MacWilliams identities and a symbolic manipulation program we derive from the weight distribution of $\mathcal{D}_2$ that for m even and $m \geq 4$ one has

$$A_7 = \frac{q-1}{7!}(q^3 + 2q^2 + 44q - 272). \tag{11}$$

Counting the total weight of the words of weight 7 the pigeon hole principle implies that there is a position with at least $A_7' = 7A_7/(q-1)$ words having a 1 in that position. Consider these words. There are at least $A_7'' = (7A_7' - A_7')/(q-2) = 6A_7'/(q-2)$ words of weight 7 of which the supports have two positions in common. Proceeding in the same way we find at least $A_7''' = (7A_7'' - 2A_7'')/(q-3)$ words of weight 7 of which the supports have three positions in common.

From (11) it follows that

$$A_7''' = \frac{5 \cdot 6 \cdot 7A_7}{(q-1)(q-2)(q-3)} > 1$$

for $q \geq 16$. Hence there are at least two words of weight 7 with three common positions. This proves d_2(BCH(3))=11 for $m \geq 4$, m even. □

Finally we include an example.

Example 2.4: For $q = 32$ the code BCH(3) is a $(n = 31, k = 16, d_1 = 7)$-code. According to (10) we have seven words of weight 7 with ones in the first two positions. A computer search using the generator polynomial

$$X^{15} + X^{14} + X^{13} + X^{12} + X^{10} + X^8 + X^7 + X^6 + X^5 + X^4 + 1$$

confirmed this and showed that they are the following:

11000000100000100111000000000000
11000001000100101000000100000000
11000000010010100000100010000000
11001010001000100000000000100000
11100000000000011000000100000100
11010100000000010000000000010010
11000000000001100000010000001001

ACKNOWLEDGMENT

We thank B. Ruitenburg for performing the computer search for Examples 1.4 and 2.4.

REFERENCES

[1] G. L. Feng, K. K. Tzeng, and V. K. Wei, "On the generalized Hamming weights of several classes of cyclic codes," *IEEE Trans. Inform. Theory*, vol. 38, pp. 1125–1130, 1992.

[2] G. van der Geer and M. van der Vlugt, "Reed–Muller codes and supersingular curves I," *Compositio Math.*, vol. 84, pp. 333–367, 1992.

[3] T. Helleseth, "On the covering radius of cyclic linear codes and arithmetic codes," *Discrete Appl. Math.*, vol. 11, pp. 157–173, 1985.

[4] J. Silverman, "The arithmetic of elliptic curves," *Grad. Texts in Math.*, vol. 106, New York: Springer–Verlag, 1986.

[5] V. K. Wei, "Generalized Hamming weights for linear codes," *IEEE Trans. Inform. Theory*, vol. 37, pp. 1412–1418, 1991.

Maximum-Likelihood Soft Decision Decoding of BCH Codes

Alexander Vardy and Yair Be'ery

***Abstract*— The problem of efficient maximum-likelihood soft decision decoding of binary BCH codes is considered. It is known that those primitive BCH codes whose designed distance is one less than a power of two, contain subcodes of high dimension which consist of a direct-sum of several identical codes. We show that the same kind of direct-sum structure exists in all the primitive BCH codes, as well as in the BCH codes of composite block length. We also introduce a related structure termed the "concurring-sum", and then establish its existence in the primitive binary BCH codes. Both structures are employed to upper bound the number of states in the minimal trellis of BCH codes, and develop efficient algorithms for maximum-likelihood soft decision decoding of these codes.**

***Index Terms*—BCH codes, maximum-likelihood soft-decision decoding, minimal trellises of block codes.**

I. INTRODUCTION

We consider the problem of efficient maximum-likelihood soft-decoding of binary BCH codes. For short block lengths n, say up to $n = 63$, the primitive binary BCH codes are among the best codes known [5, 16]. These codes have also found a widespread use in a variety of existing communication systems. Nevertheless, no efficient maximum-likelihood soft-decoding algorithm, applicable to the general family of the binary BCH codes, is presently known. In [6] Berlekamp has developed an efficient algorithm for bounded-distance soft decoding of binary BCH codes, which he employed for the decoding of the (128, 106) primitive BCH code. In [13]–[15] Kasami *et al.* present efficient maximum-likelihood soft decision decoders, yet their applicability is limited to several specific codes, namely, the primitive binary double-error-correcting BCH codes, and the (64, 24), (64, 45) BCH codes. A few more examples of efficient maximum-likelihood decoding of several small BCH codes may be found in [18]. A general algorithm which is presently available for maximum-likelihood soft decoding of all the other BCH codes is the conventional Viterbi decoding based on the trellis of Wolf [25]. Several generic improvements to this algorithm have been suggested in a recent work of Berger and Be'ery [4]. There is also the Fast Hadamard Transform algorithm of [2], which is more efficient for the low-rate BCH codes. In this paper we present maximum-likelihood soft decision BCH decoders whose complexity is in some cases orders of magnitude lower than that of [25], [2], or [4]. Our approach is based on exhibiting the existence of certain structures in binary BCH codes, and then employing these structures for efficient decoding.

In [12] Forney has shown that the binary Reed-Muller codes possess a high degree of structure, and in particular contain direct-

Manuscript received September 18, 1992; revised April 12, 1993. This work was supported in part by the Rothschild Fellowship administered by the Rothschild Foundation. This paper was presented in part at the 1993 IEEE International Symposium on Information Theory, San Antonio, TX, January 1993 USA.

A. Vardy was with the IBM Research Division, Almaden Research Center, San Jose, CA 95120. He is now with the Coordinated Science Laboratory, University of Illinois, Urbana, IL 61801 USA.

Y. Be'ery is with the Department of Electrical Engineering, Tel-Aviv University, Ramat-Aviv 69978, Tel-Aviv, Israel.

IEEE Log Number 9400231.

sum subcodes of high dimension. This fact was employed by Forney [12], and subsequently by Aran and Be'ery [1], for efficient soft decision decoding of Reed–Muller codes. It is well-known [5] that certain BCH codes, namely the primitive binary BCH codes whose designed distance is one less than a power of two, are supercodes of punctured RM codes. Hence these BCH codes evidently share the direct-sum structure of the RM codes. This fact was used by Kasami *et al.* [13]–[15] to construct efficient trellis diagrams for their decoders. This leads to the following question: do other BCH codes also contain direct-sum subcodes of high dimension? In the sequel we settle this question affirmatively for *all* the primitive BCH codes, and also for the BCH codes of composite block length.

Apparently, for the purpose of efficient soft-decision decoding, the main property of the direct-sum structure is that the nonzero coordinates of the codes which constitute a direct-sum subcode do not overlap. We, therefore, employ the following definition of the direct-sum structure.

Definition 1: A linear code C of length n is said to have a direct-sum structure if it contains a nontrivial subcode C' which is spanned by some h non-overlapping codes $C'_1, C'_2, \dots C'_h$. More precisely, let $\mathcal{L}_1, \mathcal{L}_2, \dots \mathcal{L}_h$ be a partition of the set $\{0, 1, \dots n-1\}$, and let the codes $C'_1, C'_2, \dots C'_h$ be such that for any $j \in \{1, 2, \dots h\}$ and for any $(c_0, c_1, \dots c_{n-1}) \in C'_j$ we have $c_i = 0$ for all $i \notin \mathcal{L}_j$. Then the direct-sum subcode C' consists of all the vectors of the form $\mathbf{c}_1 + \mathbf{c}_2 + \cdots + \mathbf{c}_h$ where $\mathbf{c}_j \in C'_j$ for $j = 1, 2, \dots h$.

It is evident from the foregoing definition that the direct-sum structure is in a sense a counterpart of the concept of zero-concurring codewords (cf. [2, 18]), obtained by substituting a code for each codeword. In the next section we shall also study a different structure, where we allow the constituent codes to overlap over a fixed set of coordinates. This structure is the corresponding counterpart of the concurring codewords of [2, 22].

Definition 2: A linear code C of length n is said to have a concurring-sum structure if it contains a nontrivial subcode C' which is spanned by some $h+1$ codes $C'_0, C'_1, C'_2, \dots C'_h$ overlapping over a fixed set of coordinates $\mathcal{L}_0$. More precisely, let $\mathcal{L}_0, \mathcal{L}_1, \mathcal{L}_2, \dots \mathcal{L}_h$ be a partition of the set $\{0, 1, \dots n-1\}$, and let the codes $C'_0, C'_1, C'_2, \dots C'_h$ be such that for any $j \in \{0, 1, 2, \dots h\}$ and for any $(c_0, c_1, \dots c_{n-1}) \in C'_j$ we have $c_i = 0$ for all $i \notin \{\mathcal{L}_j \cup \mathcal{L}_0\}$. Then the concurring-sum subcode C' consists of all the vectors of the form $\mathbf{c}_0 + \mathbf{c}_1 + \mathbf{c}_2 + \cdots + \mathbf{c}_h$ where $\mathbf{c}_j \in C'_j$ for $j = 0, 1, 2, \dots h$.

We let $s(C)$ denote the logarithm of the maximum number of states in the minimal trellis of a linear code C (cf. [17, 3]). This parameter, which was first introduced by Forney in [12], governs the complexity of maximum-likelihood decoding of C. Muder [17] claims $s(C)$ to be a fundamental descriptive characteristic of the code, and emphasizes the importance of calculating this parameter for the BCH codes. Both the direct—and the concurring-sum structures make it possible to set nontrivial upper bounds on $s(C)$ for the primitive binary BCH codes. This provides a clue for efficient maximum-likelihood soft-decision decoding, using the algorithm of Forney [12]. A table comparing the decoding complexities obtained using the techniques proposed herein with the complexity of conventional decoding [25, 2] for all the primitive BCH codes of length up to 64 is presented in Section III.

II. STRUCTURES

Let C be a binary BCH code of length n and dimension k, and let α be a primitive nth root of unity. As usual, we label the coordinates of C with powers of α. Let $\mathcal{I}$ be a subset of the set $\{0, 1, \dots n-1\}$. Further, let $C[\mathcal{I}]$ denote the subcode of C which consists of all those codewords that are nonzero only on the positions contained in $\mathcal{I}$. We define $C(\mathcal{I})$ to be the code obtained from $C[\mathcal{I}]$ by puncturing out all the (entirely zero) positions that are not contained in $\mathcal{I}$.

A. Direct-Sum Structures in BCH Codes of Composite Block Length

Although we shall restrict our attention to the BCH codes, most of the results derived in this subsection pertain to cyclic codes in general. This is so because neither of the proofs of the two propositions in the sequel requires that the zeros of the code lie at consecutive powers of α. For instance the following proposition applies to any cyclic code.

Proposition 1: Let $\mathcal{I}_1$ and $\mathcal{I}_2$ be subsets of the set $\{0, 1, \dots n-1\}$, such that for some $a \in \{0, 1, \dots n-1\}$ we have

$$\{\alpha^i : i \in \mathcal{I}_2\} = \{\alpha^a \cdot \alpha^i : i \in \mathcal{I}_1\}. \tag{1}$$

Then $C(\mathcal{I}_1) = C(\mathcal{I}_2)$.

Proof: Assume that $\mathcal{I}_1 = \{i_1, i_2, \dots i_\mu\}$, where $\mu = |\mathcal{I}_1|$ is the cardinality of $\mathcal{I}_1$. Let $\{\alpha_1, \alpha_2, \dots \alpha_r\}$, where $r = n - k$, be the set of zeros of C. Then a parity check matrix for $C(\mathcal{I}_1)$ is:

$$H_1 = \begin{bmatrix} \alpha_1^{i_1} & \alpha_1^{i_2} & \cdots & \alpha_1^{i_\mu} \\ \alpha_2^{i_1} & \alpha_2^{i_2} & \cdots & \alpha_2^{i_\mu} \\ \vdots & \vdots & & \vdots \\ \alpha_r^{i_1} & \alpha_r^{i_2} & \cdots & \alpha_r^{i_\mu} \end{bmatrix}$$

A codeword $\mathbf{c} = (c_1, c_2, \dots c_\mu) \in C(\mathcal{I}_1)$ is a solution of the following set of equations

$$\sum_{\nu=1}^{\mu} c_\nu \cdot \alpha_j^{i_\nu} = 0 \qquad \text{for } j = 1, 2, \dots r. \tag{2}$$

It follows from (1) that a parity check matrix for $C(\mathcal{I}_2)$ may be written as

$$H_2 = \begin{bmatrix} \alpha_1^{a+i_1} & \alpha_1^{a+i_2} & \cdots & \alpha_1^{a+i_\mu} \\ \alpha_2^{a+i_1} & \alpha_2^{a+i_2} & \cdots & \alpha_2^{a+i_\mu} \\ \vdots & \vdots & & \vdots \\ \alpha_r^{a+i_1} & \alpha_r^{a+i_2} & \cdots & \alpha_r^{a+i_\mu} \end{bmatrix}$$

Hence, in order to show that $C(\mathcal{I}_1) \subseteq C(\mathcal{I}_2)$ it is sufficient to observe that any solution of (2) is also a solution of

$$\sum_{\nu=1}^{\mu} c_\nu \cdot \alpha_j^{a+i_\nu} = \alpha_j^a \cdot \sum_{\nu=1}^{\mu} c_\nu \cdot \alpha_j^{i_\nu} = 0 \qquad \text{for } j = 1, 2, \dots r.$$

The converse inclusion follows from the fact that relation (1) is invertible. Namely, if (1) holds then we also have

$$\{\alpha^i : i \in \mathcal{I}_1\} = \{\alpha^{n-a} \cdot \alpha : i \in \mathcal{I}_2\}$$

Thus the foregoing argument with α^a replaced by α^{n-a} shows that $C(\mathcal{I}_2) \subseteq C(\mathcal{I}_1)$. □

Remark: Note that (1) may be equivalently stated as $\mathcal{I}_2 \equiv a + \mathcal{I}_1 (\text{mod } n)$. While the latter form is probably simpler, we have used the specific notation of (1) to emphasize the similarity between Proposition 1 and Proposition 3, to be derived in the next subsection.

Now assume that the length of C is composite, say $n = n_1 \cdot n_2$. Let $Z = \{z_1, z_2, \dots z_r\}$ be the set of zero frequencies of C, i.e. let $\{\alpha_1, \alpha_2, \dots \alpha_r\} = \{\alpha^{z_1}, \alpha^{z_2}, \dots \alpha^{z_r}\}$. Take

$$\mathcal{I}_1 = \{0, n_2, 2n_2, \dots (n_1 - 1)n_2\} \tag{3}$$

and define the set S as follows

$$S = \{s \equiv z \ (\text{mod } n_1) : z \in Z\} \tag{4}$$

Proposition 2: The code $C(\mathcal{I}_1)$ is a BCH code of length n_1 and dimension $k_1 = n_1 - |S|$. The zeros of $C(\mathcal{I}_1)$ lie at $\{\beta^s : s \in S\}$, where $\beta = \alpha^{n_2}$ is a primitive n_1-th root of unity.

Proof: The parity check matrix of $C(\mathcal{I}_1)$ is given by

$$H = \begin{bmatrix} 1 & (\alpha^{z_1})^{n_2} & (\alpha^{z_1})^{2n_2} & \cdots & (\alpha^{z_1})^{(n_1-1)n_2} \\ 1 & (\alpha^{z_2})^{n_2} & (\alpha^{z_2})^{2n_2} & \cdots & (\alpha^{z_2})^{(n_1-1)n_2} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & (\alpha^{z_r})^{n_2} & (\alpha^{z_r})^{2n_2} & \cdots & (\alpha^{z_r})^{(n_1-1)n_2} \end{bmatrix}$$

We denote by m and m_1 the multiplicative orders of 2 modulo n and n_1, respectively. Then $\alpha \in \mathrm{GF}(2^m)$ and $\beta \in \mathrm{GF}(2^{m_1})$. It is well-known [16] that $m_1 | m$ or, equivalently, $\mathrm{GF}(2^{m_1}) \subseteq \mathrm{GF}(2^m)$. Thus, we may view β as an element of $\mathrm{GF}(2^m)$ and write

$$H = \begin{bmatrix} 1 & (\beta^{z_1}) & (\beta^{z_1})^2 & \cdots & (\beta^{z_1})^{(n_1-1)} \\ 1 & (\beta^{z_2}) & (\beta^{z_2})^2 & \cdots & (\beta^{z_2})^{(n_1-1)} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & (\beta^{z_r}) & (\beta^{z_r})^2 & \cdots & (\beta^{z_r})^{(n_1-1)} \end{bmatrix}$$

Consequently $C(\mathcal{I}_1)$ consists of all the binary vectors $\mathbf{c} = (c_0, c_1, \ldots c_{n_1-1})$ which satisfy the following set of equations

$$\sum_{j=1}^{n_1-1} c_j \cdot (\beta^z)^j = 0 \quad \text{for all } z \in Z \tag{5}$$

Now if $z_1, z_2 \in Z$ are such that $z_1 \equiv z_2 \pmod{n_1}$ then $\beta^{z_1} = \beta^{z_2}$. Hence we may replace Z by S in (5). □

Remark: If C itself is not a BCH code—that is the set of zeros of C does not contain a string of consecutive powers of α, then $C(\mathcal{I}_1)$ is not necessarily a BCH code either. All other claims of Proposition 2 hold without change, however, in this case as well.

The sets Z and S are unions of cyclotomic cosets modulo n and n_1, respectively. Thus, relation (4) defines what may be referred to as *coset aliasing* between the cyclotomic cosets modulo n and modulo n_1. For example for $n = 45$ and $n_1 = 15$ we have

$$\begin{aligned} \mathbf{C}_0^{45}, \mathbf{C}_{15}^{45} &\longrightarrow \mathbf{C}_0^{15} \\ \mathbf{C}_1^{45} &\longrightarrow \mathbf{C}_1^{15} \\ \mathbf{C}_3^{45}, \mathbf{C}_9^{45}, \mathbf{C}_{21}^{45} &\longrightarrow \mathbf{C}_3^{15} \\ \mathbf{C}_5^{45} &\longrightarrow \mathbf{C}_5^{15} \\ \mathbf{C}_7^{45} &\longrightarrow \mathbf{C}_7^{15} \end{aligned} \tag{6}$$

where $\mathbf{C}_i^n$ denotes the cyclotomic coset modulo n which contains the integer i. The meaning of $\mathbf{C}_{i_1}^n \rightarrow \mathbf{C}_{i_2}^{n_1}$ is that $\alpha^{i_1} \in Z$ implies $\beta^{i_2} \in S$. However, given $\mathbf{C}_{i_1}^n \rightarrow \mathbf{C}_{i_2}^{n_1}$, $\beta^{i_2} \in S$ does not necessarily imply that $\alpha^{i_1} \in Z$, since several cyclotomic cosets modulo n may alias as a single coset modulo n_1. The coset aliasing map such as (6) provides a simple way of determining the dimension of $C(\mathcal{I}_1)$ given the zeros of C. For instance if $Z = \mathbf{C}_3^{45} \cup \mathbf{C}_5^{45} \cup \mathbf{C}_7^{45} \cup \mathbf{C}_9^{45}$, using (6) we immediately conclude that $S = \mathbf{C}_3^{15} \cup \mathbf{C}_5^{15} \cup \mathbf{C}_7^{15}$ and, hence, the dimension of $C(\mathcal{I}_1)$ is $k_1 = n_1 - |S| = 15 - |\mathbf{C}_3^{15}| - |\mathbf{C}_5^{15}| - |\mathbf{C}_7^{15}| = 5$.

It is noteworthy that if we would have employed frequency domain representation of C (cf. [7]), we would find that certain high frequencies of C alias as low frequencies in $C(\mathcal{I}_1)$. This is intuitively plausible since for $\mathcal{I}_1 = \{0, n_2, 2n_2, \ldots (n_1-1)n_2\}$ the code $C(\mathcal{I}_1)$ is just the time domain sampling of C at regular intervals.

Using Proposition 1 in conjunction with Proposition 2 one can construct direct-sum subcodes of high dimension in BCH codes of composite block length. We partition the set $\{0, 1, \ldots n-1\}$ into n_2 disjoint subsets $\mathcal{I}_1, \mathcal{I}_2, \ldots \mathcal{I}_{n_2}$ such that for $j = 1, 2, \ldots n_2$,

$$\{\alpha^i : i \in \mathcal{I}_j\} = \{\alpha^{j-1} \cdot \alpha_i : i \in \mathcal{I}_1\}$$

where $\mathcal{I}_1$ is as defined in (3). Then by Proposition 1, $C(\mathcal{I}_1) = C(\mathcal{I}_2) = \cdots = C(\mathcal{I}_{n_2})$, and by Proposition 2, $C(\mathcal{I}_1) \oplus C(\mathcal{I}_2) \oplus \cdots \oplus C(\mathcal{I}_{n_2})$ is a direct-sum subcode of C whose dimension is given by $n_2(n_1 - |S|) = n - n_2|S|$.

Example: Let C be the $(93, 53)$ binary BCH code with zeros at $\alpha^1, \alpha^3, \alpha^5, \alpha^7, \alpha^9$ and the cyclotomic conjugates thereof. Take

$$\mathcal{I}_1 = \{0, 3, 6, \ldots 90\}, \quad \mathcal{I}_2 = \{1, 4, 7, \ldots 91\}, \quad \mathcal{I}_3 = \{2, 5, 8, \ldots 92\}.$$

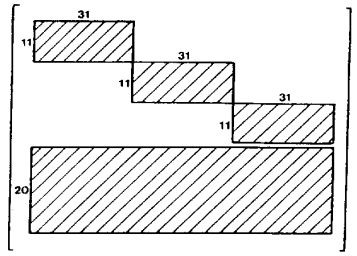
The coset aliasing map is

$$\begin{aligned} \mathbf{C}_{31}^{93}, \mathbf{C}_0^{93} &\longrightarrow \mathbf{C}_0^{31} \quad \mathbf{C}_{33}^{93}, \mathbf{C}_1^{93} \longrightarrow \mathbf{C}_1^{31} \\ \mathbf{C}_{17}^{93}, \mathbf{C}_3^{93} &\longrightarrow \mathbf{C}_3^{31} \quad \mathbf{C}_9^{93}, \mathbf{C}_5^{93} \longrightarrow \mathbf{C}_5^{31} \\ \mathbf{C}_{45}^{93}, \mathbf{C}_7^{93} &\longrightarrow \mathbf{C}_7^{31} \quad \mathbf{C}_{21}^{93}, \mathbf{C}_{11}^{93} \longrightarrow \mathbf{C}_{11}^{31} \\ \mathbf{C}_{23}^{93}, \mathbf{C}_{15}^{93} &\longrightarrow \mathbf{C}_{15}^{31} \end{aligned}$$

Hence,

$$\begin{aligned} Z &= \mathbf{C}_1^{93} \cup \mathbf{C}_3^{93} \cup \mathbf{C}_5^{93} \cup \mathbf{C}_7^{93} \cup \mathbf{C}_9^{93} \\ S &= \mathbf{C}_1^{31} \cup \mathbf{C}_3^{31} \cup \mathbf{C}_5^{31} \cup \mathbf{C}_7^{31}. \end{aligned}$$

Thus the codes $C(\mathcal{I}_1)$, $C(\mathcal{I}_2)$ and $C(\mathcal{I}_3)$ are all equivalent to the BCH code of length 31 and dimension $31 - |S| = 11$, with zeros at $\beta, \beta^3, \beta^5, \beta^7$ and their cyclotomic conjugates. Therefore the generator matrix of C has the following structure



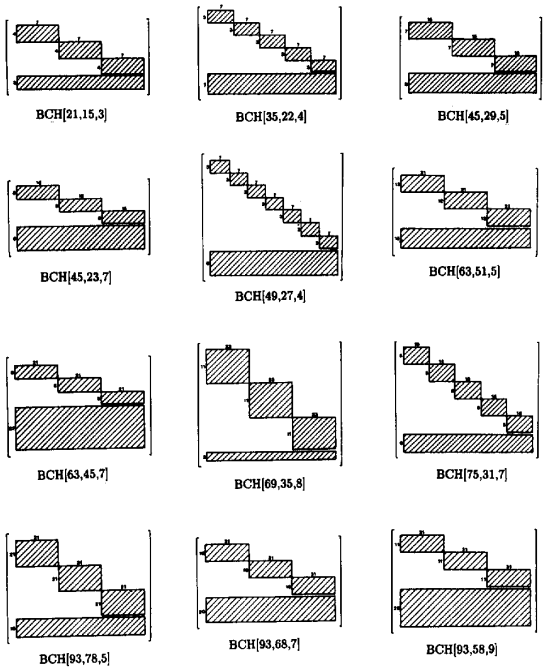
Several similar structures which were obtained using this technique are listed in Table I. This table provides a schematic representation of the direct-sum structure of the generator matrix for several composite-length BCH codes of interest.

Note in particular that the $(69, 35, 8)$ BCH code contains a $(69, 33, 8)$ subcode which is the direct-sum of three $(23, 11, 8)$ codes obtained by shortening the $(24, 12, 8)$ extended binary Golay code. This immediately enables particularly efficient soft-decision decoding of the $(69, 35, 8)$ BCH code using the methods of [21]. More specifically, the $(69, 35, 8)$ code may be decoded with only $4(3 \cdot 622 + 2) + 3 = 7475$ real operations in the worst-case, or about 213 operations per information bit.

B. Direct-Sum Structures in Primitive BCH Codes

In this and the next subsections we consider the *primitive* BCH codes, namely those BCH codes whose block length is one less than a power of two. Furthermore, we shall assume that the zeros of these codes lie at $\alpha, \alpha^2, \ldots, \alpha^{\delta-1}$ and the cyclotomic conjugates thereof. Such codes are called *narrow-sense*. Let C^* be a primitive narrow-sense BCH code of length $n = 2^m - 1$. Henceforth we shall denote by C the *extended* code of C^*, that is a code of length $n + 1 = 2^m$ and dimension k obtained by appending to a parity check matrix of C^* the all-zero column and then the all-ones row. We shall label the additional column of C by $\alpha^\infty = 0$. It turns out that the results derived in the sequel may be stated more concisely for the extended primitive BCH codes. Similar results for the non-extended codes readily follow by shortening the corresponding extended codes.

TABLE I
DIRECT-SUM STRUCTURES IN BCH CODES OF COMPOSITE BLOCK LENGTH



Proposition 3: Let $\mathcal{I}_1$ and $\mathcal{I}_2$ be subsets of the set $\{0, 1, \ldots n-1, \infty\}$, such that for some $a \in \{0, 1, \ldots n-1\}$ we have

$$\{\alpha^i : i \in \mathcal{I}_2\} = \{\alpha^a + \alpha^i : i \in \mathcal{I}_1\}. \tag{7}$$

Then $C(\mathcal{I}_1) = C(\mathcal{I}_2)$.

Proof: Again let $\mathcal{I}_1 = \{i_1, i_2, \ldots i_\mu\}$. Then a parity check matrix for $C(\mathcal{I}_1)$ is

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha^{i_1} & \alpha^{i_2} & \cdots & \alpha^{i_\mu} \\ (\alpha^{i_1})^2 & (\alpha^{i_2})^2 & \cdots & (\alpha^{i_\mu})^2 \\ \vdots & \vdots & & \vdots \\ (\alpha^{i_1})^{\delta-1} & (\alpha^{i_2})^{\delta-1} & \cdots & (\alpha^{i_\mu})^{\delta-1} \end{bmatrix}$$

and, using (7), a parity check matrix for $C(\mathcal{I}_2)$ is

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha^a + \alpha^{i_1} & \alpha^a + \alpha^{i_2} & \cdots & \alpha^a + \alpha^{i_\mu} \\ (\alpha^a + \alpha^{i_1})^2 & (\alpha^a + \alpha^{i_2})^2 & \cdots & (\alpha^a + \alpha^{i_\mu})^2 \\ \vdots & \vdots & & \vdots \\ (\alpha^a + \alpha^{i_1})^{\delta-1} & (\alpha^a + \alpha^{i_2})^{\delta-1} & \cdots & (\alpha^a + \alpha^{i_\mu})^{\delta-1} \end{bmatrix}$$

Thus in order to show that $C(\mathcal{I}_1) \subseteq C(\mathcal{I}_2)$ we have to prove that if a binary vector $(c_1, c_2, \ldots c_\mu)$ solves the following set of δ equations

$$\begin{cases} \sum_{\nu=1}^{\mu} c_\nu \cdot (\alpha^{i_\nu})^j = 0 \quad \text{for } j = 1, 2, \ldots \delta - 1 \\ \sum_{\nu=1}^{\mu} c_\nu = 0 \end{cases} \tag{8}$$

then it also satisfies

$$\begin{cases} \sum_{\nu=1}^{\mu} c_\nu \cdot (\alpha^a + \alpha^{i_\nu})^j = 0 \quad \text{for } j = 1, 2, \ldots \delta - 1 \\ \sum_{\nu=1}^{\mu} c_\nu = 0 \end{cases} \tag{9}$$

We prove this statement by induction on δ. Clearly for $\delta = 2$ the equations

$$\begin{cases} \sum_{\nu=1}^{\mu} c_\nu \cdot \alpha^{i_\nu} = 0 \\ \sum_{\nu=1}^{\mu} c_\nu = 0 \end{cases}$$

imply

$$\sum_{\nu=1}^{\mu} c_\nu \cdot (\alpha^a + \alpha^{i_\nu}) = \sum_{\nu=1}^{\mu} c_\nu \cdot \alpha^{i_\nu} + \alpha^a \cdot \sum_{\nu=1}^{\mu} c_\nu = 0.$$

By induction hypothesis any solution of (8) also satisfies (9). It remains to be shown that (8) in conjunction with

$$\sum_{\nu=1}^{\mu} c_\nu \cdot (\alpha^{i_\nu})^\delta = 0 \tag{10}$$

implies

$$\sum_{\nu=1}^{\mu} c_\nu \cdot (\alpha^a + \alpha^{i_\nu})^\delta = 0. \tag{11}$$

Using binomial coefficients we write

$$(\alpha^a + \alpha^{i_\nu})^\delta = (\alpha^a)^\delta + \sum_{r=1}^{\delta-1} \binom{\delta}{r} (\alpha^{i_\nu})^r (\alpha^a)^{\delta-r} + (\alpha^{i_\nu})^\delta.$$

Hence

$$\sum_{\nu=1}^{\mu} c_\nu \cdot (\alpha^a + \alpha^{i_\nu})^\delta = \sum_{\nu=1}^{\mu} c_\nu \cdot (\alpha^a)^\delta + \sum_{\nu=1}^{\mu} \sum_{r=1}^{\delta-1} c_\nu \cdot \binom{\delta}{r} \times (\alpha^{i_\nu})^r (\alpha^a)^{\delta-r} + \sum_{\nu=1}^{\mu} c_\nu \cdot (\alpha^{i_\nu})^\delta$$

Yet, by (8)

$$\sum_{\nu=1}^{\mu} \sum_{r=1}^{\delta-1} c_\nu \cdot \binom{\delta}{r} (\alpha^{i_\nu})^r (\alpha^a)^{\delta-r} = \sum_{r=1}^{\delta-1} \binom{\delta}{r} (\alpha^a)^{\delta-r} \sum_{\nu=1}^{\mu} c_\nu \cdot (\alpha^{i_\nu})^r = 0,$$

and

$$\sum_{\nu=1}^{\mu} c_\nu \cdot (\alpha^a)^\delta = (\alpha^a)^\delta \sum_{\nu=1}^{\mu} c_\nu = 0$$

Hence if $(c_1, c_2, \ldots c_\mu)$ satisfies (8), then

$$\sum_{\nu=1}^{\mu} c_\nu \cdot (\alpha^a + \alpha^{i_\nu})^\delta = \sum_{\nu=1}^{\mu} c_\nu \cdot (\alpha^{i_\nu})^\delta.$$

In view of (10) this proves the induction step. As in the proof of Proposition 1, the converse inclusion follows by observing that relation (7) is symmetric. Namely, provided α belongs to a field of characteristic two,

$$\{\alpha^i : i \in \mathcal{I}_2\} = \{\alpha^a + \alpha^i : i \in \mathcal{I}_1\}$$

implies

$$\{\alpha^i : i \in \mathcal{I}_1\} = \{\alpha^a + \alpha^i : i \in \mathcal{I}_2\}$$

and vice versa □

Proposition 3 may be thought of as the "addition" counterpart of Proposition 1. Indeed, we have obtained direct-sum subcodes in composite block length BCH codes by partitioning the set $\{\alpha^0, \alpha^1, \ldots \alpha^{n-1}\}$ into disjoint subsets satisfying the "product"

relation (1) with respect to a given subset. Similarly, we can exhibit the existence of direct-sum subcodes in the extended primitive BCH codes by means of partitioning the set $\{\alpha^0, \alpha^1, \ldots \alpha^{n-1}, \alpha^\infty\}$ into disjoint subsets, each satisfying the "addition" relation (7) with respect to some given subset, and then applying Proposition 3.

In case of the primitive BCH codes the set $\{\alpha^0, \alpha^1, \ldots \alpha^{n-1}, \alpha^\infty\}$ constitutes a field GF(2^m), where $m = \log(n+1)$. Thus it would suffice to find a partition $A_1, A_2, \ldots A_h$ of GF(2^m), such that

1) $A_1 \cup A_2 \cup \cdots \cup A_h = \mathrm{GF}(2^m)$
2) For any $i \neq j$: $A_i \cap A_j = \emptyset$
3) For any j_1, j_2 and for some $\alpha^a \in \mathrm{GF}(2^m)$: $A_{j_1} = \{\alpha^a + \alpha_i : \alpha_i \in A_{j_2}\}$ (12)

One way to obtain a partition satisfying (12) is to regard GF(2^m) as a vector space and partition it into a subspace and its cosets. This is certainly not the only possible way. Consider for instance:

$$\mathrm{GF}(16) = \left\{\begin{smallmatrix}0000\\1000\\0100\\0010\end{smallmatrix}\right\} \bigcup \left\{\begin{smallmatrix}1110\\0110\\1010\\1100\end{smallmatrix}\right\} \bigcup \left\{\begin{smallmatrix}1101\\0101\\1001\\1111\end{smallmatrix}\right\} \bigcup \left\{\begin{smallmatrix}0011\\1011\\0111\\0001\end{smallmatrix}\right\}$$

Such nonlinear partitions of the m-dimensional Hamming space are studied in [9]. However the partition into a subspace and its cosets is much more convenient to deal with than the nonlinear partitions of [9]. The number of different partitions into a subspace and its cosets is given by the binary Gaussian coefficient

$$\begin{bmatrix} m \\ m - \log h \end{bmatrix} = \frac{\prod_{i=0}^{m-\log h-1}(2^m - 2^i)}{\prod_{i=0}^{m-\log h-1}(2^{m-\log h} - 2^i)} \qquad (13)$$

Given a partition $A_1, A_2, \ldots A_h$ satisfying (12), define

$$\mathcal{I}_j = \{i : \alpha^i \in A_j\} \qquad \text{for } j = 1, 2, \ldots h.$$

Then by Proposition 3, $C(\mathcal{I}_1) = C(\mathcal{I}_2) = \cdots = C(\mathcal{I}_h)$, and $C(\mathcal{I}_1) \oplus C(\mathcal{I}_2) \oplus \cdots \oplus C(\mathcal{I}_h)$ is a direct-sum subcode of C of dimension $h \cdot \dim C(\mathcal{I}_1)$.

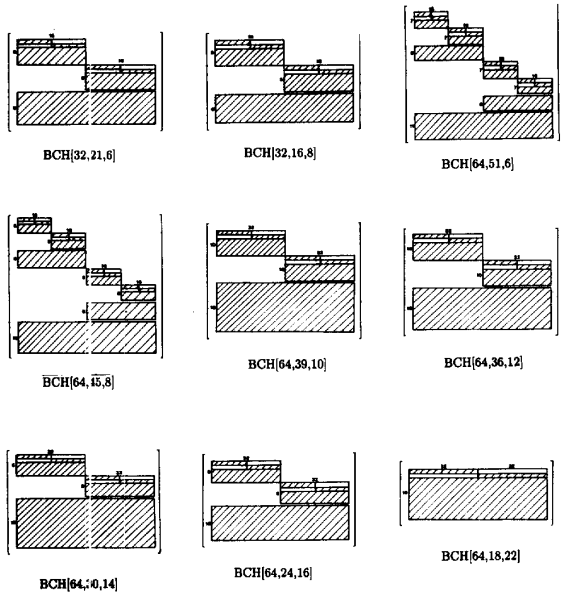
For n up to 63 the number of different linear partitions, given by (13), is upper bounded by 1395. Therefore we could readily go through all the possibilities, using a computer, and choose those which yield direct-sum subcodes of highest dimension. Some of the structures we have obtained this way in extended primitive BCH codes of lengths 16, 32, and 64 are listed in Table II. Note that the direct-sum structure for the (32, 16, 8) BCH code, which coincides with the second-order Reed-Muller code of the same length, is identical to the direct-sum structure found by Forney in [12]. The direct-sum structures for the (64, 45, 8) and (64, 24, 16) BCH codes, which are supercodes of the corresponding Reed-Muller codes, are the same as those found by Kasami et al. [13, 14, 15]. Since the direct-sum structures found in [12, 13, 14, 15] are the best possible for the codes considered, this shows that the techniques proposed herein are likely to find the "best" direct-sum structures in other BCH codes as well. Note, however, that the structure of the Reed-Muller codes, which underlies the direct-sum structure in their BCH supercodes, is actually a multidimensional Kronecker product (cf. [12]) that is not easily shown by generator matrices.

C. Concurring-Sum Structures in Primitive BCH Codes

As before, let C be an extended primitive narrow-sense BCH code of length 2^m. In order to obtain a concurring-sum structure in C by "partitioning" GF(2^m), we need to find a family of sets $A_0, A_1, A_2, \ldots A_h$, such that

1) $A_1 \cup A_2 \cup \cdots \cup A_h = \mathrm{GF}(2^m)$
2) For any $i \neq j$: $A_i \cap A_j = A_0$
3) For any nonzero j_1, j_2 and for some $\alpha^a \in \mathrm{GF}(2^m)$: $A_{j_1} = \{\alpha^a + \alpha_i : \alpha_i \in A_{j_2}\}$ (14)

TABLE II
DIRECT-SUM STRUCTURES IN PRIMITIVE BCH CODES



For any $A_0, A_1, A_2, \ldots A_h$, satisfying (14) define $\mathcal{I}_j = \{i : \alpha^i \in A_j\}$ for $j = 0, 1, 2, \ldots h$. Then by Proposition 3, $C(\mathcal{I}_1) = C(\mathcal{I}_2) = \cdots = C(\mathcal{I}_h)$, and $\bigcup_{j=0}^{h} C(\mathcal{I}_j)$ is a concurring-sum subcode of C of dimension

$$h \cdot \Big(\dim C(\mathcal{I}_1) - \dim C(\mathcal{I}_0)\Big) + \dim C(\mathcal{I}_0),$$

where $\mathcal{L}_0 = \{i : \alpha^i \in A_0\} = \mathcal{I}_0$ and $\mathcal{L}_j = \{i : \alpha^i \in A_j \wedge \alpha^i \notin A_0\} = \mathcal{I}_j \backslash \mathcal{I}_0$, according to Definition 2.

In the following we describe a simple way to obtain a partially overlapping partition, satisfying (14). Let V_1, V_2 be two distinct subspaces of GF(2)^m, such that $V_1 \cup V_2$ contains a basis of GF(2)^m. Denote by v_1, v_2 the dimensions of V_1 and V_2, respectively. Let u be the dimension of $V_1 \cap V_2$, and let $\{\alpha_1, \alpha_2, \ldots \alpha_u\}$ be a basis of $V_1 \cap V_2$. Find a set $\{\beta_1, \beta_2, \ldots \beta_{v_1-u}\}$ such that $\{\alpha_1, \alpha_2, \ldots \alpha_u, \beta_1, \beta_2, \ldots \beta_{v_1-u}\}$ is a basis of V_1. Let $\{\gamma_1, \gamma_2 \ldots \gamma_h\}$ be the span of $\{\beta_1, \beta_2, \ldots \beta_{v_1-u}\}$, i.e. the set of all elements of V_1 that may be expressed in the form $a_1\beta_1 + a_2\beta_2 + \cdots + a_{v_1-u}\beta_{v_1-u}$, where $a_i \in$GF(2). Take $A_0 = V_1$ and define

$$A_j = \{\gamma_j + \alpha : \alpha \in V_1 \cup V_2\} \qquad \text{for } j = 1, 2, \ldots h. \qquad (15)$$

Then the family of sets $A_0, A_1, A_2, \ldots A_h$ satisfies (14).

The following parameters of the concurring-sum structure of C are of interest:

1) h—number of constituent codes;
2) ℓ_0—number of coordinates in which these codes overlap;
3) $\ell_1, \ell_2, \ldots \ell_h$—number of nonoverlapping coordinates in each of these codes.

If the concurring-sum structure of C is obtained using (14) and Proposition 3, then evidently $\ell_1 = \ell_2 = \cdots = \ell_h$. Furthermore, if the underlying "partition" of GF(2^m) is derived from (15) we may express h, ℓ_0 and ℓ_1 in terms of v_1 and v_2. Since $V_1 \cup V_2$ must contain a basis of GF(2^m), we have $u = \dim\{V_1 \cap V_2\} = v_1 + v_2 - m$.

Therefore,

$$h = \frac{|\, V_1 \,|}{|\, V_1 \cap V_2 \,|} = 2^{v_1 - u} = 2^{m - v_2}$$

$$\ell_0 = |\, A_0 \,| = |\, V_1 \,| = 2^{v_1} \tag{16}$$

$$\ell_1 = |\, A_1 \,| - \ell_0 = |\, V_1 \cup V_2 \,| - |\, V_1 \,| = 2^{v_1} + 2^{v_2} - 2^{u} - 2^{v_1} = 2^{v_2}(1 - 2^{v_1 - m}).$$

For a given m, we wish to establish the number of concurring-sum structures of C with distinct parameters h, ℓ_0 and ℓ_1. To exclude the trivial structures we require that

$$2 \le h, \ell_0 \le 2^{m-1}.$$

Using (16) this implies

$$1 \le v_1, v_2 \le m - 1.$$

Since $u \ge 0$, we must have $v_2 \ge m - v_1$. Hence there are exactly $\frac{m(m-1)}{2}$ possible choices for v_1 and v_2, each yielding a different set of parameters h, ℓ_0 and ℓ_1. Finally, we wish to enumerate the number of possible ways to choose the subspaces V_1 and V_2. For a given v_1 there are

$$\begin{bmatrix} m \\ v_1 \end{bmatrix} = \frac{\prod_{i=0}^{v_1-1}(2^m - 2^i)}{\prod_{i=0}^{v_1-1}(2^{v_1} - 2^i)}$$

ways to choose the subspace V_1. Now let $K(m, v, u, w)$ be the number of ways to choose a subspace of dimension v from a space of dimension m over GF(2), such that it intersects with a given subspace of dimension w at exactly 2^u points. We have an explicit expression for $K(m, v, u, w)$,

$$K(m, v, u, w) = \begin{bmatrix} w \\ u \end{bmatrix} \cdot \frac{\prod_{i=0}^{v-u-1}(2^m - 2^{w+i})}{\prod_{i=0}^{v-u-1}(2^v - 2^{u+i})}$$

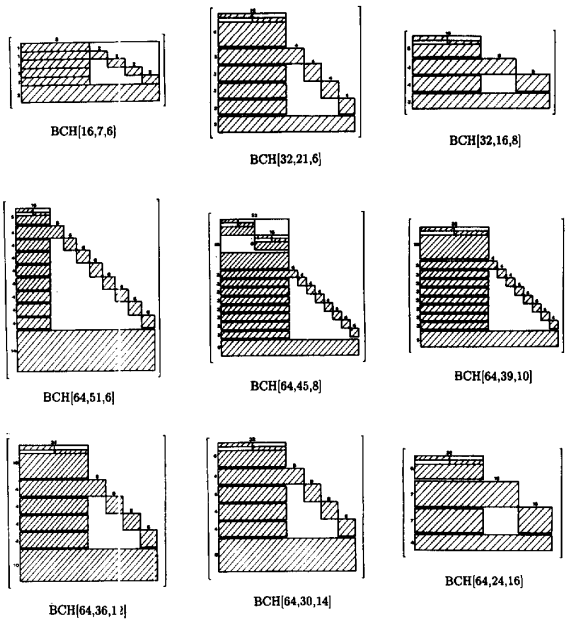
Obviously, once the subspace V_1 has been chosen, there are exactly $K(m, v_2, u, v_1)$ possible ways to choose the subspace V_2. Hence altogether we have

$$\sum_{i=1}^{m-1} \sum_{j=m-i}^{m-1} \begin{bmatrix} m \\ i \end{bmatrix} K(m, j, i + j - m, i)$$

possible ways to choose V_1 and V_2. For $m \le 6$ the foregoing expression is upper bounded by approximately $2.88 \cdot 10^6$. Therefore we could still check all the possibilities on a computer in a reasonable time, and choose those concurring-sum structures which yield the highest reduction in decoding complexity. Some of the structures obtained this way are listed in Table III.

Among the various possible sets of parameters h, ℓ_0 and ℓ_1 those included in Table III correspond to the concurring-sum structures which provide for the most efficient decoding of the corresponding code. Though in some cases the coordinate orderings obtained from the direct- and concurring-sum structures coincide, in general this is not true. For more details on this see the next section. Note that for most of the direct-sum structures presented in Table II, each of the non-overlapping codes constituting the direct-sum subcode C' in itself has a direct-sum structure. Unlike the direct-sum case, however, we were unable to exhibit any particular structure for the partially overlapping subcodes in Table III. Finally, the $(64, 51, 6)$ BCH code is peculiar in that it is the only code in Table III for which ℓ_1 is not a power of two. We have no explanation for this phenomenon.

TABLE III
CONCURRING-SUM STRUCTURES IN PRIMITIVE BCH CODES



III. SOFT DECISION DECODING

We now show how the direct- and concurring-sum structures of BCH codes, derived in the previous section, can be employed for efficient soft decision decoding. In general, we shall use two different approaches to maximum-likelihood soft decision decoding of linear block codes. The first of these is Forney trellis decoding [12]. In the sequel we explicitly calculate the exact computational complexity (that is, the number of real addition-equivalent operations) required by the trellis decoder of Forney, and show how the direct- and concurring-sum structures of BCH codes may be incorporated in this decoder. In some cases however, particularly for the very low-rate codes, decoding by direct decomposition into cosets based on the direct-sum structure yields lower decoding complexity than Forney trellis decoder. Since this technique is superior to Forney trellis decoding in only few instances (see Table V), we will not describe all the relevant details here, and refer the reader to the general discussions in [1, 10, 18], and especially [20].

In [12] Forney presents the following trellis construction for linear block codes. Let the coordinates of C, a binary linear code of length N and dimension K, be arranged in some definite order. In compliance with the notation of [12] we shall henceforth label the coordinates of C by $1, 2, \ldots N$. For each $i = 0, 1, \ldots N$ define the dimension of the future subcode f_i and the dimension of the past subcode p_i as follows

$$f_i = \dim C(\mathcal{F}_i) \quad p_i = \dim C(\mathcal{P}_i)$$

where $\mathcal{F}_i = \{i + 1, i + 2, \ldots N\}$, $\mathcal{P}_i = \{1, 2, \ldots i\}$, and the dimension of $C(\emptyset)$ is taken to be zero. Then the dimension of the state space at position i is given by

$$s_i = K - f_i - p_i.$$

The sequence $s_0, s_1, \ldots s_N$ is said to be the *state-enumeration* sequence for C. The code C may now be described (cf. [12, 17]) by a trellis diagram having 2^{s_i} states at each position $i = 0, 1, 2, \ldots N$. This trellis may be used for maximum-likelihood soft decoding of C.

It is intuitively clear that the computational complexity of a decoder following such a trellis for C is dominated by the values of the state-enumeration sequence $s_0, s_1, \ldots s_N$. A slightly better measure of decoding complexity is given by the sequence $b_0, b_1, \ldots b_{N-1}$, where b_i is the logarithm of the total number of branches in the trellis at the transition from position i to position $i+1$. For a full bit-by-bit binary trellis we have $b_i = s_i + f_i - f_{i+1} = K - p_i - f_{i+1}$. Since $f_i - f_{i+1} = 0$ or 1, the branch complexity of a code is closely approximated by the state complexity. However, the state complexity may be somewhat artificially reduced by means of a technique referred to as "alphabet manipulations" in [17]. That is, instead of using the full bit-by-bit length N trellis, one may use a trellis consisting of fewer than N sections, whereby each section operates on a symbol alphabet corresponding to several consecutive code bits. This technique can often greatly reduce state complexity. For example, decoding the $(24, 12, 8)$ Golay code or the $(32, 16, 8)$ Reed-Muller code by means of a trellis with 8-bit sections reduces the maximum value of the state-enumeration sequence from 9 to 6 (cf. [12]). However, since the reduction in the number of states is achieved at the expense of increasing the number of branches starting at each state, this usually does not reduce the branch complexity. For instance, for the $(24, 12, 8)$ and $(32, 16, 8)$ codes the maximum value of b_i is $\log_2 512 = 9$, for both the binary and the 8-bit section trellises. Since for the binary trellis we have $\max_i s_i \leq \max_i b_i \leq \max_i s_i + 1$, we may conclude that the maximum value of the binary state-enumeration sequence is a good approximate measure of decoding complexity. In the sequel however, notwithstanding the importance of the parameter $\max_i s_i$, we shall assess the decoding complexity in a more exact manner by explicitly calculating the number of real addition-equivalent operations performed by a maximum-likelihood decoder following Forney trellis for C.

Let σ_i and τ_i be the number of additions and, respectively, the number of comparisons performed by such a decoder at the transition from position i to position $i+1$. Clearly, one addition has to be performed at the beginning of each branch. Further, if t branches are incident upon a state, $t-1$ comparisons have to be performed to select the survivor. For each position i, the number of branches starting at a single state is given by $2^{f_i - f_{i+1}}$ and the total number of branches is $2^{s_i} \cdot 2^{f_i - f_{i+1}}$. Therefore

$$\sigma_i = 2^{s_i} \cdot 2^{f_i - f_{i+1}}$$
$$\tau_i = 2^{s_i} \cdot 2^{f_i - f_{i+1}} - 2^{s_{i+1}}$$

Hence the total number of real addition-equivalent operations performed by a decoder following Forney trellis is given by

$$\mathcal{N}_1 = \sum_{i=0}^{N-1} (\sigma_i + \tau_i) = \sum_{i=0}^{N-1} (2 \cdot 2^{s_i + f_i - f_{i+1}} - 2^{s_{i+1}}) \tag{17}$$

As discussed above, the complexity of (17) may be further reduced by using an appropriate sectioning strategy. Using a trellis with fewer than N sections requires a set of section boundaries $\{a_0, a_1, a_2, \ldots a_\nu\}$, where $a_0 = 0 < a_1 < \cdots < a_\nu = N$. Given such a set of section boundaries we may rewrite the codewords of C as $\mathbf{c} = (\alpha_1, \alpha_2, \ldots, \alpha_\nu)$, where each $\alpha_i = (c_{a_{i-1}+1}, c_{a_{i-1}+2}, \ldots c_{a_i})$ is a symbol from an alphabet of size at most $2^{a_i - a_{i-1}}$. For more details on this see [12, 17]. Assuming that a Gray code is employed to calculate the symbol metrics from the metrics of the individual bits, the complexity of the corresponding trellis decoder is given by

$$\mathcal{N}_2 = \sum_{i=1}^{\nu} \left[2^{a_i - a_{i-1}} + (a_i - a_{i-1}) - 2\right] + \sum_{i=1}^{\nu} \left[2^{s_{a_{i-1}} + f_{a_{i-1}} - f_{a_i + 1}} - 2^{s_{a_i}}\right] \tag{18}$$

where the first term represents the complexity of metric computation and the second term stands for the complexity of trellis decoding itself. Notably, for some codes $\mathcal{N}_2$ may be appreciably less than $\mathcal{N}_1$. Indeed, the set of section boundaries should be chosen so as to minimize (18). However the problem of determining the optimal set of section boundaries for a given binary state- and branch-enumeration sequences is, to the best of our knowledge, as yet unresolved. Herein we have employed (possibly suboptimal) trellises with sections of equal length L. Obviously, if L is a power of two this choice of section boundaries is in alignment with the natural boundaries of the direct-sum subcodes of primitive BCH codes.

As discussed above the complexity of Forney trellis decoder, given by (17) and (18), is governed by the values of the state-enumeration sequence. These values in turn depend on the order in which the coordinates of C are arranged. In particular there exists a permutation of the coordinates of C, with the corresponding state-enumeration sequence $s_0, s_1, \ldots s_N$, such that

$$s(C) \stackrel{\text{def}}{=} \max\{s_0, s_1, \ldots s_N\}$$

is less than or equal to the maximum value of the state-enumeration sequence resulting from any other arrangement of the coordinates of C. The parameter $s(C)$, which is usually referred to as the size of the minimal trellis of C, is claimed in [17] to be a fundamental descriptive characteristic of the code, comparable to quantities such as length, dimension and minimum distance. Indeed, to compute $s(C)$ for a given code C we need to find the permutations which minimize the values of the state-enumeration sequence for C. In the following paragraph we show that the direct- and concurring-sum structures of BCH codes, derived in the previous section, may be used to find "good" permutations which yield relatively low values for the state-enumeration sequence.

The following upper bound on $s(C)$ follows from the trellis of Wolf [25]:

$$s(C) \leq \min\{K, N - K\} \tag{19}$$

This bound may be improved by finding zero-concurring or concurring codewords in either C or its dual code, as suggested in [3]. We shall employ the direct-sum and the concurring-sum structures of C instead. The latter approach is considerably more successful, at least for the primitive binary BCH codes. Assume that C contains a subcode C' which is a direct-sum of h identical codes $C'_1, C'_2, \ldots C'_h$, each of length n and dimension k. Arrange the coordinates of C in such a way that every codeword $\mathbf{c} \in C'$ may be written as

$$\mathbf{c} = (c_0, c_1, \ldots c_{N-1}) = (c_{11}, c_{21}, \ldots c_{n1}, c_{12}, c_{22} \ldots c_{n2}, \ldots c_{1h}, c_{2h}, \ldots c_{nh})$$

where $(c_{1j}, c_{2j}, \ldots c_{nj}) \in C'_j$ for each $j = 1, 2, \ldots h$. We shall say that such permutation of the coordinates of C is *in alignment* with its direct-sum structure. It is easy to see that arranging the coordinates of C in alignment with its direct-sum structure yields the following upper bound on $s(C)$

$$s(C) \leq K - (h-1)k. \tag{20}$$

Substituting the parameters of some of the direct-sum structures derived in the previous section into (20) yields upper bounds on $s(C)$ which are often tighter than the bound of (19). Furthermore, in many cases the actual value of $s(C)$ obtained by permuting the coordinates of C in alignment with its direct-sum structure is even lower than the value predicted by (20). Now let C' be a concurring-sum of $h+1$ codes $C'_0, C'_1, C'_2, \ldots C'_h$, overlapping in the first ℓ_0 coordinates. Assume that the coordinates of C are arranged in such

TABLE IV
BOUNDS ON THE SIZE OF THE MINIMAL TRELLIS FOR THE PRIMITIVE BINARY BCH CODES

Code	Bounds on $s(C)$				
	Wolf bound	DS and CS structures	Structure used	Lower bound	Reference
1. BCH[8,4,4]	4	3	DS,CS	3	[15]
2. BCH[16,11,4]	5	4	DS,CS	4	[15]
3. BCH[16,7,6]	7	6	DS,CS	5	(22)
4. BCH[16,5,8]	5	4	DS,CS	4	[15]
5. BCH[32,26,4]	6	5	DS,CS	5	[15]
6. BCH[32,21,6]	11	10	DS,CS	9	(23),[11]
7. BCH[32,16,8]	16	9	DS,CS	9	[15]
8. BCH[32,11,12]	11	10	DS,CS	9	(22)
9. BCH[32,6,16]	6	5	DS,CS	5	[15]
10. BCH[64,57,4]	7	6	DS,CS	6	[15]
11. BCH[64,51,6]	13	12	DS,CS	11	(23),[11]
12. BCH[64,45,8]	19	14	DS	11	[15]
13. BCH[64,39,10]	25	20	DS	12	[17],[8]
14. BCH[64,36,12]	28	19	DS	15	[17],[23]
15. BCH[64,30,14]	30	21	DS,CS	16	[17],[23]
16. BCH[64,24,16]	24	16	DS,CS	14	[15]
17. BCH[64,18,22]	18	17	DS,CS	16	(22)
18. BCH[64,16,24]	16	15	DS,CS	14	(22)
19. BCH[64,10,28]	10	9	DS,CS	9	(21)
20. BCH[64,7,32]	7	6	DS,CS	6	[15]

a way that every codeword $\mathbf{c} \in C'$ may be written as

$$\mathbf{c} = (c_0, c_1, \ldots c_{N-1})$$
$$= (c_{10}, c_{20}, \ldots c_{\ell 0}, c_{11}, c_{21}, \ldots c_{\ell_1 1}, \ldots c_{1h}, c_{2h}, \ldots c_{\ell_1 h})$$

where the vector $(a_1, a_2, \ldots a_{\ell_0}, c_{1j}, c_{2j}, \ldots c_{\ell_1 j}) \in C'_j$ for each $j = 1, 2, \ldots h$ and for some $(a_1, a_2, \ldots a_{\ell_0}) \in \mathrm{GF}(2)^{\ell_0}$. Such permutation of coordinates of C is said to be in alignment with its concurring-sum structure. Even though in this case we do not have an explicit bound on $s(C)$, permuting the coordinates of C in alignment with its concurring-sum structure also yields low values of $s(C)$ in all the primitive binary BCH codes. Some of the upper bounds on $s(C)$ obtained by arranging the coordinates of the code in alignment with its direct- or concurring-sum structure are listed in Table IV.

A comparison between the direct- and concurring-sum structures seems to be due at this stage. In most cases the best permutations resulting from the two kinds of structures yield the same values for $s(C)$, even though the corresponding state-enumeration sequences do not necessarily coincide. In each of the entries in Table IV we indicate which type of structure was used to obtain the upper bound. It may be seen that the direct-sum structure produces tighter bounds on $s(C)$ for the $(64, 45, 8)$, $(64, 39, 10)$ and $(64, 36, 12)$ BCH codes, while in all other cases the upper bounds derived from the two structures coincide. One may get the impression from Table IV that the direct-sum structure is uniformly superior to the concurring-sum structure. To show that this is not necessarily so we give an example where, although the values of $s(C)$ obtained from the two structures coincide, the concurring-sum structure yields lower values for some of the entries in the state-enumeration sequence, as compared with the best state-enumeration sequence found using the direct-sum structure. The positions at which the two state-enumeration sequences disagree are set in boldface.

Note that the state-enumeration sequences resulting from the direct-sum structure are palindromes, that is $s_i = s_{N-i}$. This is due to the fact that the direct-sum subcode is symmetric: writing the code from left to right or from right to left produces the same structure in the generator matrix. This kind of palindrome symmetry constraint on the state-enumeration sequence is absent in the concurring-sum structure and, at least in the example above, this results in non-palindrome state-enumeration sequence with slightly lower values.

We now relate to the lower bounds on $s(C)$ in Table IV. Most of these follow from the results of [15]. In particular [15] accounts directly for entries $1, 2, 4, 5, 7, 9, 10, 12, 16, 20$. Entries $3, 8, 17, 18, 19$ follow from the next proposition. This proposition gives a simple lower bound on $s(C)$ for low-rate high-distance codes, showing that the Wolf upper bound of (19) cannot be improved by more than 1 or 2 units, when d/n is greater than about 0.4 or 0.33, respectively.

Proposition 4: Let d be the minimum distance of linear code C of length N and dimension K. If $d \geq \frac{2}{5}(N+2)$ then

$$s(C) \geq K - 1 \tag{21}$$

If $d \geq \frac{1}{3}(N+2)$ then

$$s(C) \geq K - 2 \tag{22}$$

Proof: The r-th generalized Hamming distance (cf. Wei [24]) of a linear code C, denoted $d_r(C)$, may be defined as the minimum cardinality of a subset $\mathcal{I} \subset \{1, 2, \ldots N\}$, such that $\dim C(\mathcal{I}) \geq r$. For our purposes an inverse function is more useful. Thus we define $\rho_i(C)$ to be the maximum value of r, such that there exists $\mathcal{I} \subset \{1, 2, \ldots N\}$ of cardinality i with $\dim C(\mathcal{I}) \geq r$. With this definition we have $s_i \geq K - \rho_i(C) - \rho_{N-i}(C)$, and hence

$$s(C) \geq K - \min_i \Big(\rho_i(C) + \rho_{n-i}(C)\Big) \tag{23}$$

Clearly $\rho_i(C) = r$ if and only if $d_r(C) \leq i < d_{r+1}(C)$. By the Griesmer bound [16] we have $d_2(C) \geq d + \lceil d/2 \rceil$, and therefore $\rho_{N-d+1}(C) \leq 1$ provided $d \geq \frac{2}{5}(N+2)$. Hence in this case $s(C) \geq K - \rho_{d-1}(C) - \rho_{N-d+1}(C) \geq K - 1$. By a similar argument, if $d \geq \frac{1}{3}(N+2)$ then $\rho_{\lceil \frac{N}{2} \rceil}(C) \leq 1$ and therefore $s(C) \geq K - 2\rho_{\lceil \frac{N}{2} \rceil}(C) \geq K - 2$. □

The bounds for the $(32, 21, 6)$ and the $(64, 51, 6)$ double-error correcting BCH codes can be obtained from (23) by considering their duals whose generalized Hamming distance hierarchy is given in [11]. Finally, since little is known regarding the distance hierarchy of the $(64, 39, 10)$, $(64, 36, 12)$ and $(64, 30, 14)$ BCH codes, we have derived the lower bounds for these codes using [17, Proposition 6] in conjunction with the table of [8, 23].

A general algorithm which is presently available for maximum-likelihood soft decoding of BCH codes is the conventional Viterbi decoder based on the trellis of Wolf [25]. The complexity of such a decoder is given by

$$\mathcal{N}_3 = \begin{cases} (6K - 3N + 5) \cdot 2^{N-K} - 5 & N \leq 2K \\ (3N - 6K + 5) \cdot 2^K - 5 & N > 2K \end{cases} \tag{24}$$

For low-rate codes the complexity of the FHT decoder of [2] given by $\mathcal{N}_4 = K \cdot 2^K$ is often lower. The complexity of these conventional techniques, given by the minimum of $\mathcal{N}_3$ and $\mathcal{N}_4$, is compared in Table V to the decoding complexities that we were able to obtain for the primitive BCH codes of length up to 64, using the direct and concurring-sum structures in conjunction with the techniques outlined in this section. All the figures in the table are given in terms of the number of real addition-equivalent operations required per bit of information. For each code we indicate the specific decoding technique and the type of structure used. For the majority of the codes the most efficient decoding is obtained using Forney trellis, in which case the decoding complexity is given by the minimum of $\mathcal{N}_1$ and $\mathcal{N}_2$. For some low-rate codes the complexity of decoding by decomposition into cosets, using the direct-sum structure, is lower. The reader is referred to [20] for the description of this technique.

Note that the computational gain obtained reaches several orders of magnitude in many cases, and in particular for codes of rate near $1/2$. For instance for the $(64, 30, 14)$ BCH code the proposed techniques are about 1,000 times more efficient. For those BCH codes which

TABLE V
COMPLEXITY OF DECODING PRIMITIVE BINARY BCH CODES

Code	Decoding complexity		
	Conventional decoding	Proposed techniques	Technique employed
1. BCH[8,4,4]	16	6	coset-decoding
2. BCH[16,11,4]	66	26	Forney-trellis – DS,CS
3. BCH[16,7,6]	128	42	Forney-trellis – DS,CS
4. BCH[16,5,8]	32	13	coset-decoding
5. BCH[32,26,4]	160	66	Forney-trellis – DS,CS
6. BCH[32,21,6]	3,413	1,094	Forney-trellis – DS,CS
7. BCH[32,16,8]	20,480	251	Forney-trellis – DS,CS
8. BCH[32,11,12]	2,048	398	Forney-trellis – DS,CS
9. BCH[32,6,16]	64	27	coset-decoding
10. BCH[64,57,4]	348	132	Forney-trellis – DS,CS
11. BCH[64,51,6]	19,113	6,761	Forney-trellis – DS
12. BCH[64,45,8]	$9.67 \cdot 10^6$	21,891	Forney-trellis – DS
13. BCH[64,39,10]	$4.04 \cdot 10^7$	$8.81 \cdot 10^5$	Forney-trellis – DS
14. BCH[64,36,12]	$2.16 \cdot 10^6$	$3.77 \cdot 10^5$	Forney-trellis – DS
15. BCH[64,30,14]	$6.08 \cdot 10^6$	$6.06 \cdot 10^5$	Forney-trellis – DS
16. BCH[64,24,16]	$1.68 \cdot 10^7$	19,645	Forney-trellis – CS
17. BCH[64,18,22]	$2.62 \cdot 10^5$	33,683	Forney-trellis – DS,CS
18. BCH[64,16,24]	65,510	10,581	Forney-trellis – CS
19. BCH[64,10,28]	1,023	461	coset-decoding
20. BCH[64,7,32]	128	55	coset-decoding

coincide with Reed-Muller codes, our complexity results are roughly the same as those of Forney [12]. They are also similar to the results of Kasami et al. for those BCH codes which have been specifically investigated in [13, 14, 15]. However, the approach presented herein applies uniformly well to *all* the primitive BCH codes.

ACKNOWLEDGMENT

The authors are grateful to the referees for their valuable comments on the manuscript. They are indebted to Noga Alon, Yuval Berger, and David Forney for helpful discussions. Alexander Vardy also wishes to thank Hagit Itzkowitz.

REFERENCES

[1] S. Aran, "Efficient soft decision decoding of Reed-Muller codes and related non-linear generalizations," M.Sc. thesis, Tel-Aviv Univ., 1989.
[2] Y. Be'ery and J. Snyders, "Optimal soft decision block decoders based on Fast Hadamard Transform," *IEEE Trans. Inform. Theory*, vol. IT-32, pp. 355–364, 1986.
[3] Y. Berger and Y. Be'ery, "Bounds on the trellis size of linear block codes," *IEEE Trans. Inform. Theory*, vol. IT-39, pp. 203–209, 1993.
[4] Y. Berger and Y. Be'ery, "Soft trellis-based decoder for linear block codes," *IEEE Trans. Inform. Theory*, 1994, to be published.
[5] E. R. Berlekamp, *Algebraic Coding Theory*. New York: McGraw-Hill, 1968.
[6] ——, "The construction of fast, high-rate, soft decision block decoders," *IEEE Trans. Inform. Theory*, vol. IT-29, pp. 372–377, 1983.
[7] R. E. Blahut, *Theory and Practice of Error Control Codes*. Reading, MA: Addison-Wesley, 1983.
[8] A. E. Brouwer and T. Verhoeff, "An updated table of minimum-distance bounds for binary linear codes," *IEEE Trans. Inform. Theory*, vol. IT-39, pp. 662–677, 1993.
[9] G. D. Cohen, S. Litsyn, A. Vardy, and G. Zemor, "Perfect tilings of binary spaces," preprint.
[10] 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. IT-32, pp. 41–50, 1986.
[11] G. L. Feng, K. K. Tzeng, and V. K. Wei, "On the generalized Hamming weights of several classes of cyclic codes," *IEEE Trans. Inform. Theory*, vol. 38, pp. 1125–1130, 1992.
[12] G. D. Forney, Jr., "Coset codes II: Binary lattices and related codes," *IEEE Trans.Inform. Theory*, vol. 34, pp. 1152–1187, 1988.
[13] T. Kasami, T. Takata, T. Fujiwara, and S. Lin, "Trellis diagram construction for some BCH codes," in *IEEE Int. Symp. Inform. Theory and Appl.*, Honolulu, HI, 1990.
[14] ——, "On complexity of trellis structure of linear block codes," *IEEE Int. Symp. Inform. Theory*, Budapest, Hungary, 1991.
[15] ——, "On the optimum bit orders with respect to the state complexity of trellis d agrams for binary linear codes," *IEEE Trans. Inform. Theory*, vol. 39, pp. 242–243, 1993.
[16] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*. New York: North-Holland, 1977.
[17] D. J. Mader, "Minimal trellises for block codes," *IEEE Trans. Inform. Theory*, vol. 34, pp. 1049–1053, 1988.
[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, pp. 963–975, 1989.
[19] A. Vardy and Y. Be'ery, "On the problem of finding zero-concurring codewords," *IEEE Trans. Inform. Theory*, vol. 37, pp. 180–187, 1991.
[20] ——, "Bit-level soft decision decoding of Reed-Solomon codes," *IEEE Trans. Commun.*, vol. 39, pp. 440–445, 1991.
[21] ——, "More efficient soft-decision decoding of the Golay codes," *IEEE Trans. Inform. Theory*, vol. 37, pp. 667–672, 1991.
[22] A. Vardy, J. Snyders, and Y. Be'ery, "Bounds on the dimension of codes and subcodes with prescribed contraction index," *Linear Algebra Appl.*, vol. 142, pp. 237–261, 1990.
[23] T. Verhoeff, "An updated table of minimum-distance bounds for binary linear codes," *IEEE Trans. Inform. Theory*, vol. IT-33, pp. 665–680, 1987.
[24] V. K. Wei, "Generalized Hamming weights for linear codes," *IEEE Trans. Inform. Theory*, vol. 37, pp. 1412–1418, 1991.
[25] J. K. Wolf, "Efficient maximum-likelihood decoding of linear block codes," *IEEE Trans. Inform. Theory*, vol. IT-24, pp. 76–80, 1978.

Generalized Hamming Weights of Trace Codes

Henning Stichtenoth and Conny Voß

***Abstract*—Linear codes over $\mathbb{F}_p$ often admit a natural representation as trace codes of codes that are defined over an extension field $\mathbb{F}_{p^m}$. In this paper, we obtain estimates for the weights of subcodes of such trace codes. Our main result is a far-reaching generalization of the Carlitz–Uchiyama bound for the duals of binary BCH codes. In particular, we prove sharp bounds for the generalized Hamming weights of a large class of codes, including duals of BCH codes, classical Goppa codes, Melas codes, and arbitrary cyclic codes of length $n = p^m - 1$. Our main tool is the theory of algebraic functions over finite fields, in particular the Hasse–Weil bound for the number of places of degree one.**

***Index Terms*—Trace code, BCH code, Goppa code, generalized Hamming weight, algebraic function field, algebraic curve, cryptography.**

I. INTRODUCTION

Let $\mathbb{F}$ be a finite field. For a nonempty subset $A \subseteq \mathbb{F}^n$, we define the *support* of A by

$$\text{supp}(A) := \{i \mid \text{there is an element } a = (a_1, \cdots, a_n) \in A \text{ with } a_i \neq 0\}$$

Manuscript received March 2, 1993.

The authors are with Universität-GH-Essen, Fachbereich 6, Mathematik und Informatik, D-4300 Essen 1, Germany.

IEEE Log Number 9215877.