Low-complexity trellis decoding of linear block codes

B. Honary
G. Markarian
M. Darnell

Indexing terms: Trellis decoding, Linear block codes

Abstract: The authors introduce a new simple encoding technique which allows the design of a wide variety of linear block codes. They present a number of examples in which the most widely used codes (Reed–Muller, Hamming, Golay, optimum etc.) have been designed. They also introduce a novel trellis design procedure for the proposed codes. It is shown that the trellises of the designed codes are similar to the trellises of coset codes and allow low complexity soft maximum likelihood decoding.

1 Introduction

Trellis decoding technique for linear block codes has been under investigation since 1974 [1–3]. Later, a number of efficient soft decision maximum likelihood decoding (MLD) algorithms for block codes were proposed [4, 6, 7]. Recent publications by Forney [8, 9] and others [5, 10, 11] have stimulated interest in low-complexity trellis decoding of block codes for both practical and theoretical reasons. On the practical side, a minimal trellis structure of a block code can be used for performing low-complexity maximum likelihood soft decision decoding. From the theoretical viewpoint, such trellises can be used for construction of other codes [10].

Recently, generalised array codes (GACs) and their trellis structures have been introduced [12]. The technique allows the design of array codes with the same code length, n, and $d_{min} = 4$, but with an increased number of information digits. These codes are simple and flexible to design; they also allow low-complexity soft maximum likelihood trellis decoding (SMLTD). It has been shown that the concept of GACs can be applied to the design of Hamming, Reed–Muller (RM) and Golay codes with a low-complexity trellis structure [13, 14]. In this paper we describe a new encoding/decoding technique for different types of linear codes based on GACs and their trellis structure. The trellises of the designed codes are similar to trellises of the coset codes designed by Forney [8] and provide lower complexity SMLTD than that achievable with conventional techniques.

2 Code construction

A generalised (n, k, d_{min}) array code is an array code in which the column and row subcodes may have different numbers of information and parity check symbols; the code length $n = n_1 n_2$ and the total number of information digits $k = k_1 + k_2 + \cdots + k_{n_2}$, where n_1 and n_2 represent the number of columns and rows respectively, k_p is the number of information digits in the pth row [12]. Using the concept of the GACs, a wide variety of known linear block codes together with their low complexity trelisses can be designed. The procedure for designing a linear (n_0, k_0, d_0) GAC is as follows:

(i) Design a binary, $n = n_1 n_2$, $(n = n_0)$ basic product code, C_1, as shown in Fig. 1*a*, with a single parity check column and $R_1 = (n_1, k_1, d_1)$ row codes, where $d_1 = \lceil d_0/2 \rceil$ and $\lceil \cdot \rceil$ is the nearest greatest integer (if n_0 is a prime number choose $n = n_0 + 1$).

(ii) Design a binary, $n_1 n_2$, additional product code, $C_2 = |PA|$, as shown in Fig. 1*b*, where P is a binary $k_1 n_2$, array with only parity check elements; A is a binary, $(n_1 - k_1)n_2$, matrix where the first row consists of only $k' = n_1 - k_1$ information digits, and all column codes are repetition codes.

(iii) Design (if not all information digits have been used) a second additional binary product code, C_3, as shown in Fig. 1*c*, where $B = (n_2, 1, d_0)$ is a repetition row code with k_0 th information digit.

(iv) Add the designed codes as follows:

$$C = C_1 \oplus C_2 \oplus C_3 \tag{1}$$

where addition is on modulo − 2 basis.

(iv) If $n = n_0 + 1$, delete the symbol which is located in the n_2 th row and n_1 th column.

This procedure can be also described as an array representation of coset codes introduced by Forney [8]. Code C_1, is a linear product codes, and the overall designed C can be represented as a union of cosets of C_1. This union forms a linear nonsystematic code with the following parameters: (n_0, k_0, d_0). In this section we show that the proposed technique can be employed for the design of a wide variety of linear block codes.

Paper 2037I (E5), first received 2nd September 1994 and in revised form 24th April 1995

B. Honary and G. Markarian are with the Communications Research Centre, Lancaster University, Lancaster LA1 4YR, United Kingdom

M. Darnell is with the Department of Electronic and Electrical Engineering, University of Leeds, Leeds LS2 9JT, United Kingdom

The authors would like to express their gratitude towards Prof. R.J. McEliece for his very helpful advice and comments regarding the revision of the paper. This work was supported by the Science and Engineering Research Council (SERC), UK.

2.1 (8, 4, 4) and (7, 4, 3) codes design

An example is given for an (8, 4, 4) GAC which is equivalent to the well known (8, 4, 4) RM code [15, 16]. The procedure for code design is given below:

(i) Design the basic, (8, 3, 4), ($n = 2 \times 4$, $n_1 = 2$, $n_2 = 4$) array code, C_1, with the single-parity check (4, 3, 2) column and repetition $R_1 = (2, 1, 2)$ row codes ($d_1 = d_0/2 = 2$):

$$C_1 = \begin{bmatrix} x_1 & p_1 \\ x_2 & p_2 \\ x_3 & p_3 \\ p_4 & p_4 \end{bmatrix} \tag{2}$$

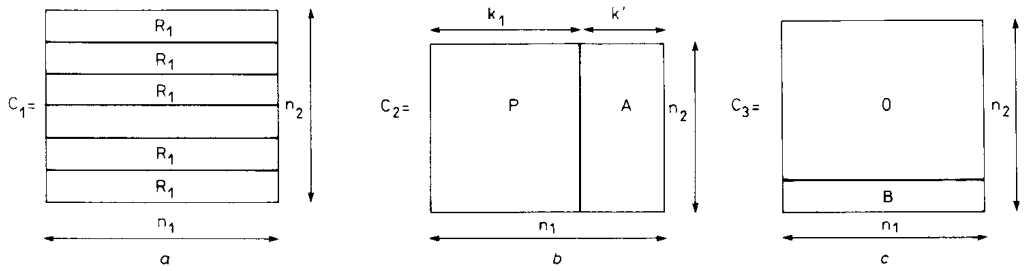

Fig. 1 *Code structures*
a Basic product code, C_1 *b* Additional product code, C_2 *c* Additional product code, C_3

where x_i, $i = 1, 2, 3$ represent information digits and p_j, $j = 1, \ldots, 4$, represent parity check symbols:

$$\left.\begin{aligned} p_j &= x_j \; \forall j \leqslant 3 \\ p_4 &= x_1 + x_2 + x_3 \end{aligned}\right\} \tag{3}$$

(ii) Design an additional array code, $C_2 = |PA|$ with the following structure:

$$C_2 = \begin{bmatrix} 0 & x_4 \\ 0 & x_4 \\ 0 & x_4 \\ 0 & x_4 \end{bmatrix} \tag{4}$$

where x_4 is an information digit, and P is an all-zero column.

(iii) Since all information digits have been used, there is no need to design a second additional array code, C_3.

(iv) Add the two codes C_1 and C_2 on a modulo-2 basis and read the designed code row-by-row as follows:

$$C = C_1 \oplus C_2 = \begin{bmatrix} x_1 & x_4 \oplus p_1 \\ x_2 & x_4 \oplus p_2 \\ x_3 & x_4 \oplus p_3 \\ p_4 & x_4 \oplus p_4 \end{bmatrix}$$

$$= (x_1, x_4 \oplus p_1, x_2, x_4 \oplus p_2, x_3, x_4 \oplus p_3, p_4, x_4 \oplus p_4) \tag{5}$$

(v) Since $n_0 = n$, there is no need for the deletion of the parity-check symbol which is located in the 4th row and 2nd column.

The designed code is a linear nonsystematic code and has the following parameters: ($n_0 = 8$, $k_0 = 4$, $d_0 = 4$). The weight distribution function (WDF) of the code is shown in Fig. 2 and is similar to the WDF of the (8, 4, 4) RM code. It has been shown [13] that by deleting the symbol which is located in the second column and 4th row, the designed (8, 4, 4) code will be transformed to the (7, 4, 3) Hamming code:

$$C = C_1 \oplus C_2 = \begin{bmatrix} x_1 & x_4 \oplus p_1 \\ x_2 & x_4 \oplus p_2 \\ x_3 & x_4 \oplus p_3 \\ p_4 & \end{bmatrix}$$

$$= (x_1, x_4 \oplus p_1, x_2, x_4 \oplus p_2, x_3, x_4 \oplus p_3, p_4) \tag{6}$$

2.2 (16, 5, 8) and (15, 5 7) codes design

An example is now given for (16, 5, 8) and (15, 5, 7) GACs. The procedure for codes design is shown below:

(i) Design the basic, (16, 3, 8), ($n = 4 \times 4$, $n_1 = n_2 = 4$) product code, C_1, with the repetition row code $R_1 = (4, 1, 4)$, and (4, 3, 2) single parity check column code ($d_1 = d_0/2 = 4$):

$$C_1 = \begin{bmatrix} x_1 & p_1 & p_1 & p_1 \\ x_2 & p_2 & p_2 & p_2 \\ x_3 & p_3 & p_3 & p_3 \\ p_4 & p_4 & p_4 & p_4 \end{bmatrix} \tag{7}$$

where x_i, $i = 1, 2, 3$, represent information digits and parity check symbols, p_j, $j = 1, 2, 3, 4$, are defined according to eqn. 3.

(ii) Design an additional product code, $C_2 = |PA|$ with the following structure:

$$C_2 = \begin{bmatrix} 0 & p_5 & x_4 & x_5 \\ 0 & p_5 & x_4 & x_5 \\ 0 & p_5 & x_4 & x_5 \\ 0 & p_5 & x_4 & x_5 \end{bmatrix} \tag{8}$$

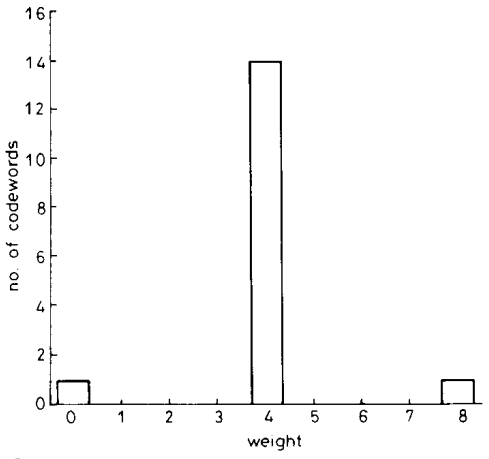

Fig. 2 *Weight distribution function of (8, 4, 4) GAC*

where x_4 and x_5 are information digits, P is a 2×4 parity matrix and $p_5 = x_4 + x_5$.

(iii) Since all information digits have been used, there is no need to design a second additional array code, C_3.

(iv) Add the two codes C_1 and C_2 on a modulo 2 basis as follows:

$$C = C_1 \oplus C_2 = \begin{bmatrix} x_1 & p_5 \oplus p_1 & x_4 \oplus p_1 & x_5 \oplus p_1 \\ x_2 & p_5 \oplus p_2 & x_4 \oplus p_2 & x_5 \oplus p_2 \\ x_3 & p_5 \oplus p_3 & x_4 \oplus p_3 & x_5 \oplus p_3 \\ p_4 & p_5 \oplus p_4 & x_4 \oplus p_4 & x_5 \oplus p_4 \end{bmatrix} \quad (9)$$

This code is equivalent to the (16, 5, 8) RM code from which a (15, 5, 7) GAC can be derived by simple deletion of the parity chack symbol located in the 4th row and 4th column:

$$C = \begin{bmatrix} x_1 & p_5 \oplus p_1 & x_4 \oplus p_1 & x_5 \oplus p_1 \\ x_2 & p_5 \oplus p_2 & x_4 \oplus p_2 & x_5 \oplus p_2 \\ x_3 & p_5 \oplus p_3 & x_4 \oplus p_3 & x_5 \oplus p_3 \\ p_4 & p_5 \oplus p_4 & x_4 \oplus p_4 & \end{bmatrix} \quad (10)$$

The weight distribution function (WDF) of the designed code is shown in Fig. 3. It is evident that the derived code has the following parameters: ($n_0 = 15$, $k_0 = 5$, $d_0 = 7$), and is equivalent to a corresponding BCH code.

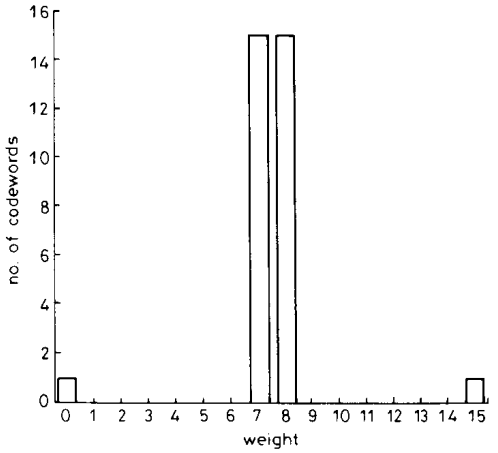

Fig. 3 *Weight distribution function of (15, 5, 7) GAC*

2.3 (16, 11, 4) and (15, 11, 3) codes design

The proposed technique can be implemented for the design of other RM codes with higher information rates. An example is given for a (16, 11, 4) GAC which is equivalent to the corresponding RM code. The procedure for code design is as follows:

(i) Design a basic 4×4 array code C_1, with the single parity check row and column codes ($R_1 = (4, 3, 2)$, $d_1 = d_0/2$):

$$C_1 = \begin{bmatrix} x_1 & x_2 & x_3 & p_1 \\ x_4 & x_5 & x_6 & p_2 \\ x_7 & x_8 & x_9 & p_3 \\ p_4 & p_5 & p_6 & p_7 \end{bmatrix} \quad (11)$$

where x_i, $i = 1, 2, \ldots, 9$, represent information digits and p_j, $j = 1, 2, \ldots, 7$, represent single parity check digits for row and column codes.

(ii) Design a binary additional product code $C_2 = |PA|$

$$C_2 = \begin{bmatrix} 0 & 0 & 0 & x_{10} \\ 0 & 0 & 0 & x_{10} \\ 0 & 0 & 0 & x_{10} \\ 0 & 0 & 0 & x_{10} \end{bmatrix} \quad (12)$$

where P is an all zero 4×3 matrix, A is a 4×1 repetition column matrix and x_{10} is an information digit.

(iii) Since not all information digits have been used, design second additional array, C_3, with the repetition code at the last row:

$$C_3 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ x_{11} & x_{11} & x_{11} & x_{11} \end{bmatrix} \quad (13)$$

where x_{11} is an information digit.

(iv) Add the three codes on a modulo-2 basis as follows:

$$C = C_1 \oplus C_2 \oplus C_3 = \begin{bmatrix} x_1 & x_2 & x_3 & p_1 \oplus x_{10} \\ x_4 & x_5 & x_6 & p_2 \oplus x_{10} \\ x_7 & x_8 & x_9 & p_3 \oplus x_{10} \\ p_4 \oplus x_{11} & p_5 \oplus x_{11} & p_6 \oplus x_{11} & p_7 \oplus x_{11} \oplus x_{10} \end{bmatrix} \quad (14)$$

It follows from the construction that the designed code is a (16, 11, 4) GAC which is equivalent to the corresponding RM code. It is also apparent that the (15, 11, 3) Hamming code can be simply derived by deleting the parity check symbol located in the 4th row and 4th column of the designed (16, 11, 4) code.

2.4 (15, 7, 5) code design

The procedure outlined above can be applied to the design of a (15, 7, 5) GAC. To illustrate the flexibility of the proposed technique, we choose a different size of the basic array. The encoding procedure is as follows:

(i) Design the basic, (15, 4, 6) ($n = 5 \times 3$, $n_1 = 5$, $n_2 = 3$) product code, C_1, with a $R_1 = (5, 2, 3)$ row code ($d_1 = [d_0/2] = 3$) and (3, 2, 2) single parity check column code:

$$C_1 = \begin{bmatrix} x_1 & x_2 & p_1 & p_2 & p_3 \\ x_3 & x_4 & p_4 & p_5 & p_6 \\ p_7 & p_8 & p_9 & p_{10} & p_{11} \end{bmatrix} \quad (15)$$

where x_i, $i = 1, 2, 3, 4$, represent information digits and $p_j, j = 1, \ldots, 11$, represent parity check symbols.

(ii) Design two additional product codes, C_2 and C_3 with the following structure:

$$C_2 = \begin{bmatrix} 0 & 0 & p_{12} & x_5 & x_6 \\ 0 & 0 & p_{12} & x_5 & x_6 \\ 0 & 0 & p_{12} & x_5 & x_6 \end{bmatrix}$$

$$C_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ x_7 & x_7 & x_7 & x_7 & x_7 \end{bmatrix} \quad (16)$$

where x_i, $i = 5, 6, 7$ represent information digits and $p_{12} = x_5 + x_6$.

(iii) Add the three codes on a modulo-2 basis, as follows:

$$C = C_1 \oplus C_2 \oplus C_3$$

$$= \begin{bmatrix} x_1 & x_2 & p_1 \oplus p_{12} & x_5 \oplus p_2 & x_6 \oplus p_3 \\ x_3 & x_4 & p_4 \oplus p_{12} & x_5 \oplus p_5 & x_6 \oplus p_6 \\ p_7 \oplus x_7 & p_8 \oplus x_7 & p_9 \oplus x_7 \oplus p_{12} & p_{10} \oplus x_7 \oplus x_5 & p_{11} \oplus x_7 \oplus x_6 \end{bmatrix} \quad (17)$$

The designed code has the following parameters: ($n_0 = 15$, $k_0 = 7$, $d_0 = 5$), and is equivalent to a corresponding BCH code.

2.5 (17, 9, 5) code design

Application of the proposed technique is not restricted to the RM and Hamming codes. The technique provides a 'good' co-ordinate reordering for other codes, for example optimum codes. An example is given for (17, 9, 5) optimum code [16]; however, other known optimum codes can be obtained easily. The procedure is as follows:

(i) Design the basic (18, 6, 6) ($n = 6 \times 3$, $n_1 = 6$, $n_2 = 3$, $n = n_0 + 1$) product code, C_1, with a $R_1 = (6, 3, 3)$ row code ($d_1 = \lceil d_0/2 \rceil = 3$), and (3, 2, 2) single parity check column code:

$$C_1 = \begin{bmatrix} x_1 & x_2 & x_3 & p_1 & p_2 & p_3 \\ x_4 & x_5 & x_6 & p_4 & p_5 & p_6 \\ p_7 & p_8 & p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \quad (18)$$

where x_i, $i = 1, 2, \ldots, 6$, represent information digits and p_j, $j = 1, \ldots, 12$, represent parity check symbols.

(ii) Design two additional product codes, C_2 and C_3, as follows:

$$C_2 = \begin{bmatrix} 0 & 0 & 0 & p_{13} & x_7 & x_8 \\ 0 & 0 & 0 & p_{13} & x_7 & x_8 \\ 0 & 0 & 0 & p_{13} & x_7 & x_8 \end{bmatrix}$$

$$C_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ x_9 & x_9 & x_9 & x_9 & x_9 & x_9 \end{bmatrix} \quad (19)$$

where x_i, $i = 7, 8, 9$, represent information digits and $p_{13} = x_7 + x_8$.

(iii) Add the three codes, C_1, C_2 and C_3, on a modulo-2 basis;

(iv) Delete the symbol which is located in the 3rd row and 6th column:

$$C = \begin{bmatrix} x_1 & x_2 & x_3 & p_1 \oplus p_{13} & p_2 \oplus x_7 & p_3 \oplus x_8 \\ x_4 & x_5 & x_6 & p_4 \oplus p_{13} & p_5 \oplus x_7 & p_6 \oplus x_8 \\ p_7 \oplus x_9 & p_8 \oplus x_9 & p_9 \oplus x_9 & p_{10} \oplus p_{13} \oplus x_9 & p_{11} \oplus x_7 \oplus x_9 & \end{bmatrix} \quad (20)$$

It is apparent that the designed code is a linear non-systematic optimum code with the following parameters: ($n_0 = 17$, $k_0 = 9$, $d_0 = 5$).

2.6 Golay codes design

The (24, 12, 8) binary extended Golay code occupies a remarkable place among binary block codes [8]. It is a unique, perfect, self-dual code which has been intensively investigated with various efficient encoding and decoding algorithms [4, 6, 17]. The trellis diagram for the (24, 12, 8) extends Golay code, which consists of 64 states, has been introduced by Forney [8]. However, this technique requires complex encoder and can not be used for the trellis design of the (23, 12, 7) Golay code. In this section we implement the concept of GACs for the design of both (24, 12, 8) and (23, 12, 7) Golay codes. We start with the design of (24, 12, 8) extended Golay code. The procedure is as follows:

(i) Design the basic (24, 8, 8) ($n = 8 \times 3$, $n_1 = 8$, $n_2 = 3$) product code, C_1, with the (3, 2, 2) column and (8, 4, 4) row codes:

$$C_1 = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & p_1 & p_2 & p_3 & p_4 \\ x_5 & x_6 & x_7 & x_8 & p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} & p_{13} & p_{14} & p_{15} & p_{16} \end{bmatrix} \quad (21)$$

where x_i, $i = 1, 2, \ldots, 8$, represent information digits; p_j, $j = 1, \ldots, 16$, represent parity check symbols and the (8, 4, 4) row code has been described in Section 2.1.

(ii) Design two additional product codes, C_2 and C_3, with the following structures:

$$C_2 = \begin{bmatrix} c_1 & c_2 & c_3 & c_4 & c_5 & x_9 & x_{10} & x_{11} \\ c_1 & c_2 & c_3 & c_4 & c_5 & x_9 & x_{10} & x_{11} \\ c_1 & c_2 & c_3 & c_4 & c_5 & x_9 & x_{10} & x_{11} \end{bmatrix}$$

$$C_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ x_{12} & x_{12} & x_{12} & x_{12} & x_{12} & x_{12} & x_{12} & x_{12} \end{bmatrix} \quad (22)$$

where row code in C_2 is (8, 3, 3) code and is shown in Table 1.

(iii) Add the three codes, C_1, C_2 and C_3 on a modulo-2 basis:

$$C = \begin{bmatrix} x_1 \oplus c_1 & x_2 \oplus c_2 & x_3 \oplus c_3 & x_4 \oplus c_4 & p_1 \oplus c_5 & p_2 \oplus x_9 & p_3 \oplus x_{10} & p_4 \oplus x_{11} \\ x_5 \oplus c_1 & x_6 \oplus c_2 & x_7 \oplus c_3 & x_8 \oplus c_4 & p_5 \oplus c_5 & p_6 \oplus x_9 & p_7 \oplus x_{10} & p_8 \oplus x_{11} \\ x_{12} \oplus p_9 \oplus c_1 & x_{12} \oplus p_{10} \oplus c_2 & x_{12} \oplus p_{11} \oplus c_3 & x_{12} \oplus p_{12} \oplus c_4 & x_{12} \oplus p_{13} \oplus c_5 & x_{12} \oplus p_{14} \oplus x_9 & x_{12} \oplus p_{15} \oplus x_{10} & x_{12} \oplus p_{16} \oplus x_{11} \end{bmatrix} \quad (23)$$

The weight distribution function for the designed (24, 12, 8) code is shown in Fig. 4*a*. This is similar to the WDF of the (24, 12, 8) extended Golay code [17], thus, following Reference 17, the designed code is an extended (24, 12, 8) Golay code. The (23, 12, 7) code can be derived by simple deletion of the parity-check symbol which is located in the 3rd row and 8th column. The weight distribution function of the designed (23, 12, 7) code is shown in Fig. 4*b*. This is identical to the weight distribution function of the (23, 12, 7) Golay code [17].

Example: A binary vector of information digits is given as: $X = (x_1, x_2, \ldots, x_{12}) = (001101011111)$. Following the procedure outlined above, C_1, C_2 and C_3 codewords

can be written as follows:

$$C_1 = \begin{bmatrix} 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

$$C_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \quad (12)$$

$$C_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

and the encoded codeword $C = (10111011\ 11011101\ 00101101)$.

Table 2 illustrates the code design procedure for a number of codes with lengths up to 64. This Table includes the quasiperfect codes listed in [16], and provides all parameters that are required for the design of a GAC. For example, the optimum (20, 11, 5) code [16] can be designed by choosing a 3×7 basic array with the $R_1 = (7, 4, 3)$ row and (3, 2, 2) column codes. The first additional code C_2, should consist of a (7, 2, 3) row and (3, 1, 3) column codes while the second additional code C_3, should have x_{11} repeated in the last row.

Similar to the technique, introduced by Forney [8], the proposed technique can be implemented for the design of the (16, 8, 8) Nordstrom–Robinson code [20]. This code can be obtained from the (24, 12, 8) Golay code by simple deletion of the second row in the C_1, C_2, C_3 and C.

Table 1: Code table of (8, 3) code

Information vector (x_9, x_{10}, x_{11})	Encoded codeword	Information vector (x_9, x_{10}, x_{11})	Encoded codeword
0 0 0	0 0 0 0 0 0 0 0	1 0 0	1 1 1 0 0 1 0 0
0 0 1	1 1 0 1 0 0 0 1	1 0 1	1 1 0 0 1 0 1 0
0 1 0	1 0 1 1 0 0 1 0	1 1 0	0 1 0 1 0 1 1 0
0 1 1	1 0 0 1 1 1 0 0	1 1 1	1 0 0 0 0 1 1 1

2.7 Other known codes

The technique described can be readily implemented for the design of other known codes. For example, following the procedure outlined above, one can design the (32, 6, 16) RM code by choosing a 4×8 ($n_1 = 4$, $n_2 = 8$) basic array code, C_1, with an $R_1 = (8, 1, 8)$ row and (4, 3, 2) single parity check column codes. The first additional product code, C_2, should consist of a 4×5 matrix P and a 4×3 matrix A of repetition column codes. The third additional product code, C_3, must be an all zero matrix. The (31, 6, 15) code can be derived easily be deleting the parity check symbol which is located in the 4th row and 8th column.

The (32, 26, 4) and (32, 16, 8) RM codes which were discussed in [8] can also be designed by using the proposed algorithm. The (31, 26, 3) and (31, 16, 7) codes, which are equivalent to the corresponding BCH codes, can be derived from the corresponding RM codes by deleting the parity check symbols, located in the last column and row. It is apparent that since any RM code can be decomposed into two codes with shorter code lengths [19] the concept of GACs can be applied for the design of all RM codes.

3 Trellis design procedure

The trellis design procedure for GACs [12] can be extended for the design of low-complexity regular trellises for all the codes described above. The procedure is as follows:

(i) Choose the trellis depth (number of columns), N_c, and number of states, N_s, as

$$N_c = n_2 = 1 \qquad N_s = 2^{\max k_p} \quad (25)$$

(ii) Identify each state of depth p by the $\max\{k_p\}$-tuple binary vector $S_p(A) = S_p(a_1, a_2, \ldots, a_{\max(kp)})$, where $a_j = 0, 1$.

(iii) The trellis branches start and finish at depth $p = 0$ and $p = n_2$; these are labelled as $S_0(00, \ldots, 0)$ and $S_{n2}(00, \ldots, 0)$, respectively.

(iv) The trellis branches at depth p are labelled X_p/C_p, where X_p represent k_p-tuple binary vectors of information digits for the pth row code and C_p correspond to the encoded codewords in the pth row:

$$C_p = X_p^1 G_p + C_p^2 \quad (26)$$

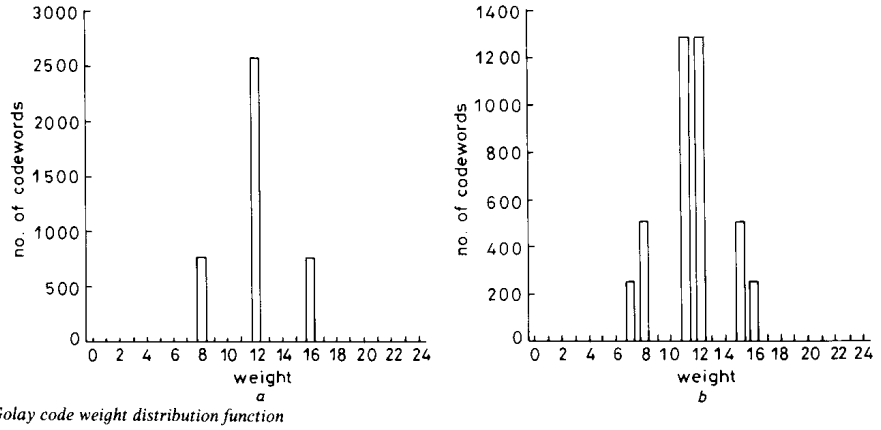

Fig. 4 *Golay code weight distribution function*
a (24, 12, 8) Golay code *b* (23, 12, 7) Golay code

where G_p is a generator matrix for the pth row code and X_p^1 and C_p^2 are codewords from the pth rows of the C_1 and C_2 codes, respectively.

(v) There are 2^{k_p} branches starting from each state, $S_p(A)$, at depth p, $p < N_c$; each branch is connected with state $S_{p+1}(A)$ at depth $p + 1$, which is defined as follows:

$$S_{p+1}(A_j) = S_p(A_i) + X_p \tag{27}$$

(vi) If a second additional code, C_3, is used for a code design, at the final depth all states must be connected to final state, $S_n(00, \ldots, 0)$, with two parallel branches; the labels of these branches complement to each other.

There are

$$N_0 = \prod_{p=1}^{n_2} 2^{k_p} \tag{28}$$

distinct paths through this trellis diagram and each path corresponds to a unique codeword from the code.

3.1 Trellis diagrams of the (8, 4, 4) and (7, 4, 3) codes

Let our aim be to design the trellis diagram of the (8, 4, 4) code described in the Section 2.1. Following the technique outlined above, the trellis diagram of the (8, 4, 4) code will have $N_c = 4 + 1 = 5$ columns and $N_s = 2^2 = 4$ states. We identify the states by a 2-tuple binary vectors and at states $p = 0$ and $p = 5$ the trellis has only one state, namely $S_0(00)$ and $S_4(00)$, respectively. Following the above procedure, the trellis diagram for the (8, 4, 4) code is presented in Fig. 5*a*, and is similar to that given by Forney [8, 9]. The trellis diagram of the (7, 4, 3) Hamming code is similar to the trellis of the (8, 4, 4) code and differs only in the number of digits being used for labelling the branches at final depth (Fig. 5*b*).

3.2 Trellis diagrams of (16, 5, 8) and (15, 5, 7) code

Now, let our aim be to design the trellis diagrams of the (16, 5, 8) and (15, 5, 7) code described in the Section 2.2. Since the trellis diagram of the (15, 5, 7) code can be derived from the trellis of the (16, 5, 8) RM code we start this example with the trellis design for (16, 5, 8) code. The trellis will have $N_c = 4 + 1 = 5$ columns and $N_s = 2^3 = 8$ states. We identify each state by a 3-tuple binary vector and at depths $p = 0$ and $p = 4$, the trellis has only one state, namely $S_0(000)$ and $S_4(000)$, respectively. Following the above procedure, the trellis diagram for the (16, 5, 8) GAC will be similar to that given by Forney for a corresponding RM code [8]. The trellis diagram of the (15, 5, 7) code can be derived easily by deleting one parity check symbol in the labelling of branches at depth $p = 5$, and is presented in Fig. 6*b*. It is apparent that the designed codes have similar trellis diagrams, which differ only in the number of digits being used for labelling the branches at the greatest depth.

Table 2: Encoding procedure for different GACs

No.	Type of code	R	Size of array	Row code in C_1	Column code in C_1	Row code in C_2	Last row in C_3	Last symbol
	$d_{min} = 3$							
1	(6, 3)	0.5	2 × 3	(2, 1, 2)	(3, 2, 2)	(2, 1, 1)	n/a	exists
2	(7, 4)	0.57	2 × 4	(2, 1, 2)	(4, 3, 2)	(2, 1, 1)	n/a	deleted
3	(9, 5)	0.55	2 × 5	(2, 1, 2)	(5, 4, 2)	(2, 1, 1)	n/a	deleted
4	(9, 6)	0.67	3 × 3	(3, 2, 2)	(3, 2, 2)	(3, 1, 1)	(3, 1, 3)	exists
5	(11, 6)	0.545	2 × 6	(2, 1, 2)	(6, 5, 2)	(2, 1, 1)	n/a	deleted
6	(12, 8)	0.67	3 × 4	(3, 2, 2)	(4, 3, 2)	(3, 1, 1)	(3, 1, 3)	exists
7	(13, 7)	0.538	2 × 7	(2, 1, 2)	(7, 6, 2)	(2, 1, 1)	n/s	deleted
8	(15, 11)	0.73	4 × 4	(4, 3, 2)	(4, 3, 2)	(4, 1, 1)	(4, 1, 4)	deleted
9	(18, 12)	0.67	3 × 6	(3, 2, 2)	(6, 5, 2)	(3, 1, 1)	(3, 1, 3)	exists
10	(30, 20)	0.67	3 × 10	(3, 2, 2)	(10, 9, 2)	(3, 1, 1)	(3, 1, 3)	exists
11	(45, 30)	0.67	3 × 15	(3, 2, 2)	(15, 14, 2)	(3, 1, 1)	(3, 1, 3)	exists
12	(63, 42)	0.67	3 × 21	(3, 2, 2)	(21, 20, 2)	(3, 1, 1)	(3, 1, 3)	exists
	$d_{min} = 4$							
13	(8, 4)	0.5	2 × 4	(2, 1, 2)	(4, 3, 2)	(2, 1, 1)	n/a	exists
14	(10, 5)	0.5	2 × 5	(2, 1, 2)	(5, 4, 2)	(2, 1, 1)	n/a	exists
15	(12, 7)	0.58	3 × 4	(3, 2, 2)	(4, 3, 2)	(3, 1, 1)	n/a	exists
16	(14, 7)	0.5	2 × 7	(2, 1, 2)	(7, 6, 2)	(2, 1, 1)	n/a	exists
17	(15, 9)	0.6	3 × 5	(3, 2, 2)	(5, 4, 2)	(3, 1, 1)	n/a	exists
18	(16, 11)	0.69	4 × 4	(4, 3, 2)	(4, 3, 2)	(4, 1, 1)	(4, 1, 4)	exists
19	(30, 19)	0.63	3 × 10	(3, 2, 2)	(3, 2, 2)	(3, 1, 1)	n/a	exists
	$d_{min} = 5$							
20	(15, 7)	0.47	5 × 3	(5, 2, 3)	(3, 2, 2)	(5, 2, 2)	(5, 1, 5)	exists
21	(17, 9))	0.53	6 × 3	(6, 3, 3)	(3, 2, 2)	(6, 2, 2)	(6, 1, 6)	deleted
22	(19, 10)	0.53	7 × 3	(7, 4, 3)	(3, 2, 2)	(7, 1, 3)	(5, 1, 5)	2 deleted
23	(20, 11)	0.55	7 × 3	(7, 4, 3)	(3, 2, 2)	(7, 2, 3)	(6, 1, 6)	deleted
24	(21, 12)	0.57	7 × 3	(7, 4, 3)	(3, 2, 2)	(7, 3, 3)	(6, 1, 6)	exists
	$d_{min} = 7$							
25	(15, 5)	0.33	4 × 4	(4, 1, 4)	(3, 2, 2)	(4, 2, 2)	n/a	deleted
26	(23, 12)	0.52	8 × 3	(8, 4, 4)	(3, 2, 2)	(8, 3, 3)	(7, 1, 7)	deleted
27	(31, 16)	0.52	8 × 4	(8, 4, 4)	(4, 3, 2)	(8, 3, 3)	(7, 1, 7)	deleted
28	(63, 32)	0.51	8 × 8	(8, 4, 4)	(8, 7, 2)	(8, 3, 3)	(7, 1, 7)	deleted
	$d_{min} = 8$							
29	(16, 5)	0.31	4 × 4	(4, 1, 4)	(3, 2, 2)	(4, 2, 2)	n/a	exists
30	(24, 12)	0.5	8 × 3	(8, 4, 4)	(3, 2, 2)	(8, 3, 3)	(8, 1, 8)	exists
31	(32, 16)	0.5	8 × 4	(8, 4, 4)	(4, 3, 2)	(8, 3, 3)	(8, 1, 8)	exists
32	(64, 32)	0.5	8 × 8	(8, 4, 4)	(8, 7, 2)	(8, 3, 3)	(8, 1, 8)	exists

3.3 Trellis diagrams of the (16, 11, 4) and (15, 11, 3) codes

The trellis diagram of the (16, 11, 4) GAC can be designed in two different ways [12]:

(i) with $N_c = 5$ columns and $N_c = 16$ states (if no parallel branches are used);

(ii) with $N_c = 5$ columns and $N_s = 8$ states (if parallel branches are used at every depth of the trellis).

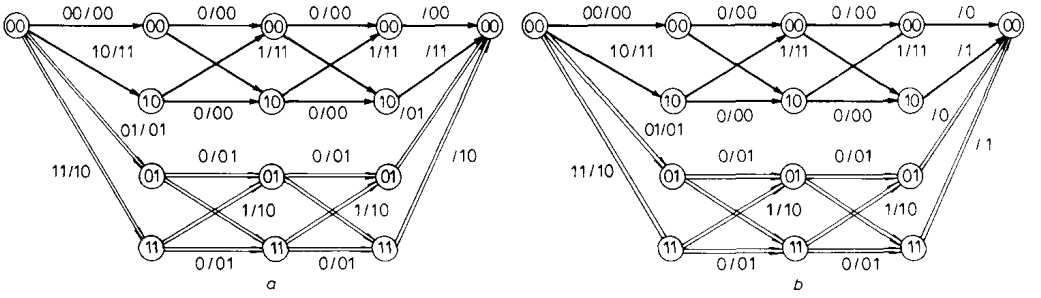

Fig. 5 *Trellis diagrams*
a (8, 4, 4) RM code *b* (7, 4, 3) Hamming code

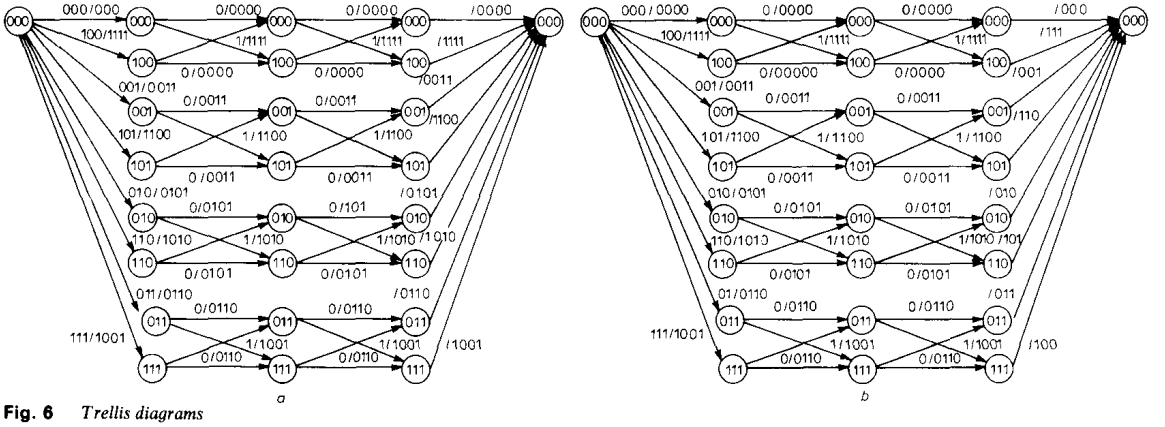

Fig. 6 *Trellis diagrams*
a (16, 5, 8) RM code *b* (15, 5, 7) GAC

In both cases, the trellis has 2^{11} different paths and is isomorphic to the trellis diagram of the (16, 11, 4) RM code [8]. The (15, 11, 3) Hamming code will have a similar trellis with the only difference being that three digits are used for labelling of branches at the final depth.

3.4 Trellis diagram of the (15, 7, 5) code

The proposed technique can be applied to the design of the trellis diagram for the (15, 7, 5) code described previously. Following the procedure outlined above, the trellis will have $N_s = 16$ and $N_c = 4$. Each state of the trellis is identified by a 4-tuple binary vector and trellis branches are labelled according to eqn. 26. Since a second additional code, C_3, has been used for the code design, at the depth $p = N_c$ each state is connected to the state $S_3(0000)$ with two parallel branches, as is shown in Fig. 7. The number of states for the designed (15, 7, 5) code can be reduced to 8, if parallel branches are used at each depth of the trellis diagram [18].

3.5 Trellises of other known codes

The above technique can be implemented for the trellis design of almost all known codes. Figs 8–11 present trellis diagrams for different codes which were obtained by using the new technique. The trellis diagram of the (31, 6, 15) (Fig. 8) has the following parameters: $N_s = 16$, $N_c = 5$ and is similar to the trellis diagram of the (32, 6, 16) RM code [8]. The only difference is in the number of digits which are used for labelling branches in the last depth of the trellis.

The trellis diagram of the (24, 12, 8) Golay code (Fig. 9) represents a set of 8 similar subtrellises. Each subtrellis has 8 states which are connected by the two parallel

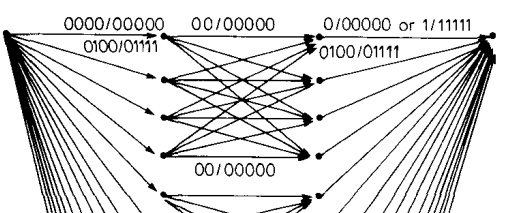

Fig. 7 *Trellis diagram of (15, 7, 5) GAC*

branches. The labelling of each subtrellis can be easily obtained from the labelling of the first subtrellis by adding corresponding codewords of the (8, 3, 3) code (Table 1). The overall trellis has the following parameters: $N_s = 64$, $N_c = 3$ and is similar to the trellis diagram given in Reference 8. The trellis diagram of the (23, 12, 7) code has the same structure as (24, 12, 8) code

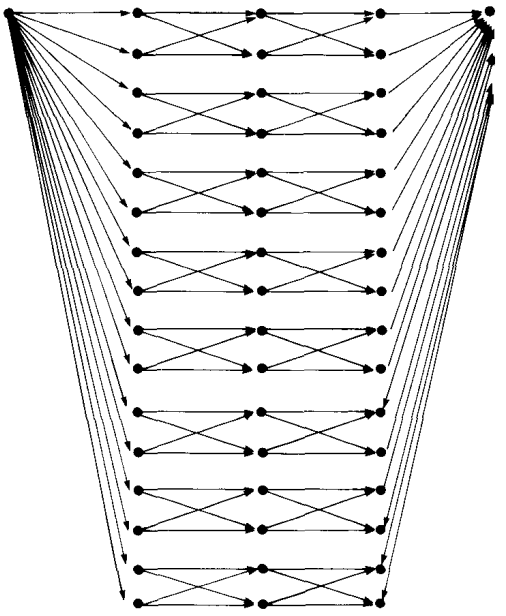

Fig. 8 *Trellis diagram of (31, 6, 15) code*

It is apparent from these Figures that the code design technique described in this paper provides codes with regular trellis structure, thus allowing the low-complexity trellis decoding techniques, such as coset decoding, to be implemented.

4 Conclusion

A new low-complexity encoding technique for linear block codes, based on generalised array codes, is described. The technique allows the design of linear block codes together with their low-complexity trellises. Examples are presented for a number of widely used codes, (Reed–Muller, Hamming, Golay, optimum etc.); however, the technique can be employed in the design of a wide variety of known linear block codes. The designed trellises have a regular structure and allow low-complexity VLSI and DSP implementations.

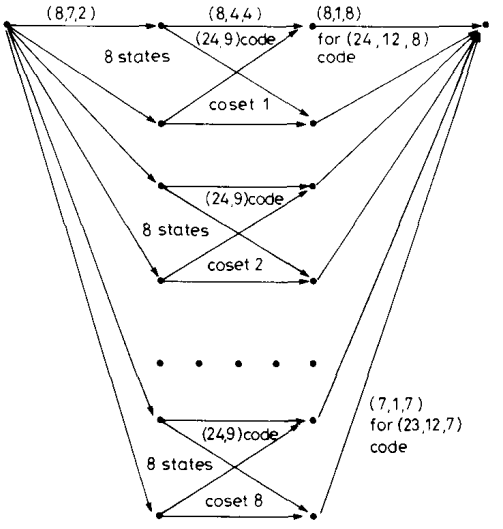

Fig. 9 *Trellis diagrams of Golay codes*

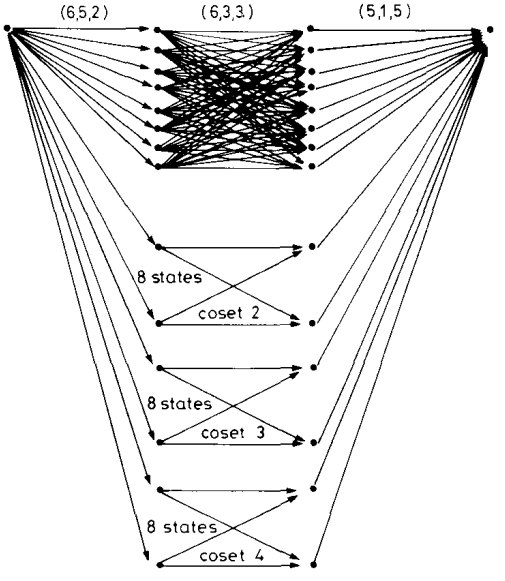

Fig. 10 *Trellis diagram of (17, 9, 5) GAC*

and differs only in number of digits which are used for labelling branches in the last column of the trellis (all branches in this column are labelled with 7-tuple codewords).

The trellis diagram of the (17, 9, 5) code has the following parameters: $N_s = 32$, $N_c = 3$, and is shown in Fig. 10. The trellis diagrams for some quasi-perfect codes are shown in Fig. 11.

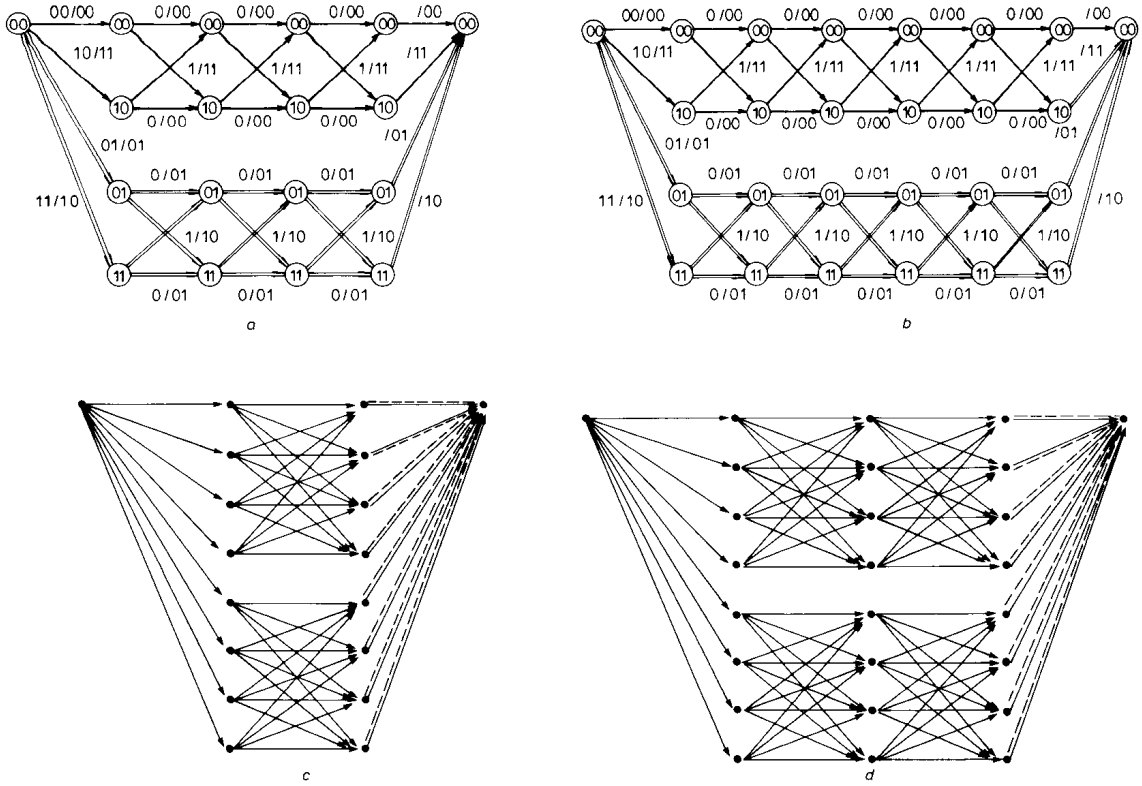

Fig. 11 *Trellis diagrams of GACs*
a (10, 5, 4) GAC *b* (14, 7, 4) GAC *c* (9, 6, 3) GAC *d* (12, 8, 3) GAC

5 References

1 BAHL, L.R., COCKE, J., JELINEK, F., and RAVIV, J.: 'Optimal decoding of linear codes for minimising symbol error rate', *IEEE Trans. Inf. Theory*, 1974, **20,** pp. 284–287
2 WOLF, J.K.: 'Efficient maximum likelihood decoding of linear block codes using a trellis', *IEEE Trans. Inf. Theory*, 1978, **24,** (1), pp. 76–80
3 MASSEY, J.: 'Foundation and methods of channel encoding'. Proceedings of international conference on *Information theory*, **65,** 18–20 Sept., 1978, Berlin
4 CONWAY, J.H., and SLOANE, N.J.: 'Soft decoding techniques for codes and lattices, including Golay code and the Leech lattice', *IEEE Trans. Inf. Theory*, 1986, **32,** (1), pp. 41–50
5 KASAMI, T., TAKATA, T., FUJIWARA, T., and LIN, S.: 'On complexity of trellis structure of linear block codes', *IEEE Trans. Inf. Theory*, 1993, **39,** (1), pp. 242–245
6 SNYDERS, J.: 'Reduced lists of error patterns for maximum likelihood soft decoding', *IEEE Trans. Inf. Theory*, 1991, **37,** (4), pp. 667–672
7 BE'ERY, Y., and SNYDERS, J.: 'Optimal soft decision decoders based on fast Haddamard transform', *IEEE Trans. Inf. Theory*, 1986, **32,** pp. 355–364
8 FORNEY, G. D.: 'Coset codes — Part 2: Binary lattices and related codes', *IEEE Trans. Inf. Theory*, 1988, **34,** (5), pp. 1152–1187
9 FORNEY, G.D., and TROTT, M.D.: 'The dynamics of group codes: state spaces, trellis diagrams, and cannonical encoders', *IEEE Trans. Inf. Theory*, 1993, **39,** (5), pp. 1491–1523
10 MUDER, D.J.: 'Minimal trellises for block codes', *IEEE Trans. Inf. Theory*, 1988, **34,** (5), pp. 1049–1053
11 BERGER, Y., and BE'ERY, Y.: 'Bounds of the trellis size of linear block codes', *IEEE Trans. Inf. Theory*, 1993, **39,** (1), pp. 203–208
12 HONARY, B., MARKARIAN, G., and FARRELL, P.: 'Generalised array codes and their trellis structure'. *Electron. Lett.*, 1993, **29,** (6), pp. 541–542
13 HONARY, B., and MARKARIAN, G.: 'Low complexity trellis decoding of Hamming codes', *Electron. Lett*, 1993, **29,** (12), pp. 1114–1116
14 HONARY, B., and MARKARIAN, G.: 'New simple encoder and trellis decoder for Golay codes', *Electron. Lett.*, 1993, **29,** (25), pp. 2170–2171
15 LIN, S., and COSTELLO, D.: 'Error control coding: fundamental and applications' (Prentice-Hall, Englewood Cliffs, NJ, 1983)
16 PETERSON, W.W., and WELDON, E.J.: 'Error correcting codes' (MIT Press, 2nd edn., 1975)
17 GOETHALS, J.M.: 'On the Golay perfect binary code', *J. Combinatorial Theory*, 1971, **11,** pp. 178–186
18 ZYABLOV, V.V., and SIDORENKO, V.: 'Bounds of complexity of trellis decoding of linear blockcodes'. Proceedings of the Swedish-Russian workshop on *Information theory*, Aug. 1993, Sweden
19 McWILLIAMS, F.J., and SLOANE, N.J.A.: 'The theory of error correcting codes'. NHPC, 1978.
20 NORDSTROM, A.W., and ROBINSON, J.P.: 'An optimum nonlinear code', *Inf. & Control*, 1967, **11,** pp. 613–616