Journal of ELECTRICAL ENGINEERING, VOL. 63, NO. 1, 2012, 59–64

SOFT DECISION FANO DECODING OF BLOCK CODES OVER DISCRETE MEMORYLESS CHANNEL USING TREE DIAGRAM

H. Prashantha Kumar * — **Udupi Sripati** *
K. Rajesh Shetty * — **B. Setty Shankarananda** **

A novel low complexity soft decision technique which allows the decoding of block codes with tree structure is proposed. These codes are shown to have a convenient tree structure that allows Fano decoding techniques to be used to decode them. The Fano algorithm searches through the tree structure of the block code for a path which has the optimal value of the Fano metric function. When a new candidate codeword is found, an optimality check is performed on it by using the threshold. If checked successfully, the candidate codeword is the most likely codeword and the search stops. The basic idea of this approach is to achieve a good error performance progressively in a minimum number of steps. For each decoding step, the error performance is tightly bounded and the decoding is terminated at the stage where either optimum or near optimum error performance is achieved. As a result, more flexibility in the trade off between performance and decoding complexity is provided. Some examples of the tree construction and the soft decision Fano decoding procedure are discussed.

K e y w o r d s: soft decision decoding, extended Hamming code, tree diagram, Fano algorithm, threshold selection

1 INTRODUCTION

Error control codes enable a decoder to recover from errors produced by noise in a communication channel. Error control coding (ECC) algorithms have constituted a significant enabler in the telecommunications revolution, the internet, digital recording and space exploration. ECC is nearly ubiquitous in modern, information based society. The last decade has been characterized not only by an exceptional increase in data transmission and storage but also by a rapid development in microelectronics, providing us with both a need for and the possibility of implementing sophisticated algorithms for error control [1]. According to the manner in which redundancy is added to messages, ECC can be divided into two classes: block and convolutional. Block codes implement a one-to-one mapping of a set of k information symbols on to a set of n code symbols. Convolutional codes offer an approach to error control substantially different from that of block codes. A convolutional encoder converts the entire data stream, regardless of its length, into a single codeword. Both types of coding schemes have found practical applications. Historically convolutional codes have been preferred, apparently because of the availability of soft decision decoding (SDD) algorithm and the belief over many years that block codes could not be efficiently decoded with soft decisions. The main problem is the fundamentally algebraic structure of block codes. Although this structure allows elegant algebraic decoding techniques to be applied when hard decisions are made, the reliance on finite field arithmetic for decoding makes it difficult to exploit soft decisions. One of the most general approaches is that of Wolf [2], who showed that any linear block codes can be represented by a trellis, and that the Viterbi algorithm can therefore be used for soft decision decoding of block codes. For example, a $(7, 4)$ Hamming code is represented by the parity check matrix

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Its minimal trellis representation based on above parity check matrix is shown in Fig. 1. It is interesting to note that there is no need to label the branches with the coded bits. A transition between two states with the same level corresponds to coded bit 0. Linear block codes have trellises with a time-varying number of states. The minimum number of states can be quite large, for example, 2^{64} for the $(128, 64)$ extended BCH code [3]. Although a certain permutation of the code achieves 2^{43} states, which is still exceedingly large for practical implementations of the Viterbi or Bahl-Cocke-Jelinek-Raviv (BCJR) algorithm [4]. In spite of exponential increase in computational complexity, the soft decision decoding using trellis diagram performs 2 to 3 dB better than hard decision decoding (HDD) over additive white Gaussian noise (AWGN) channel. This much amount of coding gain is very significant. 3 dB of coding gain can reduce the required bandwidth by 50 % or increase data throughput by a factor of 2 or increase range by 40 % or reduce antenna size by 30 % or reduce transmitter power by a factor of 2. Therefore collectively we can say that coding gain increases the system performance or reduces cost or both [5]. SDD increases the error correcting capability of the code by correcting a number of soft errors. This yields an increase in the coding gain compared to HDD.

* Department of Electronics and Communication Engineering, National Institute of Technology Karnataka, India, hprashanthakumar@gmail.com, sripati_acharya@yahoo.co.in, krshetty_nitte@yahoo.co.in;

** Department of Electronics and Communication, Gopalan College of Engineering, Bangalore, India, bsnanda2000@yahoo.co.uk

DOI: 10.2478/v10187-012-0009-y, ISSN 1335-3632

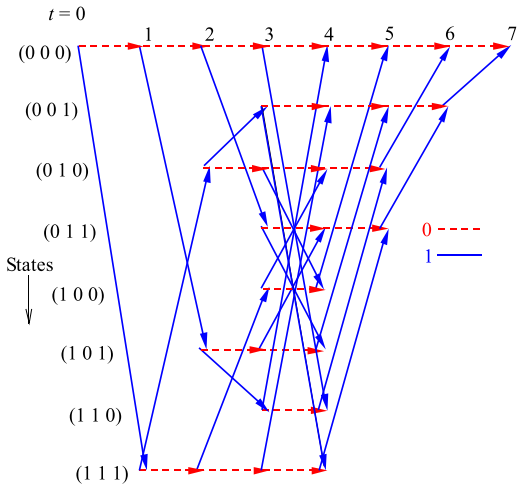

Fig. 1. Trellis representation of a $(7, 4)$ Hamming code

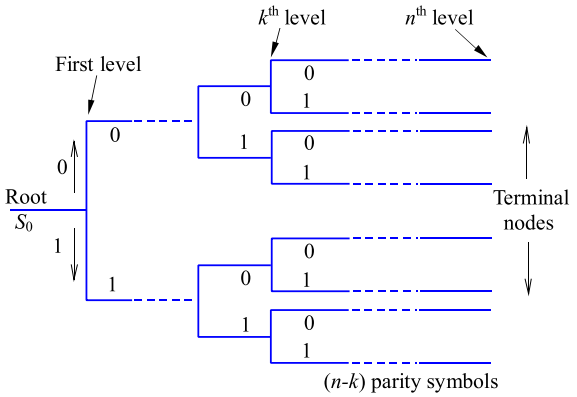

Fig. 2. Tree representation of a binary (n, k) systematic block code

Many good block codes are presently known. Several of them have been used in applications ranging from deep space communication to error control in storage systems. But the primary difficulty with Viterbi and BCJR decoding of block codes is that, even though they are optimum decoding methods, the promised bit error rates are not achieved in practice at rates close to capacity. This is because the decoding effort is fixed and grows with block length, and thus only short block length codes can be used [6]. Therefore, an important practical question is whether a suboptimal realizable soft decision decoding method can be found for block codes. A noteworthy result based on this question is described in the following section. This result of suboptimal decoding will be used as motivation for the investigation of different soft decision decoding methods for linear block codes and also for motivation for the development of efficient decoding algorithms.

2 TREE REPRESENTATION OF SYSTEMATIC LINEAR BLOCK CODES

We begin by noting that the fixed amount of computation required by the Viterbi algorithm is not always needed, particularly when the noise is light (or high signal to noise ratio) [7]. For example, assume that an (n, k) linear block code is transmitted without error over a channel. The Viterbi algorithm will still perform on the order of $2^{\min\{k,n-k\}}$ computations per decoded information block, all of which is wasted effort in this case. In other words, it is sometimes desirable to have a decoding procedure whose effort is adaptable to the noise level. Sequential decoding using tree diagram is such a type of algorithm. Sequential decoding describes any algorithm for decoding channel codes which successively explores the code tree by moving to new nodes from an already explored node. The purpose of tree searching algorithms is to search through the nodes of the code tree in efficient way, that is, without having to examine too many nodes, in an attempt to find the maximum likelihood path. Each node examined represents a path through part of the tree. Whether a particular path is likely to be part of the maximum likelihood path depends on the metric value associated with that path. The metric is a measure of the closeness of a path to the received sequence [8].

Every linear block code can be represented graphically by means of a tree. The code tree can be treated as an expanded version of the trellis, where every path is totally distinct from every other path. Figure 2 represents general tree representation for an (n, k) systematic linear block code.

This tree has the following structures [9]:

1. Tree consists of $n + 1$ levels.
2. For $0 \leq i < k$, there are 2^i nodes at the i^{th} level of the tree. There is only one node s_0 at the zeroth level of the tree called the initial node (or root) of the tree, and there are 2^k nodes at the n^{th} level of the tree, which are called the terminal node of the tree.
3. For $0 \leq i < k$, there are two branches leaving every node s_i at level-i and connecting to two different nodes at level-$(i + 1)$. One branch is labeled with an information symbol 0, and the other branch is labeled with an information symbol 1. For $k \leq i \leq n$, there is only one branch leaving every node s_i at level-i and connecting to one node at level-$(i+1)$. This branch is labeled with a parity check symbol 0 or 1.
4. The label sequence of path connecting the initial node s_0 to a node s_k at the k^{th} level corresponds to an information sequence $\boldsymbol{m}$ of k bits. The label sequence of the path connecting the initial node s_0 through a node s_k at the k^{th} level to a terminal node s_n at the n^{th} level is a codeword $\mathbf{C}$. The label sequence of the tail connecting node s_k to node s_n corresponds to the $n - k$ parity check symbols of the codeword.

The generator matrix $\mathbf{G}$ of an $(8, 4)$ extended Hamming code with minimum Hamming distance $\mathrm{d}_{\min} = 4$ is given below. This matrix is represented in systematic form by performing Gaussian elimination with pivoting

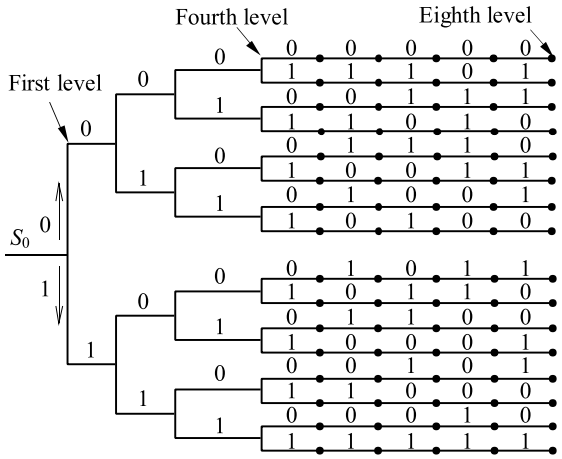

Fig. 3. Tree representation of a binary $(8, 4)$ extended Hamming code

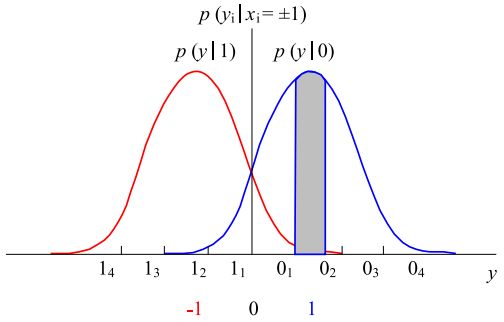

Fig. 4. Probability distribution function for received symbol y_i

over non systematic form.

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}.$$

The above generator matrix generates all possible valid codewords in systematic form. Figure 3 shows the tree representation of a binary extended Hamming code generated by $\mathbf{G}$.

3 DECODING WITH THE FANO ALGORITHM

The tree representation of a linear block code can be used to facilitate Fano decoding. The Fano algorithm compares paths of differing length when deciding whether to continue through the tree or to back up and try a different branch without having to examine too many nodes in an attempt to find the maximum likelihood path. The decoder examines a sequence of nodes in the tree, starting with the origin (or root) node and ending with one of the terminal nodes. The decoder never jumps from node to node, as in the stack algorithm [10], but always moves to an adjacent node, either forward to one of the 2^k nodes leaving the present node, or backward to the node leading to the present node. The metric of the next node to be examined can then be computed by adding (or subtracting) the metric of the connecting branch to the metric of the present node. This process eliminates the need for storing the metrics of previously examined nodes, as required by the stack algorithm, however, some nodes are visited more than once, and in this case their metric values must be recomputed. The decoder moves forward through the tree as long as the metric value along the path being examined continues to increase. When the metric value dips below a threshold the decoder backs up and begins to examine other paths. If no path can be found whose metric value stays above the threshold, the threshold is then lowered, and the decoder attempts to move forward again with a lower threshold. Each time a given node is visited in the forward direction, the threshold is lower than on the previous visit to that node. This prevents looping in the algorithm, and the decoder eventually must reach the end of the tree is taken as the decoded path [11].

Fano algorithm commonly uses a probabilistic branch metric, namely, the Fano metric, which can be written for a continuous (or Gaussian) channel as [3]

$$M\big(r_l|v_l\big) = -\log_2\Big[1+\exp\Big(-4\frac{(2v_l-1)r_l\sqrt{E_b}}{N_0}\Big)\Big]. \quad (1)$$

where $M\big(r_l|v_l\big)$ is the branch metric for the l^{th} branch, E_b is the energy per transmitted bit and N_0 is the one-sided noise power density. For a discrete memoryless channel (DMC) with a uniformly distributed source and a crossover probability p, the above Fano metric reduces to

$$M\big(r_l|v_l\big) = \log_2 p\big(r_l|v_l\big) - \log_2 p(r_l) - R\,. \quad (2)$$

Here, R is the rate of the code in use, $p\big(r_l|v_l\big)$ is the channel transition probability of the received symbol r_l given the transmitted symbol v_l, $p(r_l)$ is a channel output symbol probability. Fanos original selection of this metric was based on a heuristic argument, and on occasion other researchers/designers have used other metrics [12].

We assume a binary phase shift keying (BPSK) modulation, where the bits $c_i \in \{0, 1\}$ are mapped to the transmission bits $x_i \in \{+1, -1\}$ corresponding to the relation

$$x_i = (-1)^{C_i}\,;\; i \in [1, n]\,. \quad (3)$$

After transmission over the AWGN channel, we obtain the probability distribution depicted in Fig. 4.

We assume that the y-axis in above figure is divided in to intervals of width δy. In practical systems, this value is often quantized. In our decoder analysis, the received signal is quantized to 3 bits, resulting in 2^3 different quantization levels, using uniformly spaced quantization thresholds [13, 14]. The block interprets 0_4 as the most confident decision that the codeword bit is a 0 and interprets 1_4 as the most confident decision that the codeword bit is a 1. The values in between these represent less confident decisions. Thus a binary input, continuous valued output has changed to 8-ary DMC. Figure 5 shows 8-level soft quantized DMC.

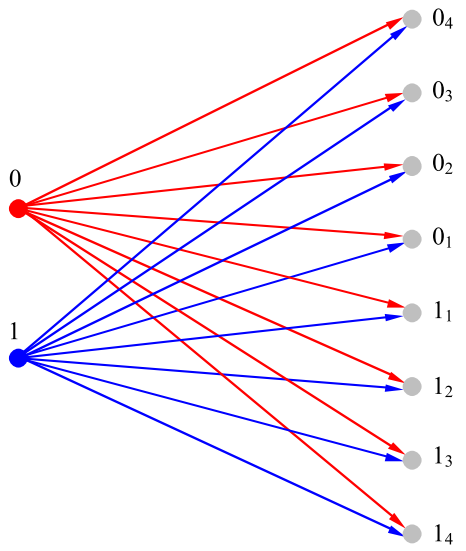

Fig. 5. Binary input 8-ary output DMC

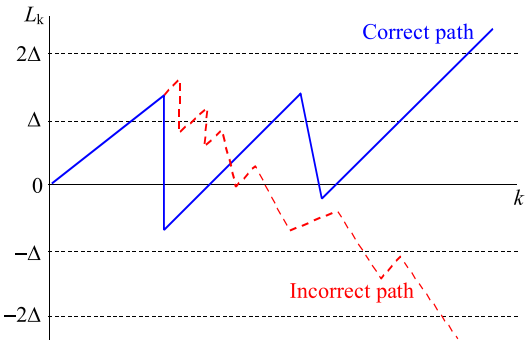

Fig. 6. Typical metric behavior of correct and incorrect paths

We now proceed to a description of the Fano decoding algorithm by means of a set of rules for calculating the threshold and for backward or forward translations.

T ⟶ Threshold value maintained by the algorithm
Δ ⟶ Threshold increment
M_F ⟶ Path metric of the forward node or next node
M_B ⟶ Path metric of the backward node
$M_F \geq T \implies P^+$ ⟶ Move forward
$M_B \geq T \implies P^?$ ⟶ Move backward
$M_F < T \implies$ Visit the previous node
$M_B < T \implies T = T - \Delta$

The path metric at the root node is set at $M_B = -\infty$.

EXAMPLE. A binary $(8, 4)$ extended Hamming code associated with the tree in Fig. 3 is used to encode the information sequence $\boldsymbol{x} = (0\ 0\ 0\ 0)$, resulting in the codeword

$$\boldsymbol{v} = 0\,0\,0\,0\,0\,0\,0\,0$$

The codeword is transmitted over the binary input, 8-ary output DMC with transition probabilities $p(r|v)$ given by the below table [15]

v	r							
	0_4	0_3	0_2	0_1	1_1	1_2	1_3	1_4
0	0.1402	0.3203	0.2864	0.166	0.0671	0.177	0.0024	0.0001
1	0.0001	0.0024	0.177	0.0671	0.166	0.2864	0.3203	0.1402

The sequence $\boldsymbol{r} = 0_4\mathbf{1}_2 0_4\mathbf{1}_1 0_4\mathbf{1}_1 0_4 0_4$ is received. Error bits (2^{nd}, 4^{th}, and 6^{th} position) are shown in dark. Using (2), the Fano metric is computed as follows.

v	r							
	0_4	0_3	0_2	0_1	1_1	1_2	1_3	1_4
0	0.499	0.4892	0.4134	0.0103	-1.2965	-3.6027	-6.571	-9.9543
1	-9.9543	-6.571	-3.6027	-1.2965	0.0103	0.4134	0.4892	0.499

The metrics are scaled by 50/0.499 to obtain integer metrics as shown below.

v	r							
	0_4	0_3	0_2	0_1	1_1	1_2	1_3	1_4
0	50	49	41	1	-130	-361	-658	-997
1	-997	-658	-361	-130	1	41	49	50

Some of the possible threshold increments Δ is calculated from the relation

$$\Delta = \left|M(1_1) + M(1_2)\right| \approx 500\,, \tag{4}$$

$$\Delta = \left|\frac{M(0_4) + \cdots + M(0_1) + M(1_1) + \cdots + M(1_4)}{2}\right| \approx 1000\,, \tag{5}$$

$$\Delta = \left|\frac{M(1_1) + M(1_3)}{2}\right| \approx 400\,, \tag{6}$$

$$\Delta = \left|M(1_1) + M(1_2) + M(1_3) + M(1_4)\right| \approx 2150\,, \tag{7}$$

$$\Delta = \left|M(0_4) + M(0_3)\right| \approx 100\,. \tag{8}$$

For example, from (4) we calculated $\Delta \approx 500$ and decoding proceeds as follows. The position P of the decoder is denoted by the binary input sequence corresponding to the current node in the tree. X is used to represent the root. We can note the movement of the decoder back and forth through the tree by following the variation in the length of the representation for the decoder position. The results are shown in Table 1.

Here we can note that Fano algorithm corrected three bit soft errors. We have checked the algorithm exhaustively for all possible combinations of this type of error pattern and found that Fano decoder corrects all $d_{\min} - 1$ number of soft errors. So the code was able to decode beyond the minimum distance and corrected more than one soft error.

For a particular received codeword, the value of Δ has an influence on the number of steps required by the algorithm to perform decoding. A large increment in the value of Δ can cause incorrect decoding as has been indicated in Table 2.

In general, the larger Δ is, the fewer number of computations are required. Ultimately, Δ must be below the likelihood of the maximum likelihood path, and so must be lowered to that point. If Δ is too small, then many iterations might be required to get that point. On the

Table 1. Decoding steps for the soft decision Fano algorithm

Position P	Computation &Result	Metric M	Threshold T	Position P	Computation & Result	Metric M	Threshold T
X	$M_F = 50$	0	0	01010	$M_B = 142$	192	-500
0	$M_F \geq T \implies P^+$	50	0	0101	$M_B \geq T \implies P^-$	142	-500
0	First Visit	50	0	0101	Not First Visit	142	-500
0	$M_F = 91$	50	0	0101	$M_B = 141$	142	-500
01	$M_F \geq T \implies P^+$	91	0	010	$M_B \geq T \implies P^-$	141	-500
01	First Visit	91	0	010	Not First Visit	141	-500
01	$M_F = 141$	91	0	010	$M_F = 11$	141	-500
010	$M_F \geq T \implies P^+$	141	0	0100	$M_F \geq T \implies P^+$	11	-500
010	First Visit	141	0	0100	Not First Visit	11	-500
010	$M_F = 142$	141	0	0100	$M_F = -986$	11	-500
0101	$M_F \geq T \implies P^+$	142	0	0100	$M_B = 141$	11	-500
0101	First Visit	142	0	010	$M_B > T \implies P^-$	141	-500
0101	$M_F = 192$	142	0	010	$M_B = 91$	141	-500
01010	$M_F \geq T \implies P^+$	192	0	01	$M_B \geq T \implies P^-$	91	-500
01010	First Visit	192	0	01	Not First Visit	91	-500
01010	$M_F = 62$	192	0	01	$M_B = 50$	91	-500
010100	$M_F \geq T \implies P^+$	62	0	0	$M_B \geq T \implies P^-$	50	-500
010100	First Visit	62	0	0	$NotFirstVisit$	50	-500
010100	$M_F = -935$	62	0	0	$M_F = -311$	50	-500
010100	$M_B = 192$	62	0	00	$M_F \geq T \implies P^+$	-311	-500
01010	$M_B > T \implies P^-$	192	0	00	First Visit	-311	-500
01010	$M_B = 142$	192	0	00	$M_F = -261$	-311	-500
0101	$M_B \geq T \implies P^-$	142	0	000	$M_F \geq T \implies P^+$	-261	-500
0101	Not First Visit	142	0	000	First Visit	-261	-500
0101	$M_B = 141$	142	0	000	$M_F = -260$	-261	-500
010	$M_B \geq T \implies P^-$	141	0	0001	$M_F \geq T \implies P^+$	-260	-500
010	First Visit	141	0	0001	First Visit	-260	-500
010	$M_F = 11$	141	0	0001	$M_F = -1257$	-260	-500
0100	$M_F \geq T \implies P^+$	11	0	0001	$M_B = -261$	-260	-500
0100	First Visit	11	0	000	$M_B > T \implies P^-$	-261	-500
0100	$M_F = -986$	11	0	000	$M_F = -391$	-261	-500
0100	$M_B = 141$	11	0	0000	$M_F \geq T \implies P^+$	-391	-500
010	All forward nodes are tested. $T = T - \Delta$	141	-500	0000	First Visit	-391	-500
010	$M_F = 142$	141	-500	0000	$M_F = -341$	-391	-500
0101	$M_F \geq T \implies P^+$	142	-500	00000	$M_F \geq T \implies P^+$	-341	-500
0101	Not First Visit	142	-500	00000	First Visit	-341	-500
0101	$M_F = 192$	142	-500	00000	$M_F = -471$	-341	-500
01010	$M_F \geq T \implies P^+$	192	-500	000000	$M_F \geq T \implies P^+$	-471	-500
01010	Not First Visit	192	-500	000000	First Visit	-471	-500
01010	$M_F = 62$	192	-500	000000	$M_F = -421$	-471	-500
010100	$M_F \geq T \implies P^+$	62	-500	0000000	$M_F \geq T \implies P^+$	-421	-500
010100	Not First Visit	62	-500	0000000	First Visit	-421	-500
010100	$M_F = -935$	62	-500	0000000	$M_F = -371$	-421	-500
010100	$M_B = 192$	62	-500	00000000	$M_F \geq T \implies P^+$	-371	-500
01010	$M_B \geq T \implies P^-$	192	-500	**00000000**	$P = LEAF \implies$ END	-371	-500

↘

Successful Decoding

other hand, if Δ is lowered in steps that are too big, then the threshold might be set low enough that other paths which are not the maximum likelihood path also exceed the threshold and can be considered by the decoder. Thus the selection of an optimum Δ value is very much crucial.

The primary benefit of this algorithm is that each correct decision contributes to limiting the amount of computation that must be performed subsequently. At the same time, the path metric is providing an indication of the correctness of earlier decisions. The path metric through node k is simply

$$L_k = \sum_{i=0}^{k} M_i \,. \tag{9}$$

The threshold increment is chosen such that the path metric will be increasing in value over a correct path and decreasing in value over an incorrect path. The typical metric behavior is shown in Fig. 6.

Although the metric for the correct path may temporarily show large decreases due to channel noise, over a longer period of time it should be an increasing function. Also, if a burst of channel noise occurs, the metric on an incorrect path may increase temporarily making the path look like a good path, but it will usually start decreasing when the noise subsides.

The main benefit of the Fano algorithm in comparison to the stack or Viterbi algorithm is its parsimonious use of memory. The various partial path metrics are computed as needed, so they need not be stored. Despite its complexity, when the noise is low the Fano algorithm tends to

Table 2. Performance of Fano algorithm on a received codeword as a function of Δ

Δ	Number of decoding steps	Correct decoding
100	73	yes
400	32	yes
500	30	yes
1000	36	yes
2150	16	no

decode faster than the stack algorithm. Its drawback is a certain loss in speed compared to the stack algorithm for higher rates, but for moderate rates the Fano algorithm decodes faster than the stack algorithm. It seems that the Fano algorithm is the preferred choice for practical implementation of sequential decoding algorithms [16].

4 CONCLUSIONS

A simple, efficient and near optimal decoding scheme for linear block codes using tree representation is proposed. The equation for obtaining optimum threshold increment has been defined. It is interesting to notice that the technique proposed in this paper can actually be used to decode any linear block codes. 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 deviation are always bounded. Since Fano algorithm does not require any storage and suffers a speed disadvantage only on very noisy channels, a typical application for the Fano algorithm could be applied in automatic- repeat-request (ARQ) schemes to improve the reliability and throughput. Although at present decoding complexity seems to be quite high, however, in return there is potentially much better performance to be attained. We have observed for an (8, 4) single bit error correction extended Hamming code, hard decision decoding corrects only a single bit error and soft decision Fano decoding corrects any three bits of soft errors. This in turn results in a coding gain over AWGN channel compare to HDD. This is to say that, with soft decision decoding, the transmitted power can be lowered to hard decision decoding, which translates into smaller transmit antennas or, alternatively, smaller receive antennas for the same transmission power.

References

[1] COSTELLO, D.—FORNEY, D.: Channel Coding: The Road to Channel Capacity, Proceedings of the IEEE **95** No. 6 (June 2007).

[2] WOLF, J. K.: Efficient Maximum Likelihood Decoding of Linear Block Codes Using a Trellis, IEEE Transaction on Information Theory **IT-24** (1978), 76–80.

[3] SOROKINE—KSCHISCHANG: A Sequential Decoder for Linear Block Codes with a Variable Bias Term Metric, IEEE Transactions on Information Theory **144** No. 1 (Jan 1998).

[4] BAHL, L. R.—COCKE, J.—JELINEK, F.—RAVIV, J.: Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate, IEEE Transaction on Information Theory **IT-20** (1974), 284–287.

[5] www.ewh.ieee.org/r6/scv/comsoc/0009.pdf.

[6] KASAMI, T. *et al*: On the Trellis Structure of Block Codes, IEEE Transaction on Information Theory **IT-39** (1993), 1057–1064.

[7] BLAHUT, R. E.: Algebraic Codes for Data Transmission, first Edition, Cambridge University Press, 2002.

[8] SCHLEGEL, C. B.—PEREZ, L. C.: Trellis and Turbo Coding, IEEE Press, Wiley Inter Science, 2004.

[9] SHU LIN—COSTELLO, D.: Error Control Coding, Pearson Education Inc., 2004.

[10] WICKER, S. B.: Error Control Systems for Digital Communication and Storage, Prentice-Hall, Edition 1995.

[11] McCLURE, J. H.—JOINER, L. L.: Soft Decision Decoding of Reed-Solomon Codes Using the Fano Sequential Algorithm, Proc. IEEE Southeastcon 01, March 2001,, pp. 131–135.

[12] MOON, T. K.: Error Correction Coding, Wiley Inter Science, Edition 2006.

[13] CHEN JUN SUN RONG—WANG XINMEI: Optimal Threshold Sequential Decoding Algorithms for Linear Block Codes, in Vehicular Technology Conference Proceedings, 2000, VTC 2000-Spring Tokyo.

[14] WU-HSIANG—CHEN, J.—FOSSORIER, M. P. C.—SHU LIN: Quantization Issues for Soft Decision Decoding of Linear Block Codes, IEEE Transactions on Communications **47** No. 6 (June 1999).

[15] JOHANNESON, R.—ZIGANGIROV, K. S.: Fundamentals of Convolutional Coding, Wiley-IEEE Press, 1999.

[16] SINGH, S. K. *et al*: Algorithms and Circuit Co-Design for a Low-Power Sequential Decoder, IEEE International Conf. on Signals, Systems and Computers, 1999.

Received 26 January 2011

Prashantha Kumar H. received his BE degree in Electronics and Communication Engineering from MCE Hassan in 1998 and received his M.Tech degree in Digital Electronics and Communication from MIT Manipal in 2001. Currently, he is a PhD student in the Department of Electronics and Communication Engineering, National Institute of Technology Karnataka, Surathkal, India.

Uupi Sripati received his PhD degree from IISC Bangalore in Electrical Communication Engineering Department in 2004. He is currently Associate Professor in Electronics and Communication Engineering at National Institute of Technology Karnataka, Surathkal, India. His major research interests include error control coding, spacetime communication, wireless sensor networks, next generation mobile communication and free space optics.

K. Rajesh Shetty received his BE degree in Electronics and Communication Engineering from NMAMIT Nitte in 1990 and received his M Tech degree in Digital Electronics and Advanced Communication from NITK Surathkal in 2004. Currently, he is a PhD student in the department of Electronics and Communication Engineering, National Institute of Technology Karnataka, Surathkal, India.

Bangara Setty Shankarananda received his PhD degree from IIT Bombay in Electrical Engineering Department. He is currently Principal of Gopalan College of Engineering, Bangalore, India. His major research interests include digital communication, optical networks and error control coding.