

## A Sequential Decoder for Linear Block Codes with a Variable Bias-Term Metric

Vladislav Sorokine, *Student Member, IEEE*,  
and Frank R. Kschischang, *Member, IEEE*

**Abstract**—A sequential decoder for linear block codes that performs maximum-likelihood soft-decision decoding is described. The decoder uses a metric computed from a lower bound on the cost of the unexplored portion of the code tree. It is shown that for certain block codes the average computational complexity of this metric is superior to that of the Fano metric. A new function, the cumulative column distance function, is introduced for linear block codes. This function is an important factor that determines the average computational effort of a sequential decoder for a linear block code with an arbitrary maximum-likelihood metric. Simulation results show that a sequential decoder for linear block codes with a fast growing cumulative column distance function achieves a low computational complexity, a result analogous to that for convolutional codes.

**Index Terms**—Maximum-likelihood soft-decision sequential decoding, variable-bias term metric.

### I. INTRODUCTION

The Fano metric is widely used in sequential decoding of convolutional codes on discrete memoryless channels. The optimality of the Fano metric (in the sense of achieving a good balance between computational effort and error probability) is discussed in [1]–[4]. Algorithms for linear block-code trellis construction [5]–[11] naturally lead to the introduction of soft-decision maximum-likelihood (ML) decoders for arbitrary linear block codes (or group block codes in general [9]). Linear block codes are generally easier to study than convolutional codes, and many good block codes are presently known. Their practical implementation in some applications is held back by the lack of soft-decision decoding algorithms. Trellises bypass this difficulty. Linear block codes, however, have trellises with a time-varying number of states. The maximum number of states can be quite large, for example,  $2^{64}$  for the (128, 64) extended BCH code. Although a certain permutation of the code achieves  $2^{43}$  states [12], that is still exceedingly large for practical implementations of the Viterbi algorithm.

Sequential decoding algorithms can be readily extended from convolutional codes to linear block codes. Compared with the Viterbi algorithm, such schemes can provide significant savings in the average number of computations needed for decoding. Here we only consider ML sequential decoders, i.e., decoders with large enough stacks (in the Zigangirov–Jelinek (ZJ) algorithm [13]–[15]) and no time constraints. Sequential decoding of linear block codes was investigated in [16]–[18]. Massey [2] observed that a modification of

Manuscript received July 1, 1995; revised July 14, 1997. This work was supported in part by the Natural Sciences and Engineering Research Council, Canada. The material in this correspondence was presented, in part, at the 1993 Canadian Workshop on Information Theory, Rockland, Ont., May 30–June 2, 1993.

V. Sorokine was with the Department of Electrical and Computer Engineering, University of Toronto, Toronto, Ont., Canada M5S 3G4. He is now with Qualcomm, Inc., San Diego, CA 92121-2779 USA.

F. R. Kschischang is with the Massachusetts Institute of Technology, Cambridge, MA 02139 USA, on leave from the Department of Electrical and Computer Engineering, University of Toronto, Toronto, Ont., Canada M5S 3G4.

Publisher Item Identifier S 0018-9448(98)00074-1.

the Fano metric that maximizes the joint probability of the received signal and the considered path plus the tail of the code tree in certain instances achieves better computational complexity than the Fano metric.

The Fano metric as well as other metrics proposed for sequential decoding algorithms, can be considered as modifications of priority-first search algorithms developed in the computer science [19]. The aim of such algorithms is to find a minimum cost path from the root node in a tree to a goal node. The cost of partial paths leading from the root is determined, with the paths extended in an order determined by assigning a *priority* to each such path. For example, if priority is given to shortest paths, a breadth-first search of the tree results. For decoding, a breadth-first search of the code tree is expected to be inefficient.

In the so-called Algorithm A [19, pp. 74–81] the priority measure consists of two parts. One part reflects the actual cost of the *explored* path leading to the given node and the other is an estimate of the *future* cost of extending this path to reach the goal node. (For block codes the goal node lies at a depth equal to the code length.) In the Fano metric [2] the part that reflects the future cost is the expected value of the future cost taken over all possible extensions of the current node. Thus sequential decoding with the Fano metric is an instance of Algorithm A in the terminology of [19].

It was observed in [20] that the general approach of estimating the future cost by taking the expectation of the future metric with respect to the unexplored portion of the search tree is not always computationally optimal, particularly when additional information about the future is available. Algorithm A is renamed Algorithm A\* when the part of the cost that depends on the unexplored part of the search tree is a *lower bound* on the future cost obtained from certain knowledge about the problem structure. One instance of Algorithm A\*, in which the algorithm relies on knowledge about the code (such as a set or superset of the codeword Hamming weights), was proposed in [18]. More recently, a tutorial paper on the Algorithm A\*, with applications to ML soft-decision decoding of binary block codes, appeared in [21].

In Section II, we describe a metric (priority measure) that differs from both the Fano metric and the metric of [18] and that is trivial to compute compared with them. This metric was discovered independently by the authors and in [16] and [22]. We call this metric the *variable bias term* (VBT) metric, and we give an empirical argument to justify the introduction of the metric in this particular form. Sequential decoding with the VBT metric is another instance of Algorithm A\*.

In Section III, we analyze the performance of sequential decoders that use the VBT metric. By introducing the cumulative column distance function (CCDF), we are able to extend known bounds on computational complexity from convolutional codes to linear block codes. We argue that the VBT metric is particularly suitable for codes with a fast growing CCDF, and we derive an upper bound on this growth.

We note also that the CCDF depends on the particular permutation of code symbols that is assumed. We conjecture that code permutations that yield code trellises of small complexity will also achieve a fast-growing CCDF. The task of finding such permutations is a well known research problem [23]. Luckily, many known good codes (such as the Golay code) have a fast-growing CCDF and so the proposed metric will outperform the Fano metric for these codes. For such codes, the metric we propose performs computationally well

even if the CCDF is not optimal (as in the case of a code in systematic form).

In Section IV, we present the results of simulating the sequential decoder with the VBT metric, the Fano metric, and the zero bias-term (the Viterbi) metric for the (24, 12, 8) Golay code and the former two metrics for the (48, 24, 12) extended quadratic residue code for which good orderings that achieve a fast growing CCDF are known.

When comparing our results with convolutional decoders, it should be noted that a branch extension in the trellis of a block code usually represents one symbol whereas a branch extension in the trellis of a convolutional code may represent several symbols. Thus it would seem that more branch extensions are required for a block code than for a convolutional code of the same length. However, if the block code trellis is sectionalized so that a trellis branch represents several code symbols, the number of branch extensions for a block code can be reduced. We have not pursued such sectionalization in this correspondence, however.

## II. VARIABLE-BIAS TERM METRIC

Sequential decoding schemes for decoding convolutional codes commonly use a probabilistic branch metric, namely, the Fano metric, which can be written for a discrete memoryless channel as

$$M(r_i|v_i) = \log_2 \frac{p(r_i|v_i)}{p(r_i)} - B \quad (1)$$

where  $p(r_i|v_i)$  is the channel transition probability,  $p(r_i)$  is the channel output symbol probability, and  $B$  is a bias term, often chosen to be the code rate  $R$  [1], [2]. For continuous channels (AWGN channels with BPSK modulation, to be exact) the Fano metric assumes the form

$$M(r_i|v_i) = -\log_2 \left[ 1 + \exp \left( -\frac{4(2v_i - 1)r_i\sqrt{E_s}}{N_0} \right) \right] \quad (2)$$

where  $E_s$  is the energy per transmitted bit and  $N_0$  is the one-sided noise power density.

We note that the statistic to be maximized by the ML sequential decoder ( $Pr(m, t_m, y)$  in the notation of [2]) depends on the unexplored part of the code tree, and this dependency is removed by averaging over future "tails" of the code tree [2]. Thus the Fano metric is the expected value of this statistic with respect to the future part of the code tree that is unexplored. The underlying assumption that the code is random assures that the Fano metric will have a good performance over a vast variety of codes. This assumption, however, does not assure the superior performance of the Fano metric in individual cases when more information about the code structure is available.

Let us formulate a different strategy that turns out to be a "winning" strategy for a certain codes. Consider first the Viterbi metric (that is, just the squared distance between a received symbol and a branch symbol). The accumulated Viterbi metric is the sum of the partial metrics and has the geometric interpretation of a squared Euclidean distance between the received point and the closest code point. Suppose a binary code with a  $\pm 1$  binary phase-shift keying (BPSK) scheme is used, and suppose the decoder receives a signal of amplitude  $r_i, i = 1, 2, \dots, n$ , in a particular time interval. We introduce a *variable-bias term* (VBT) branch metric in the following way:

$$M_0(i) = \|r_i - (-1)\|^2 - B_i \quad (3)$$

(corresponding to the transmission of  $-1$ ), and

$$M_1(i) = \|r_i - 1\|^2 - B_i \quad (4)$$

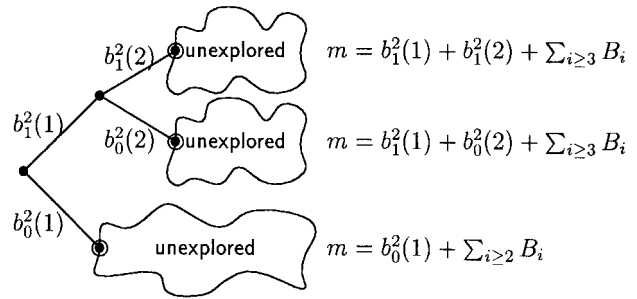


Fig. 1. A lower bound on the total cost to be accumulated in the unexplored future is given by  $\sum B_i$ . The metric associated with the circled nodes is indicated by  $m$ .

(corresponding to the transmission of  $+1$ ), where

$$B_i = \min\{\|r_i - (-1)\|^2, \|r_i - 1\|^2\}.$$

Note that  $B_i$  changes with  $i$ . The squaring operation in (3) and (4) can be avoided for BPSK signaling. Simple manipulations show that

$$M_0(i) = \begin{cases} 0, & \text{if } r_i \leq 0 \\ 4r_i, & \text{if } r_i > 0 \end{cases}$$

and

$$M_1(i) = \begin{cases} 0, & \text{if } r_i \geq 0 \\ -4r_i, & \text{if } r_i < 0. \end{cases}$$

We may consider the accumulated bias term as a lower bound for the accumulated metric of the ML codeword. Fig. 1 illustrates how we make use of this lower bound. The total estimated "cost" associated with any particular path through the tree is given the sum of a) the (actual) cost through the already explored portion of the tree and b) a lower bound on cost through the unexplored portion of the tree. Let  $m(s)$  be the cost estimate associated with a state  $s$  in the tree and let  $b^2$  be the actual cost (squared Euclidean distance) associated with a branch connecting state  $s$  to its successor state  $s^+$ . Then our cost estimate for  $s^+$  is given by

$$m(s^+) = m(s) + (b^2 - B_i) \quad (5)$$

where  $i$  is the depth of  $s^+$  in the tree. We take the quantity  $b^2 - B_i$  as a modified branch metric in our decoding algorithm. The estimated cost of opening the root node of the tree is given by  $m(0) = \sum_{i \geq 1} B_i$ . Thus the VBT decoder uses the *least* possible cost as the estimate of the unexplored part of the code tree. In the following section we will see that this results in computational savings when employing the VBT decoder for certain codes (that have a fast growing CCDF). We will also give a heuristic explanation of this effect.

The decoder with the VBT metric is ML by the following argument. Suppose the total accumulated metric along the path at the output of the decoder is

$$A_n = \sum_{i=1}^n M_j(i) \quad (6)$$

where  $j$  is 0 or 1 depending on branch labels. We use the ZJ algorithm as a sequential decoding scheme. When the decoder makes the decision, all paths in the stack below the top path of length  $n$  have larger partial metrics. The true Euclidean distance between the decoded path and the received sequence is

$$A'_n = \sum_{i=1}^n M_j(i) + \sum_{i=1}^n B_i. \quad (7)$$

The same second term in (7) will be added to any competitive path of length  $n$  as well; so by minimizing  $A_n$  the decoder also minimizes the accumulated metric  $A'_n$ .

Even though the lower bound on the future cost that we use is particularly simple, the decoder with the VBT metric opens fewer nodes than the decoder with the Fano metric for the codes we consider.

### III. ANALYSIS OF THE DECODER

#### A. The Cumulative Column Distance Function

In the analysis of computational complexity of sequential decoding of convolutional codes a prominent role belongs to the *column distance function*. We need a similar function in the analysis of block codes. We introduce the cumulative column distance function (CCDF) for linear block codes in the following way.

Denote by  $start(C) = \{i_1, i_2, \dots, i_k\}$  the set of indices (or time coordinates) where the minimal trellis diagram for an  $(n, k)$  linear block code  $C$  has branches that split from the all-zero codeword (the start set of the code [10]) and suppose  $i_1 < i_2 < \dots < i_k$ . The  $m$ th-column distance function  $d_m(j)$  is

$$d_m(j) = \begin{cases} \min\{w[c]_j : c_1, \dots, c_{i_m-1} = 0, c_{i_m} \neq 0\}, & i_m \leq j \leq n \\ 0, & 0 < j < i_m \end{cases}$$

where  $m \in \{1, \dots, k\}$ ,  $[c]_j$  is the truncation of the codeword  $c$  at the  $j$ th position ( $j = 1, \dots, n$ ),  $w[c]_j$  is the Hamming weight of the truncated codeword, and a minimum is taken over all codewords  $c$  that split from the all-zero codeword in position  $i_m$ .

It is immediately clear that CDF's for a linear block code may have different profiles depending on the index  $m$ . The "plateau" (using the terminology of convolutional codes, even though some CDF's for a block code may not have this flat portion) of each CDF (the value at  $j = n$ ) is lower-bounded by the minimal Hamming weight of the code. The collection of CDF's for the "standard ordering" [8] of the  $(8, 4)$  Reed-Muller code is shown in Fig. 2.

To analyze the performance of the sequential decoder we follow the results of [24]. An upper bound on the number of computations performed by a sequential decoder for a binary code in the  $m$ th incorrect subset is given by

$$P(\mathbf{C}_m > N_m) < \sigma \cdot n_{d_m} e^{-\mu \cdot d_m(\log_2 N_m) + \phi \cdot \log_2 N_m} \quad (8)$$

where  $\mathbf{C}_m$  is the number of computations,  $n_{d_m}$  denotes the number of incorrect paths in the  $m$ th incorrect subset whose distance from the correct path is  $d_m(\log_2 N_m)$ , i.e., the value of the  $m$ th-column distance function at  $\lceil \log_2 N_m + \frac{1}{2} \rceil$ . The parameters  $\mu$ ,  $\phi$ , and  $\sigma$  are the functions of the signal-to-noise ratio (SNR) and the code rate but not  $N_m$ .

This bound, derived originally for convolutional codes, is applicable to block codes since it considers different incorrect subsets individually and so allows to substitute different CDF's for each incorrect subset. We introduce then a *cumulative column distance function* (CCDF) to account for the fact that different subsets of a code tree of a block code have different cardinality (as opposed to convolutional codes). Since those subsets of a code tree that start later have exponentially decreasing cardinality (as a power of 2 for binary codes), we write the CCDF as

$$d(j) = \sum_{m=1}^k 2^{-m} d_m(j)$$

where  $2^{-m}$  is the weighing factor accounting for different cardinalities of incorrect subsets. The CCDF for the  $(8, 4)$  RM code is shown in Fig. 3.

We need the CCDF to apply the result that was obtained by Chevillat and Costello in [24] (see also [25]). In particular, in order

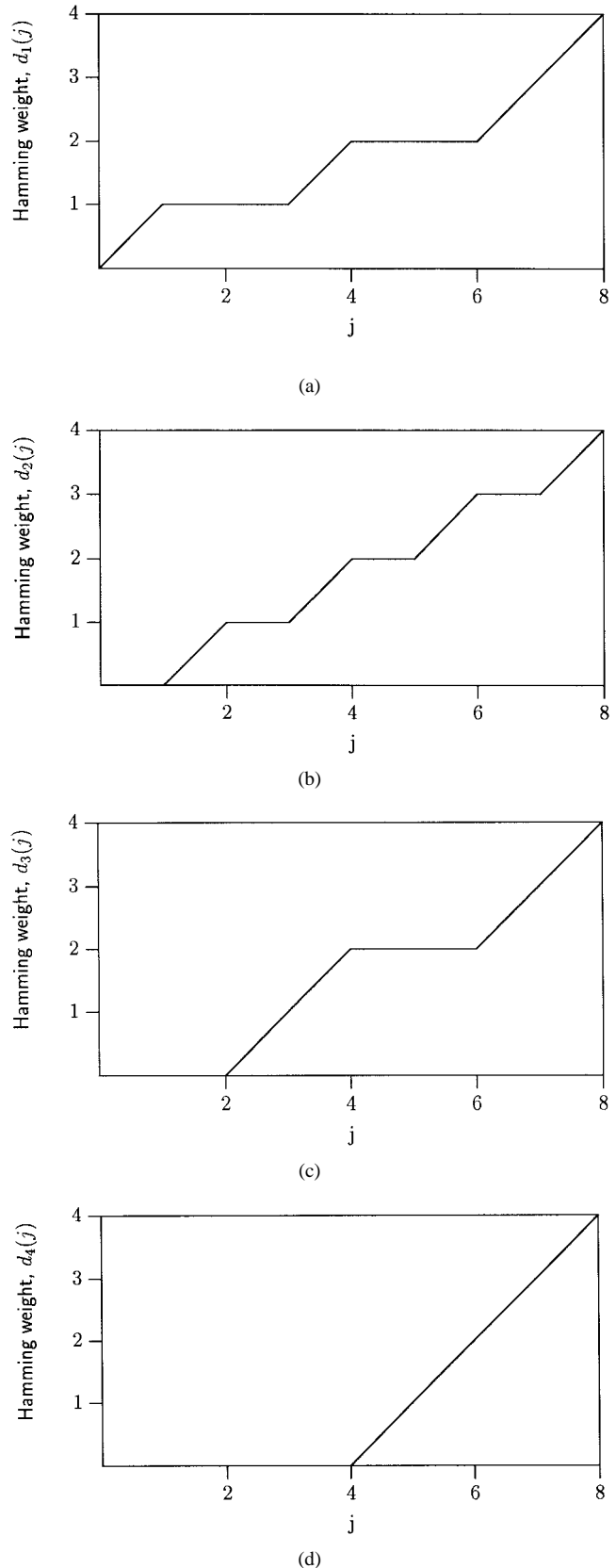


Fig. 2. (a) CDF<sub>1</sub>. (b) CDF<sub>2</sub>. (c) CDF<sub>3</sub>. (d) CDF<sub>4</sub> for the  $(8, 4, 4)$  RM code.

to minimize the computational complexity of the decoder one has to choose codes with fast growing CCDF. We can rephrase this result for block codes as follows. In order to minimize the computational

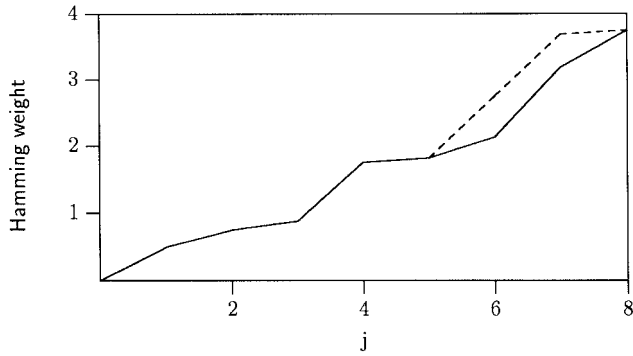


Fig. 3. CCDF (solid) and the upper bound (dashed) on the CCDF for the (8, 4, 4) RM code.

complexity of a sequential decoder for linear block codes one has to choose block codes with fast-growing CCDF. We conjecture that the minimization of the state space of the trellis diagram of a linear block code (for definitions see [9] and [10]) leads to a particular permutation that achieves fast growing CCDF's.

We now give a heuristic argument that the decoder with the VBT term metric is particularly suitable for codes with fast-growing CCDF. The growth of the CCDF influences the computational effort of the sequential decoder in a positive manner. Intuitively, the reason for this is the spatial "separation" of the code tree subsets. If the code tree subsets quickly become spatially separated after branching points of a code tree, then the decoder is less likely to switch into an incorrect subset later on in the decoding process. The degree of such separation is reflected by the growth of CCDF. A decoder with VBT metric takes a lower bound on the future cost and thus maintains a reduced list of contending paths compared, for example, with a decoder that uses the Fano metric. Since for codes with a fast-growing CCDF a sequential decoder is less likely to switch to an incorrect subset in general, such restriction of contending paths is advantageous. Simulation results (Fig. 7) show that even codes in systematic form perform well with the VBT metric. We explain this by the fact that after an initial flat portion, the CCDF grows quickly and the decoder has low computational complexity overall. For codes with a slow-growing CCDF the Fano metric is expected to outperform the VBT metric since the decoder now has to estimate the future cost over an ensemble of subsets of the code tree to reduce the computational complexity (see, for example, [17]).

Many well-known codes have a fast-growing CCDF. We consider the (8, 4, 4) Reed-Muller, (24, 12, 8) Golay, and (48, 24, 12) quadratic residue codes with standard orderings [8], [26], [27] which are known to have the best possible state complexity profile. In all three cases we observed fast growing CCDF's (see Figs. 3–5). For most linear block codes optimal orderings (i.e., permutations that minimize the trellis complexity) are unknown despite recent efforts in this direction [23], [27]–[32].

### B. An Upper Bound on the CCDF

We derive here an upper bound on the CCDF's for a particular  $(n, k, d)$  linear block code (with fixed ordering). Let us consider the first CCDF  $d_1(j)$  (this bound is applicable *mutatis mutandis* to any  $d_m(j)$ ). Here we assume the terminology of [9] and [10]. A generator of a block code is called *atomic* if it cannot be expressed as a sum of codewords with strictly smaller span lengths. (In the literature, the span of a codeword is also known as its *support*.) Observe that  $\overline{\text{start}}(C)$  is the (unique) set of indices where atomic generators start, i.e., have only zeros in the past (the past and the future are taken with respect to the current time index.) Denote by  $\text{end}(C)$  the (unique) set

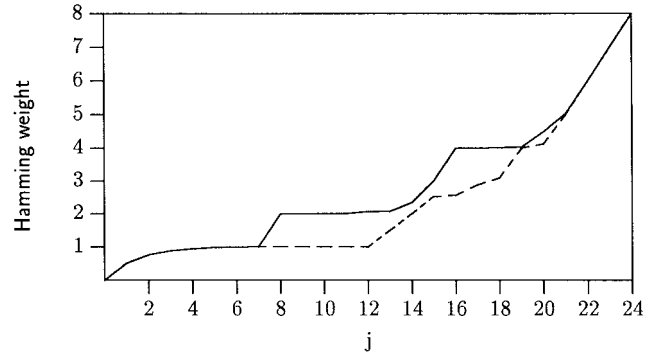


Fig. 4. CCDF's for the standard (solid) and systematic (dashed) orderings of the (24, 12, 8) Golay code.

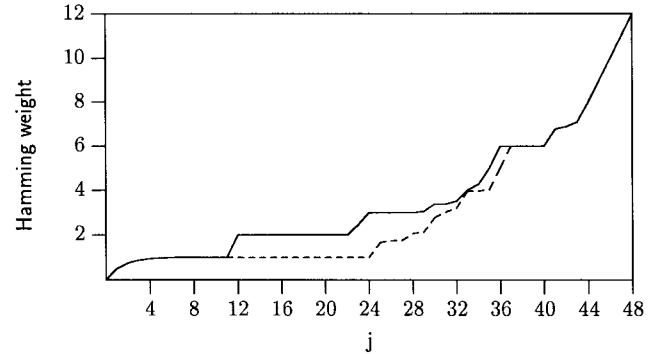


Fig. 5. CCDF's for the standard (solid) and systematic (dashed) orderings of the (48, 24, 12) quadratic residue code.

of indices where atomic generators end, i.e., have only zeros in the future. The set  $\overline{\text{start}}(C)$  is the set of indices where atomic generators for the given ordering of the block code do not start. A permutation of the block code generally changes  $\overline{\text{start}}(C)$ . For a given ordering of the code, this set is unique [10].

It follows from the definition that a CCDF cannot change where an atomic generator starts since each node has outgoing branches labeled with both 0 and 1 and so the minimum taken over the set of truncated codewords cannot increase at this position. In other words,

$$d_m(j+1) = d_m(j), \quad \text{if } j \in \text{start}(C).$$

We have  $\overline{\text{start}}(C) = \text{end}(C^\perp)$  where  $C^\perp$  is the dual code of  $C$  [10, Theorem 5]. Denote  $C_{j-}$  the  $j$ th past subcode, i.e., the subcode of  $C$  whose region of support is  $\{1, \dots, j\}$ . We have

$$d_1(j) \leq 1 + |\text{end}(C_{j-}^\perp)| \quad (9)$$

where  $|\text{end}(C_{j-}^\perp)|$  denotes the cardinality of the set  $\text{end}(C_{j-}^\perp)$  and 1 on the right-hand side due to the divergence of the codewords that determine this first CCDF, from the all-zero codeword in the first position. The second term on the right-hand side of (9) determines the dimension of the  $j$ th past subcode of the dual code  $C^\perp$ . So

$$d_1(j) \leq 1 + \dim C_{j-}^\perp$$

where  $\dim C_{j-}^\perp$  is the dimension of the  $j$ th past subcode of the dual code  $C^\perp$ . Then [10], [26]

$$\dim C_{j-}^\perp = j - (k - \dim C_{j+})$$

where  $k$  is the dimension of the code  $C$  and so

$$d_1(j) \leq 1 + j - k + \dim C_{j+}.$$

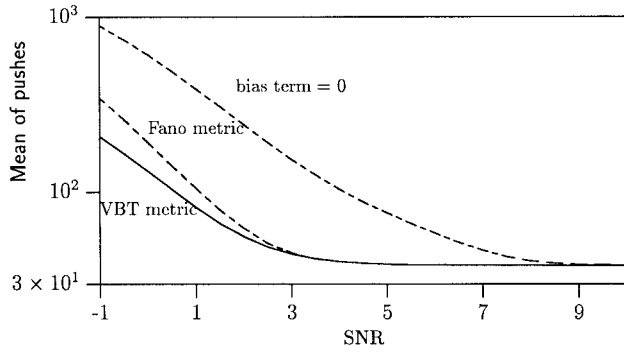


Fig. 6. Mean number of “pushes” into the stack for (24, 12) Golay code with standard ordering: VBT (solid), Fano (lower dashed), and zero bias term (upper dashed) metrics.

We can upper-bound  $\dim C_{j+}$  by noting that this subcode has the minimum distance no less than the original code. Let us denote by  $k_{\max}(n-j, d)$  the maximum dimension of a block code of length  $n-j$  and minimum distance  $d$ . Then our upper bound assumes the form

$$d_1(j) \leq 1 + j - k + k_{\max}(n-j, d).$$

Making appropriate changes we write the upper bound on the  $m$ th column distance function in the general form

$$d_m(j) \leq 1 + (j - i_m + 1) - (k - m + 1) + k_{\max}(n-j, d)$$

where  $j \geq i_m$ . From these one can also easily obtain an upper bound on the CCDF given the start set of the code.

It is shown in [26] that minimizing the trellis state complexity generally tightens the bound  $\dim C_{j+} \leq k_{\max}(n-j, d)$ . For some codes such as the (8, 4, 4) Reed–Muller and (24, 12, 8) Golay codes, uniformly optimal orderings are known. Brouwer and Verhoeff in [33] give bounds on the minimum distance of binary codes as a function of  $(n, k)$ . We used this table to compute bounds on CCDF’s for the (8, 4, 4) RM code. The CCDF and the upper bound for the (8, 4, 4) RM code are shown in Fig. 3. One can see that the standard ordering of the (8, 4, 4) RM code is close to the upper bound in terms of the CCDF growth.

It is also possible to derive a lower bound on CCDF using the result of [34]. The runs of zeros (that “slow down” the growth of CCDF’s) are bounded by the set of atomic spans in the dual code. This lower bound is, however, quite loose and we do not consider it in the present correspondence.

#### IV. SIMULATION RESULTS

The performance of the sequential decoder for the (24, 12, 8) Golay and (48, 24, 12) quadratic residue codes has been simulated. We were primarily interested in the comparative performance of the VBT and the Fano metrics, and the performance of these codes with optimal and standard orderings.

There is no universally accepted measures of the complexity of the decoder. One may use the number of additions and comparisons performed by the decoder as one such measure. We try to avoid this ambiguity by counting the number of nodes we extend. There is a fixed cost associated with extending a node. It consists of computing the branch labels, computing state labels (see [11]), and placing the node into the stack. The cost of placing the node into the stack may be relatively high if implemented in software or low if implemented in hardware [35]. The decoder also performs the extraction of the top node in the stack. The cost associated with this operation is also fixed and usually is only a fraction of the cost of inserting an

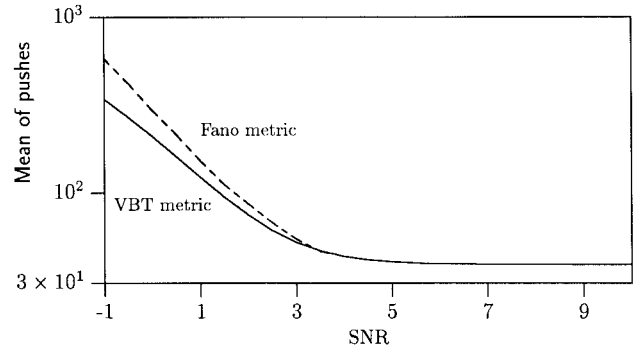


Fig. 7. Mean number of “pushes” into the stack for (24, 12) Golay code with systematic ordering: VBT (solid) and Fano (dashed) metrics.

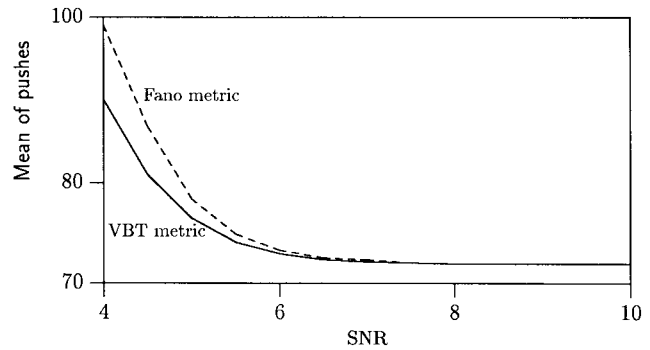


Fig. 8. Mean number of “pushes” into the stack for (48, 24) QR code with standard ordering: VBT (solid) and Fano (dashed) metrics.

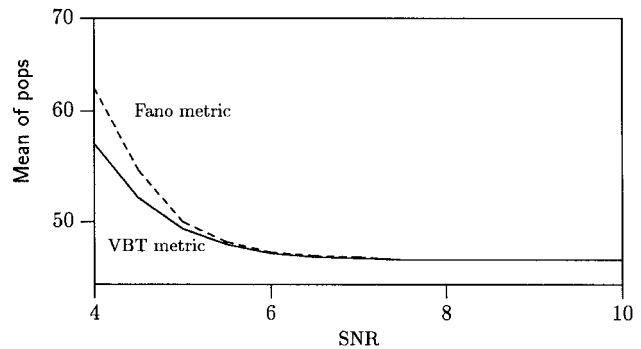


Fig. 9. Mean number of “pops” from the stack for (48, 24) QR code with standard ordering: VBT (solid) and Fano (dashed) metrics.

extended node into the stack. In Fig. 6 we show the average number of “pushes” performed by the decoder for the (24, 12, 8) Golay code with the standard ordering. In Fig. 7 the (24, 12, 8) Golay code is in systematic form and, as we expected, the computational complexity of the decoder increases. The VBT metric still performs better than the Fano metric for low SNR values and performs the same for higher SNR values. For the (48, 24, 12) quadratic residue code we show relative numbers of “pushes” and “pops” of the stack in Figs. 8 and 9. One can easily compute the actual average cost (i.e., the number of additions and comparisons) of the decoder from this data depending on a particular implementation.

In the model we assumed continuous additive white Gaussian noise (AWGN) channel characterized by a certain symbol signal-to-noise ratio (SNR in Figs. 6–9). We ran  $10^6$  simulation trials for the (24, 12, 8) Golay code and  $5 \times 10^5$  simulation trials for the (48, 24, 12) quadratic residue code. It turns out that the distribution

of computations in the sequential decoding of the block codes is approximately Pareto [11] (but always bounded from above by the cost of opening all nodes in the code tree; hence the approximation). A large number of simulation trials is required for computing the mean of the number of "pushes," since the standard deviation can be quite large especially for low SNR. We consider the average of the number of "pushes" and "pops" as a primary criterion of the complexity of the sequential decoder. This is automatically justified for the medium to large values of SNR (3–10 dB) since the standard deviation in this range is relatively small. For low SNR one would most likely use some algorithm enhancements such as those described in [11].

As illustrated in Figs. 6 and 7, the VBT metric provides about 40% computational savings compared with the Fano metric in the low and medium SNR range, and an order of magnitude of computational savings compared with the zero bias-term metric.

## V. CONCLUSIONS

The VBT metric is shown to be a good choice of metric for sequential decoding of linear block codes. The lower bound on the path-cost estimate used in the VBT metric is trivial to compute and does not depend on the ordering of the code. The decoder with the VBT metric significantly reduces the computational effort of the sequential decoder while preserving its maximum-likelihood character. Analysis of the sequential decoder for block codes is similar to that for convolutional codes. The single most important parameter that determines the computational complexity of the decoder for a particular block code is the cumulative column distance function (CCDF) herein defined.

Block codes with the fastest growing CCDF provide significant computational savings for sequential decoders. Optimal orderings are presently known only for a few block codes and we consider the search for such orderings as an interesting future research problem.

Sequential decoding schemes have some drawbacks (such as variable decoding effort) that are well known in the convolutional decoding context. Since block codes have a finite tree, the average number of computations and the standard deviation are always bounded. (In practical implementations of sequential decoders for convolutional codes the code tree is also of finite length thus making effectively a convolutional code into a block code.) However, the decoding effort still varies widely, especially in the low SNR range. Some work-arounds such as time-out mechanisms are known and we refer the reader to the relevant literature (see, for example, [4, ch. 12]).

For some applications requiring a block-oriented error-control scheme, sequential decoders present a viable means of implementing decoders for relatively powerful codes. The VBT metric can be easily adapted to most signaling schemes. We consider this flexibility as one of the most important advantages in designing sequential decoders for arbitrary linear block codes, perhaps with nonbinary alphabets.

## REFERENCES

- [1] R. M. Fano, "A heuristic discussion of probabilistic decoding," *IEEE Trans. Inform. Theory*, vol. IT-9, pp. 64–74, Apr. 1963.
- [2] J. L. Massey, "Variable-length codes and the Fano metric," *IEEE Trans. Inform. Theory*, vol. IT-18, pp. 196–198, Jan. 1972.
- [3] I. M. Jacobs and E. R. Berlekamp, "A lower bound to the distribution of computation for sequential decoding," *IEEE Trans. Inform. Theory*, vol. IT-13, pp. 167–174, Apr. 1967.
- [4] S. Lin and D. J. Costello, Jr., *Error Control Coding: Fundamentals and Applications*. Englewood Cliffs, NJ: Prentice-Hall, 1983.
- [5] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 284–287, 1974.
- [6] J. K. Wolf, "Efficient maximum-likelihood decoding of linear block codes using a trellis," *IEEE Trans. Inform. Theory*, vol. IT-24, pp. 76–80, 1978.
- [7] J. L. Massey, "Foundation and methods of channel encoding," in *Proc. Int. Conf. on Information Theory and Systems* (Berlin, Sept. 1978), vol. 65.
- [8] G. D. Forney, Jr., "Coset codes II: Binary lattices and related codes," *IEEE Trans. Inform. Theory*, vol. 34, pp. 1152–1187, Sept. 1988.
- [9] G. D. Forney, Jr., and M. D. Trott, "The dynamics of group codes: State spaces, trellis diagrams and canonical encoders," *IEEE Trans. Inform. Theory*, vol. 39, pp. 1491–1513, Sept. 1993.
- [10] F. R. Kschischang and V. Sorokine, "On the trellis structure of block codes," *IEEE Trans. Inform. Theory*, vol. 41, pp. 1924–1937, Nov. 1995.
- [11] V. Sorokine, F. R. Kschischang, and V. Durand, "Trellis-based decoding of binary linear block codes," in *Lecture Notes in Computer Science*, vol. 793. Berlin, Germany: Springer-Verlag, 1994, pp. 270–286.
- [12] T. Fujiwara, T. Kasami, R. Morelos-Zaragoza, and S. Lin, "The state complexity of trellis diagrams for a class of generalized concatenated codes," preprint.
- [13] K. S. Zigangirov, "Some sequential decoding procedures," *Probl. Pered. Inform.*, vol. 2, pp. 13–25, 1966.
- [14] F. Jelinek, "A fast sequential decoding algorithm using a stack," *IBM J. Res. Devel.*, vol. 13, pp. 675–685, Nov. 1969.
- [15] J. M. Wozencraft and B. Reiffen, *Sequential Decoding*. Boston, MA: MIT Press and New York: Wiley, 1961.
- [16] D. J. Tempel, "Sequential decoding of linear block codes," Master's thesis, Dept. Elec. Comput. Eng., Univ. of Manitoba, Winnipeg, Canada, 1993.
- [17] G. Krämer and D. J. Tempel, "An alternate metric for sequential decoding," in *Proc. EIDMA Winter Meet. on Coding Theory, Information, Theory and Cryptology*, 1994.
- [18] Y. S. Han, C. R. P. Hartmann, and C. C. Chen, "Efficient priority-first search maximum-likelihood soft-decision decoding of linear block codes," *IEEE Trans. Inform. Theory*, vol. 39, pp. 1514–1523, 1993.
- [19] N. J. Nilsson, *Principles of Artificial Intelligence*. Palo Alto, CA: Tioga, 1980.
- [20] Z. Xie, C. Rushforth, and R. Short, "Multiuser signal detection using sequential decoding," *IEEE Trans. Commun.*, vol. 38, pp. 578–583, 1990.
- [21] L. Ekroot and S. Dolinar, "A\* decoding of block codes," *IEEE Trans. Commun.*, vol. 44, pp. 1052–1057, Sept. 1996.
- [22] O. Collins, "Coding beyond the computational cutoff rate," Ph.D. dissertation, Calif. Inst. Technol., Pasadena, CA, 1989.
- [23] T. Kasami, T. Takata, T. Fujiwara, and S. Lin, "On the optimum bit orders with respect to the state complexity of trellis diagrams for binary linear codes," *IEEE Trans. Inform. Theory*, vol. 39, pp. 242–245, Jan. 1993.
- [24] P. R. Chevillat and J. D. Costello, Jr., "An analysis of sequential decoding for specific time-invariant convolutional codes," *IEEE Trans. Inform. Theory*, vol. IT-24, pp. 443–451, July 1978.
- [25] J. L. Massey, M. K. Sain, and J. M. Geist, "Certain infinite Markov chains and sequential decoding," *Discr. Math.*, vol. 3, pp. 163–175, Sept. 1972.
- [26] G. D. Forney, Jr., "Dimension/length profiles and trellis complexity of linear block codes," *IEEE Trans. Inform. Theory*, vol. 40, pp. 1741–1752, Nov. 1994.
- [27] S. Dolinar, L. Ekroot, A. Kiely, W. Lin, and R. J. McEliece, "The permutation complexity of linear block codes," in *Proc. 32nd Annual Allerton Conf. on Communications, Control and Computing* (Monticello, IL, Sept. 1994).
- [28] Y. Berger and Y. Be'ery, "Bounds on the trellis size of linear block codes," *IEEE Trans. Inform. Theory*, vol. 39, pp. 203–209, Jan. 1993.
- [29] T. Kasami, T. Takata, T. Fujiwara, and S. Lin, "On complexity of trellis structure of linear block codes," *IEEE Trans. Inform. Theory*, vol. 39, pp. 1057–1064, May 1993.
- [30] T. Takata, Y. Yamashita, T. Fujiwara, T. Kasami, and S. Lin, "On a sub-optimum decoding of decomposable block codes," in *Coded Modulation and Bandwidth-Efficient Transmission*, E. Biglieri and M. Luise, Eds. Amsterdam, The Netherlands: Elsevier, 1992, pp. 201–212.
- [31] A. D. Kot and C. Leung, "On the construction and dimensionality of linear block code trellises," in *Proc. 1993 IEEE Int. Symp. on Information Theory*, 1993, p. 291.
- [32] F. R. Kschischang and G. B. Horn, "A heuristic for ordering a linear block code to minimize trellis state complexity," in *Proc. 32nd Annual Allerton Conf. on Communications, Control, and Computing*, (Allerton Park, IL, Sept. 1994).
- [33] A. E. Brouwer and T. Verhoeff, "An updated table of minimum-distance

- bounds for binary linear codes," *IEEE Trans. Inform. Theory*, vol. 39, pp. 662–677, Mar. 1993.
- [34] G. D. Forney, Jr., "Structural analysis of convolutional codes via dual codes," *IEEE Trans. Inform. Theory*, vol. IT-19, pp. 512–518, July 1973.
- [35] P. Lavoie, D. Haccoun, and Y. Savaria, "A systolic architecture for fast stack sequential decoders," *IEEE Trans. Commun.*, vol. 42, pp. 324–335, Feb./Mar./Apr. 1994.

## Codes Correcting Phased Burst Erasures

Osnat Keren, *Member, IEEE*, and Simon Litsyn, *Member, IEEE*

**Abstract**—We introduce a family of binary array codes of size  $t \times n$ , correcting multiple phased burst erasures of size  $t$ . The codes achieve maximal correcting capability, i.e., being considered as codes over  $\text{GF}(2^t)$  they are MDS. The length of the codes is

$$n = \sum_{l=1}^L \binom{t}{l}$$

where  $L$  is a constant or is slowly growing in  $t$ . The complexity of encoding and decoding is proportional to  $nmL$ , where  $r$  is the number of correctable erasures, and  $m$  is the smallest number such that  $2^t = 1$  modulo  $m$ . This compares favorably with the complexity of decoding codes obtained from the shortened Reed–Solomon codes having the same parameters.

**Index Terms**—Array codes, complexity of decoding, decoding erasures, Reed–Solomon codes.

### I. INTRODUCTION

Let  $t \times k$  bits of information be encoded in a  $t \times n$ -bit array,  $n > k$ . Due to some reasons, the data stored in several columns can be lost or corrupted. We assume that we know in which columns it has happened. The data they carry is said to be erased. Such erasures are also referred to as phased burst erasures. Our purpose now is to reconstruct the missing data. Problems of this type arise, for example, in storage systems and information transmission over parallel channels, see [4], [5], [8], [10], [15], [17], and many references therein. Sometimes erasures may be deliberately declared for the sake of increasing performance. For example, if the  $n$  pieces of information do not arrive simultaneously, it is possible to reconstruct all the data from the partial information in our possession, while considering the remaining pieces as erased. Such approach proves to be efficient in RAID systems [7] and networks with packet transmission [2], and [13].

A binary array code  $C(t, n, r)$  corrects up to  $r$  phased burst erasures, i.e., reconstructs the correct codeword when at most  $r$  columns have been erased. Clearly, the problem of constructing such a code is equivalent to finding a code over  $\text{GF}(2^t)$  with minimum distance  $r + 1$ . The maximal error-correcting capability of such codes is achieved if they are maximum-distance separable (MDS), i.e., they correct up to  $n - k$  erasures [12].

A binary  $C(t, n, r)$  code for  $n \leq 2^t - 1$  can be obtained by shortening a Reed–Solomon (RS) code over  $\text{GF}(2^t)$ . A codeword

Manuscript received April 3, 1996; revised July 1, 1997.

The authors are with the Department of Electrical Engineering—Systems, Tel-Aviv University, Ramat-Aviv 69978, Tel-Aviv, Israel.

Publisher Item Identifier S 0018-9448(98)00069-8.

can be regarded as a vector of  $n$  symbols belonging to  $\text{GF}(2^t)$ , where each symbol is a  $t$ -bit column. With slight abuse of terminology, we say that  $n$  is the length of the array code.

The conventional decoding procedure consists of calculating the syndrome followed by computing the values of erasures. This can be done using the Forney algorithm [9] (actually, this algorithm was suggested earlier by Parker [14] for the purpose of inverting Vandermonde matrices). All the operations in the algorithm are implemented in  $\text{GF}(2^t)$  and its complexity is proportional to  $rn$  field operations or  $rnt \log t \log \log t$ -bit operations [3]. For small  $r$ , the best known algorithms for calculation of the syndrome require  $rnt$ -bit operations. So, the question is if it is possible to design codes achieving the maximum possible erasure-correcting capability, and having complexity of decoding proportional to  $rnt$ . A class of such codes was constructed by Blaum and Roth [4] (for earlier results see [5], [10], [18], and references therein). Having complexity of decoding proportional to  $rnt$ , the Blaum–Roth codes are of length at most  $t + 1$ , and  $t$  is of the form  $p - 1$  for some prime  $p$ . In our paper [11], we proposed another class of codes of length  $2t + 1$  with the same complexity. However, these constraints on the code parameters are quite restrictive.

Here we propose a new class of  $C(t, n, r)$  array codes with maximal erasure-correcting capability. The length of the codes is

$$n \leq \sum_{l=1}^L \binom{t}{l}$$

for some  $L \leq t$ . If  $L$  is chosen to be constant independent of  $t$ , the codes have length proportional to  $t^L$ . Let  $m$  be the minimal number such that  $m > t$ ,  $m|2^t - 1$ . Then, complexity of decoding the proposed codes is of order  $nmL$ , i.e., if  $m$  is close to  $t$  and  $L$  is constant, it is proportional to  $rnt$ . If  $L < \log t \log \log t$ , the decoding complexity is still better than for general shortened RS codes. If  $L = 1$  and  $m$  is prime, then  $t$  equals  $m - 1$  and our construction coincides with the one of Blaum–Roth for  $n \leq t$ .

The basic idea is that we use a shortened RS code over the field  $\text{GF}(2^t)$  defined by a *nonprimitive* polynomial  $M(x)$ , having a root  $\alpha$  of order  $m > t$ . It is desirable to have  $m$  as close as possible to  $t$ . Considering the elements of the field as polynomials in  $\alpha$ , we pick only those columns of the parity-check matrix of the RS code that correspond to the elements with at most  $L$  nonzero coefficients. The essential simplification is achieved by implementing the computations in the ring defined modulo the polynomial  $x^m - 1$  where multiplication by a power of  $\alpha$  turns out to be a cyclic shift.

The correspondence is organized as follows. In Section II, we discuss relations between the initial field and the corresponding ring. We define the code by its parity-check matrix and prove that it achieves the maximal correcting capability. In the next section, we give an example of decoding the  $C(20, 210, 2)$ -bit array code. In Section IV, we describe the decoding scheme for multiple erasures.

### II. CONSTRUCTION

Let the field  $F = \text{GF}(2^t)$  be defined modulo a *nonprimitive* irreducible polynomial  $M(x)$  of degree  $t$ . The elements of  $F$  are polynomials over  $\text{GF}(2)$  of degree at most  $t - 1$ . Denote by  $\alpha$  a root of  $M(x)$ . Since  $M(x)$  is not a primitive polynomial,  $\alpha$  has order  $m$ , where  $m > t$  is a factor of  $2^t - 1$ , i.e.,  $2^t = 1$  modulo  $m$ . Hence,  $\alpha$  is also a root of  $x^m - 1 = M(x)g(x)$ . Now, we may define the enveloping polynomial ring  $R$  modulo  $x^m - 1$ . The elements of  $R$  are polynomials over  $\text{GF}(2)$  of degree less than  $m$ . To avoid confusion