presented in Fig. 1.

#### IV. FURTHER RESULTS

The *tetracode* is the unique, up to monomial equivalence,  $(4, 2, 3)$  code over  $GF(3)$  generated by

$$\begin{bmatrix} 1 & 0 & 1 & 2 \\ 0 & 1 & 1 & 1 \end{bmatrix}.$$

In [6], [8] Conway has investigated the relation between the tetracode and the  $(12, 6, 6)$  ternary Golay code. Pless [15] has employed the tetracode for the hard decision decoding of the ternary Golay code. It turns out that the tetracode can be used for efficient soft-decision decoding as well. Thus decoding the ternary Golay code by means of projecting its 729 codewords onto the nine codewords of the tetracode requires only 530 real operations in the worst case; as compared to 656 real operations of [18] and 1061 operations of [11]. This shows that the approach employed herein for the decoding of the binary Golay code has a natural generalization, i.e., efficient soft decision decoding of both Golay codes may be obtained by projecting them onto shorter codes with less codewords. Whether similar arguments could be generalized to other codes remains an open question.

It is noteworthy that the above reduction of the complexity of maximum-likelihood decoding of the binary Golay code has immediate consequences for the decoding of the Leech lattice as well. Thus for instance, Forney [12] employs the algorithm of [18] for a bounded-distance decoding of the Leech lattice by means of about 2000 operations. Substituting the proposed algorithm into that of Forney [12] enables bounded-distance decoding of the Leech lattice with only 1500 operations, yielding a computational gain factor of about three quarters. In [2] Be'ery, Shahar, and Snyders develop an optimal maximum-likelihood algorithm for the decoding of the Leech lattice (to the best of our knowledge the most efficient algorithm ever published) which requires about 6000 operations. The algorithm presented herein implies that the computational complexity of [2] could be reduced by approximately the same factor.

#### ACKNOWLEDGMENT

This correspondence is a consequence of the last in a series of lectures on coding theory that Vera Pless gave at the Technion in the spring of 1989. During this lecture she raised the question whether the hexacode could be employed for the soft-decision decoding of the binary Golay code. In a sense the derivation above is simply an answer to her question. The authors are indebted to Vera Pless for bringing the matter to their attention. They are also grateful to the referees whose remarks have improved the presentation of the correspondence. Alexander Vardy wishes to thank Hagit Itzkowitz for her invaluable help.

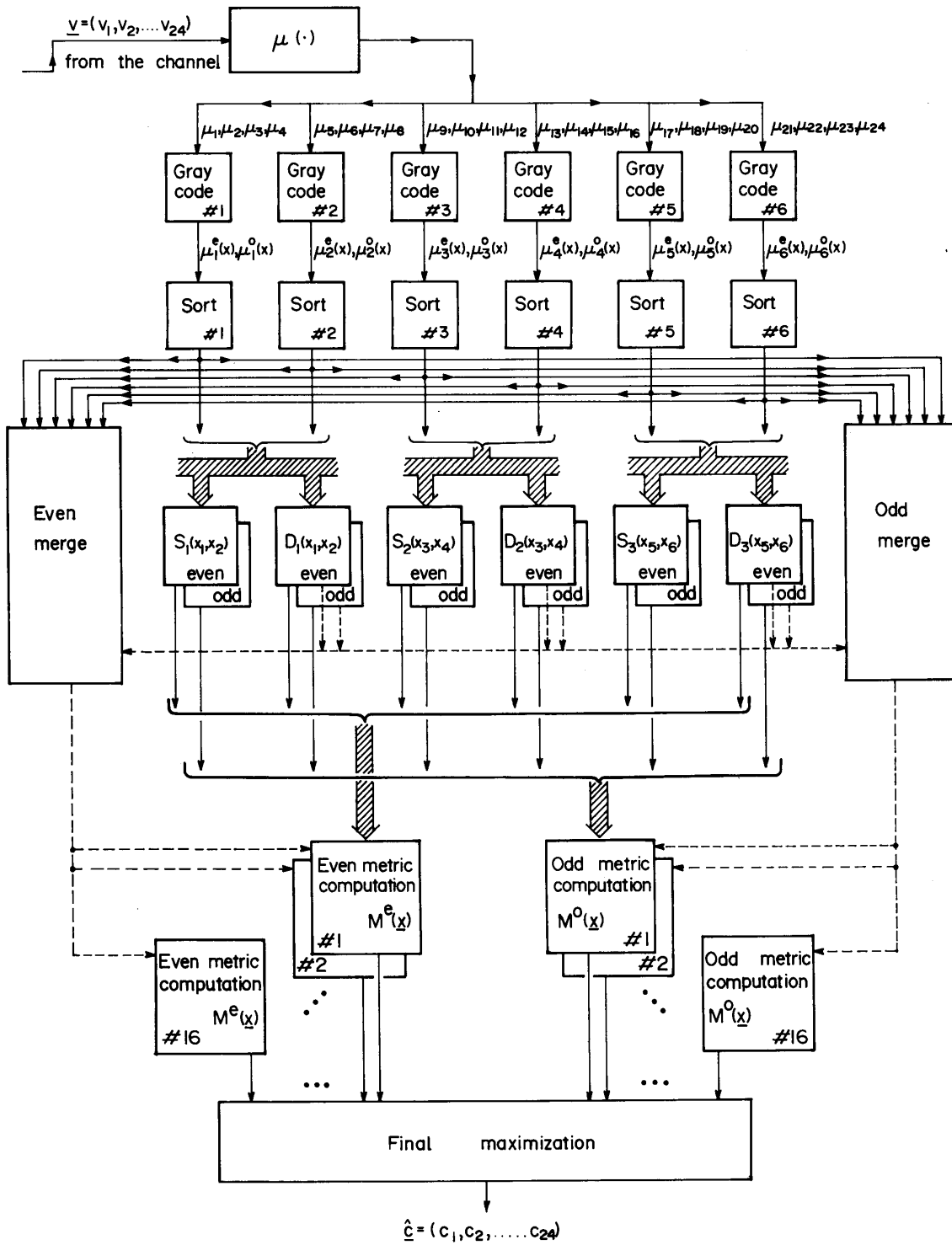


Fig. 1. Soft-decision Golay decoder.

## REFERENCES

- [1] A. D. Abbaszadeh and C. K. Rushforth, "VLSI implementation of a maximum likelihood decoder for the Golay(24,12) code," *IEEE J. Select. Areas Commun.*, vol. 6, pp. 558-565, 1988.
- [2] Y. Be'ery, B. Shahar, and J. Snyders, "Fast decoding of the Leech lattice," *IEEE J. Select. Areas Commun.*, vol. 7, pp. 959-967, 1989.
- [3] Y. Be'ery and J. Snyders, "Optimal soft decision block decoders based on fast Hadamard transform" *IEEE Trans. Inform. Theory*, vol. 32, no. 3, pp. 355-364, May 1986.
- [4] \_\_\_\_\_, "New methods for soft-decision decoding of the Golay(24,12) code," in *Proc. IEEE 15th Conf. Elect. Eng. in Israel*, Tel-Aviv, Israel, 1987.
- [5] A. R. Calderbank and N. J. A. Sloane, "New trellis codes based on lattices and cosets," *IEEE Trans. Inform. Theory*, vol. 33, no. 1, pp. 177-195, Jan. 1987.
- [6] J. H. Conway, "Hexacode and tetracode—MOG and MINIMOG," in *Computational Group Theory*, M. D. Atkinson, Ed. New York: Academic Press, 1984, pp. 359-365.
- [7] J. H. Conway and N. J. A. Sloane, "Soft decoding techniques for codes and lattices, including the Golay code and the Leech lattice," *IEEE Trans. Inform. Theory*, vol. 32, no. 1, pp. 41-50, Jan. 1986.
- [8] \_\_\_\_\_, *Sphere Packings, Lattices and Groups*. New York: Springer-Verlag, 1988.
- [9] R. T. Curtis, "A new combinatorial approach to  $N_{24}$ ," *Math. Proc. Camb. Phil. Soc.*, vol. 79, pp. 25-41, 1976.
- [10] G. D. Forney, Jr., "Coset codes II: Binary lattices and related codes," *IEEE Trans. Inform. Theory*, vol. 34, no. 5, pt. II, pp. 1152-1187, Sept. 1988.
- [11] \_\_\_\_\_, "Coset codes III: Ternary codes, lattices, and trellis codes," *IEEE Trans. Inform. Theory*, to appear.
- [12] \_\_\_\_\_, "A bounded distance decoding algorithm for the Leech lattice, with generalizations," *IEEE Trans. Inform. Theory*, vol. 35, pp. 906-909, July 1989.
- [13] D. E. Knuth, *The Art of Computer programming, vol. 3: Sorting and Searching*. Reading, MA: Addison-Wesley, 1973.
- [14] V. S. Pless, "On the uniqueness of the Golay codes," *J. Comb. Theory*, vol. 5, pp. 215-228, 1968.
- [15] \_\_\_\_\_, "Decoding the Golay codes," *IEEE Trans. Inform. Theory*, vol. 32, no. 4, pp. 561-567, July 1986.
- [16] E. M. Reingold, J. Nievergelt, and N. Deo, *Combinatorial Algorithms: Theory and Practice*. Englewood Cliffs, NJ: Prentice-Hall, 1977.
- [17] R. A. Silverman and M. Balsler, "Coding for a constant data rate source," *IRE Trans. Inform. Theory*, vol. 4, pp. 50-63, 1954.
- [18] J. Snyders and Y. Be'ery, "Maximum likelihood soft decoding of binary block codes and decoders for the Golay codes," *IEEE Trans. Inform. Theory*, vol. 35, no. 5, pp. 963-975, Sept. 1989.

### The Linear Complexity of Binary Sequences with Period $(2^n - 1)^k$

Cheng Hua and Guo-Zhen Xiao

**Abstract**—The linear complexity of binary sequences with period of the form  $(2^n - 1)^k$  is discussed. It is shown that, when  $n$  is a prime number, the linear complexity  $L$  of this kind of sequence has the lower bound

$$L \geq n \left( \sum_{i=1}^m p_i^{(k-1)e_i} \right),$$

Manuscript received May 23, 1990; revised September 19, 1990.  
C. Hua is with the Mathematics Department, Northern Jiaotong University, Beijing 100044, China.  
G.-Z. Xiao is with the Department of Mathematics, Xidian University, Xi'an City, China.  
IEEE Log Number 9041812.

where

$$2^n - 1 = \prod_{i=1}^m p_i^{e_i},$$

$p_i$  are prime numbers and  $(p_i, p_j) = 1$ .

**Index Terms**—Linear complexity of binary sequences

#### I. INTRODUCTION

Most commonly, we use the maximal linear shift register to generate a binary sequence. In recent years, some new generators of binary sequences, such as *clock-controlled shift register* [1] and *cascade-connected clock-controlled shift register* [2], are suggested. Most sequences generated by these models have period of the form  $(2^n - 1)^k$ . Further, many other kinds of binary sequences have this kind of period. In this correspondence we will give the lower bound of linear complexity of all these kinds of sequences that have period of the form  $(2^n - 1)^k$  with  $n$  being a prime.

#### II. THEORETICAL RESULTS

**Lemma:** If  $\text{Ord}_p(2) = n$ ,  $p$  is a prime factor of  $2^n - 1$  and  $(p, (2^n - 1)|r) = 1$ , then we have  $\text{Ord}_{rp}(2) = np$ .

**Proof:** Let  $2^n - 1 = rl$ , then we have

$$(2^n)^p = (1 + rl)^p = 1 + prl + \dots + (rl)^p \equiv 1 \pmod{rp}. \quad (1)$$

Since  $(p, l) = 1$ , we get

$$1 + rl \not\equiv 1 \pmod{rp}, \quad (2^n)^p \not\equiv 1 \pmod{rp^2} \quad (2)$$

such that

$$\text{Ord}_{rp}(2^n) = p, \quad \text{Ord}_{rp^2}(2^n) \neq p; \quad (3)$$

from  $n = \text{Ord}_p(2)|\text{Ord}_{rp}(2)$ , we get the result  $\text{Ord}_{rp}(2) = n \cdot \text{Ord}_{rp}(2^n) = np$ .  $\square$

**Corollary 1:** Under the conditions as in the lemma, we have  $\text{Ord}_{rp} e(2) = np^e$ , where  $e$  is a positive integer.

**Corollary 2:** If  $s$  is a positive integer and all prime factors of  $s$  are factors of  $2^n - 1$  and  $(s, (2^n - 1)|r) = 1$ , then we have  $\text{Ord}_{rs}(2) = ns$ .

**Proof:** Let  $s = p_1^{e_1} p_2^{e_2} \dots p_k^{e_k}$  be the standard factorization of  $s$ . Then every  $p_i^{e_i}$  ( $i = 1, 2, \dots, k$ ) satisfies the conditions in the lemma. Hence we have

$$\text{Ord}_{rp_i^{e_i}}(2) = np_i^{e_i} \quad (i = 1, 2, \dots, k). \quad (4)$$

By using the conclusions in [3], it is easy to get the result as in this corollary.  $\square$

**Theorem:** If the period of a binary sequence  $\{z_n\}$  is  $(2^n - 1)^k$ ,  $k > 1$ , and  $n$  is a prime, then the linear complexity  $L$  of  $\{z_n\}$  has the lower bound

$$L \geq n \left( \sum_{i=1}^m p_i^{(k-1)e_i} \right)$$

where

$$2^n - 1 = \prod_{i=1}^m p_i^{e_i}$$

is the standard factorial resolution of  $2^n - 1$ . When  $2^n - 1$  is a