Journal of
Statistical Theory and Practice
Volume 3, No. 2, June 2009

On 2-Repeated Burst Error Detecting Codes[1]

Luigia Berardi, *Department of Electrical and Information Engineering, University of L'Aquila, Piazzale Ernesto Pontieri, 1, Monteluco di Roio, 67100 L'Aquila, Italy. Email: berardi@ing.univaq.it*

Bal Kishan Dass, *Department of Mathematics, University of Delhi, Delhi - 110 007, India. Email: dassbk@rediffmail.com*

Rashmi Verma, *Department of Mathematics, University of Delhi, Delhi - 110 007, India. Email: riva_7@rediffmail.com*

Received: November 15, 2007 *Revised:* July 1, 2008

Abstract

There are several kinds of burst errors for which error detecting and correcting codes have been constructed. In this paper, we introduce a new kind of burst error which will be termed as 'repeated burst error'. Linear codes capable of detecting such errors have been studied. Further, codes capable of detecting and simultaneously correcting such errors have also been dealt with. The paper obtains lower and upper bounds on the number of parity-check digits required for such codes. An example of such a code has also been provided.

AMS Subject Classification: 94B20; 94B65.

Key-words: Burst error; Repeated burst error; Error detecting code; Error correcting code; Bounds.

1. Introduction

Investigations in coding theory have been made in a variety of directions but one of the most important aspects has been the detection and correction of errors. The beginning was made with the detection and correction of random errors (Hamming, 1950) and thereafter the advent of BCH codes for multiple error correction was taken up. Though there is a long history concerning the growth of the subject and many of the codes developed have found

[1]This work has been finalized in the present form during second author's visit to University of L'Aquila (Italy) under G.N.S.A.G.A. of I.N.D.A.M. program in October, 2007.

• 1559-8608/09-2/$5 + $1pp – see inside front cover

applications in numerous areas of practical interest, one of the areas of practical importance in which a parallel growth of the subject took place is that of 'burst error detecting and correcting codes'. In fact, it was observed that in many communication channels the likelihood of the occurrence of errors is more in adjacent digits rather than their occurrence in a random manner. Extending the work of Hamming (1950), Abramson (1959) developed codes which dealt with the correction of single and double adjacent errors. The work due to Fire (1959) depicted a more general concept of clustered errors, which in the literature are known as 'burst errors'. A burst of length b may be defined as follows:

Definition 1.1. A burst of length b is a vector whose only non-zero components are among some b consecutive components, the first and the last of which is non-zero.

Fire (1959) considered two kinds of bursts viz., 'open-loop burst' which are popularly refered to simply a burst as in Definition 1.1 and the other called 'closed-loop burst' defined as follows:

Definition 1.2. Let b be an integer and $x = (\xi_1, \ldots, \xi_n)$ be a vector in $V^n(q)$, a vector space of n-tuples over GF(q). If $2 \leq b \leq \frac{(n+1)}{2}$, then x is called a 'closed-loop burst vector of length b' whenever there is an i such that $1 \leq i \leq b-1$,

$$\xi_i \cdot \xi_{n-b+i+1} \neq 0, \quad \xi_{i+1} = \xi_{i+2} = \ldots = \xi_{n-b+i} = 0.$$

Stone (1961), and Bridwell and Wolf (1970) considered multiple bursts. There is yet other kind of a burst that has been considered in the literature viz., bursts due to Chien and Tang (1965). It is clear that the nature of burst errors differ from channel to channel depending upon the behaviour of channels or the kind of errors which occur in these channels. It is commonly seen that in very busy communication channels, errors repeat themselves. So is a situation when errors occur in the form of a burst. In a way, we need to consider repeated bursts and so is the necessity to develop repeated burst error detecting and correcting codes. The development of such codes will economize in the number of parity-check digits required not only in comparison with codes dealing with the detection and correction of the same number of random errors but also in comparison with the usual burst error detecting and correcting codes while considering such repeated bursts as single bursts.

In this paper, we introduce this kind of a burst which is an extension of the idea of an open-loop burst and obtain results in the form of bounds over the number of parity-check digits required for codes detecting and correcting such errors. We define a 2-repeated burst to begin with.

Definition 1.3. A 2-repeated burst of length b is a vector of length n whose only non-zero components are confined to two distinct sets of b consecutive components, the first and the last component of each set being non-zero.

For example, (000100200103400) is a 2-repeated burst of length 4 over GF(5).

It may be pointed out that with respect to the burst due to Chien and Tang (1965), Dass, Garg and Zannetti (2008) have considered a more general class of m-repeated burst. The new kind of 2-repeated burst considered in this paper is not a special case of the m-repeated burst studied therein because the bursts considered in these two papers are of different kinds.

In what follows, a linear code will be considered as a subspace of the space of all n-tuples over GF(q). The distance between two vectors shall be considered in the Hamming sense.

2. 2-Repeated Burst Error Detecting Codes

In this section, we consider linear codes that are capable to detect a 2-repeated burst of length b or less. Clearly, the patterns to be detected should not be code words. In other words, we consider codes that have no 2-repeated burst of length b or less as a code word. Firstly, we obtain a lower bound over the number of parity-check digits for such a code.

Theorem 2.1. *Any (n,k) linear code over* GF(q) *that detects any 2-repeated burst of length b or less must have atleast $2b$ parity-check digits.*

Proof. The result will be proved on the basis that no detectable error vector can be a code word.

Let V be an (n,k) linear code over GF(q). Consider a set X that has all those vectors which have their non-zero components confined to some two fixed distinct b consecutive components.

We claim that no two vectors of X can belong to the same coset of the standard array, else a code word shall be expressible as a sum or difference of two error vectors.

Assume, on the contrary, that there is a pair say x_1, x_2 in X belonging to the same coset of the standard array. Then their difference viz., $x_1 - x_2$ must be a code word. But $x_1 - x_2$ is a vector all of whose non-zero components are confined to the same two fixed b consecutive components and so is a member of X, i.e., $x_1 - x_2$ is a 2-repeated burst of length b or less, which is a contradiction.

Thus all the vectors in X must belong to distinct cosets of the standard array. The number of such vectors over GF(q) is clearly q^{2b}. Also, total number of cosets in an (n,k) linear code equals q^{n-k}, so we must have $q^{n-k} \geq q^{2b}$, i.e., $n-k \geq 2b$, which proves the result. □

In the following result, we derive another bound on the number of check digits required for the existence of such a code. The proof is based on the technique used to establish Varshamov-Gilbert-Sacks bound by constructing a parity check matrix for such a code (refer Theorem 4.7, Peterson and Weldon, 1972). This technique not only ensures the existence of such a code but also gives a method for the construction of such a code.

Theorem 2.2. *There shall always exist an (n,k) linear code over* GF(q) *that has no 2-repeated burst of length b or less as a codeword provided that*

$$q^{n-k} > q^{2(b-1)}[(n-2b+1)(q-1)+1]. \tag{2.1}$$

Proof. We shall prove the result by constructing an appropriate $(n-k) \times n$ parity-check matrix $H = [h_1, h_2, \ldots, h_n]$ for the desired code. Choose any non-zero $(n-k)$-tuple as the first column h_1 of H. Suppose that we have selected the first $j-1$ columns $h_1, h_2, \ldots, h_{j-1}$ of H suitably. We lay down the condition to add the j-th column h_j as follows:

h_j should not be a linear combination of the immediately preceding $b-1$ or fewer columns of H together with any b or fewer consecutive columns from amongst the first $j-b$ columns $h_1, h_2, \ldots, h_{j-b}$.

In other words,

$$\begin{aligned} h_j \neq\ & (\alpha_1 h_{j-b+1} + \alpha_2 h_{j-b+2} + \ldots + \alpha_{b-1} h_{j-1}) \\ & + (\beta_1 h_i + \beta_2 h_{i+1} + \ldots + \beta_b h_{i+b-1}) \end{aligned} \tag{2.2}$$

where $\alpha_i, \beta_i \in \mathrm{GF}(q)$ and $i + b - 1 \leq j - b$.

This condition ensures that no 2-repeated burst of length b or less will be a code word.

The above inequality (2.2) is the same as the condition laid down for the correction of single (usual) burst of length b or less (refer to Theorem 4.17, Peterson and Weldon, 1972) leading to the same computational result and is therefore being omitted. □

Remark 2.1. In view of the fact that the result obtained in Theorem 2.2 is same as the result for the correction of bursts of length b or less, such a code can serve dual purpose, i.e., it can either be used to correct bursts of length b or less, or can be used to detect a 2-repeated burst of length b or less.

3. Simultaneous Detection and Correction of 2-Repeated Burst errors

In the following, we consider linear codes which are capable to detect and correct simultaneously 2-repeated bursts and obtain a necessary condition over the number of parity-checks required for such a code.

Theorem 3.1. *Any (n,k) linear code over $\mathrm{GF}(q)$ that corrects all 2-repeated bursts of length b or less must have atleast $4b$ parity-check digits. Further, if the code corrects all 2-repeated bursts of length b or less and simultaneously detects 2-repeated bursts of length d or less $(d \geq b)$, then the code must have at least $2(b+d)$ parity-check digits.*

Proof. Consider a burst of length $4b$. Such a vector is expressible as a sum or difference of two vectors each of which is a 2-repeated burst of length b or less. These component vectors must belong to different cosets of the standard array because both such errors are correctable errors. Accordingly, such a vector viz., a burst of length $4b$ or less cannot be a code word. In view of Theorem 2.1, such a code must have at least $4b$ parity-check digits.

Further, consider a burst of length $2(b+d)$. Such a vector cannot be a code word because it is always expressible as a sum or difference of two vectors, one of which is a 2-repeated burst of length b or less and the other is a 2-repeated burst of length d or less. As earlier, any pair of such component vectors cannot belong to the same coset of the standard array and so a burst of length $2(b+d)$ cannot be a code word. Therefore, the code must have atleast $2(b+d)$ parity-check digits. □

We conclude the paper with an example.

Example 3.1. Consider a $(9,3)$ binary code with parity-check matrix

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}_{6\times 9}$$

This matrix has been constructed by the synthesis procedure outlined in the proof of Theorem 2.2 by taking $b = 3$. It can be seen from Table 3.1 that the syndromes of the different

2-repeated bursts of length 3 or less are non-zero, showing thereby that the code that is the null space of this matrix detects all 2-repeated bursts of length 3 or less.

Table 3.1

Error vectors									Syndromes					
1	0	0	0	0	0	0	0	0	1	0	0	0	0	0
1	0	0	1	0	0	0	0	0	1	0	0	1	0	0
1	0	0	0	1	0	0	0	0	1	0	0	0	1	0
1	0	0	0	0	1	0	0	0	1	0	0	0	0	1
1	0	0	0	0	0	1	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	1	0	1	1	0	0	1	0
1	0	0	0	0	0	0	0	1	1	0	1	0	0	1
1	0	0	1	1	0	0	0	0	1	0	0	1	1	0
1	0	0	0	1	1	0	0	0	1	0	0	0	1	1
1	0	0	0	0	1	1	0	0	0	0	0	1	0	1
1	0	0	0	0	0	1	1	0	0	1	0	1	1	0
1	0	0	0	0	0	0	1	1	1	1	1	0	1	1
1	0	0	1	0	1	0	0	0	1	0	0	1	0	1
1	0	0	0	1	0	1	0	0	0	0	0	1	1	0
1	0	0	0	0	1	0	1	0	1	1	0	0	1	1
1	0	0	0	0	0	1	0	1	0	0	1	1	0	1
1	0	0	1	1	1	0	0	0	1	0	0	1	1	1
1	0	0	0	1	1	1	0	0	0	0	0	1	1	1
1	0	0	0	0	1	1	1	0	0	1	0	1	1	1
1	0	0	0	0	0	1	1	1	0	1	1	1	1	1
1	1	0	0	0	0	0	0	0	1	1	0	0	0	0
1	1	0	1	0	0	0	0	0	1	1	0	1	0	0
1	1	0	0	1	0	0	0	0	1	1	0	0	1	0
1	1	0	0	0	1	0	0	0	1	1	0	0	0	1
1	1	0	0	0	0	1	0	0	0	1	0	1	0	0
1	1	0	0	0	0	0	1	0	1	0	0	0	1	0
1	1	0	0	0	0	0	0	1	1	1	1	0	0	1
1	1	0	1	1	0	0	0	0	1	1	0	1	1	0
1	1	0	0	1	1	0	0	0	1	1	0	0	1	1
1	1	0	0	0	1	1	0	0	0	1	0	1	0	1
1	1	0	0	0	0	1	1	0	0	0	0	1	1	0
1	1	0	0	0	0	0	1	1	1	0	1	0	1	1
1	1	0	1	0	1	0	0	0	1	1	0	1	0	1
1	1	0	0	1	0	1	0	0	0	1	0	1	1	0
1	1	0	0	0	1	0	1	0	1	0	0	0	1	1
1	1	0	0	0	0	1	0	1	0	1	1	1	0	1
1	1	0	1	1	1	0	0	0	1	1	0	1	1	1
1	1	0	0	1	1	1	0	0	0	1	0	1	1	1
1	1	0	0	0	1	1	1	0	0	0	0	1	1	1
1	1	0	0	0	0	1	1	1	0	0	1	1	1	1
1	0	1	0	0	0	0	0	0	1	0	1	0	0	0

(Contd.)

Error vectors									Syndromes					
1	0	1	1	0	0	0	0	0	1	0	1	1	0	0
1	0	1	0	1	0	0	0	0	1	0	1	0	1	0
1	0	1	0	0	1	0	0	0	1	0	1	0	0	1
1	0	1	0	0	0	1	0	0	0	0	1	1	0	0
1	0	1	0	0	0	0	1	0	1	1	1	0	1	0
1	0	1	0	0	0	0	0	1	1	0	0	0	0	1
1	0	1	1	1	0	0	0	0	1	0	1	1	1	0
1	0	1	0	1	1	0	0	0	1	0	1	0	1	1
1	0	1	0	0	1	1	0	0	0	0	1	1	0	1
1	0	1	0	0	0	1	1	0	0	1	1	1	1	0
1	0	1	0	0	0	0	1	1	1	1	0	0	1	1
1	0	1	1	0	1	0	0	0	1	0	1	1	0	1
1	0	1	0	1	0	1	0	0	0	0	1	1	1	0
1	0	1	0	0	1	0	1	0	1	1	1	0	1	1
1	0	1	0	0	0	1	0	1	0	0	0	1	0	1
1	0	1	1	1	1	0	0	0	1	0	1	1	1	1
1	0	1	0	1	1	1	0	0	0	0	1	1	1	1
1	0	1	0	0	1	1	1	0	0	1	1	1	1	1
1	0	1	0	0	0	1	1	1	0	1	0	1	1	1
1	1	1	0	0	0	0	0	0	1	1	1	0	0	0
1	1	1	1	0	0	0	0	0	1	1	1	1	0	0
1	1	1	0	1	0	0	0	0	1	1	1	0	1	0
1	1	1	0	0	1	0	0	0	1	1	1	0	0	1
1	1	1	0	0	0	1	0	0	0	1	1	1	0	0
1	1	1	0	0	0	0	1	0	1	0	1	0	1	0
1	1	1	0	0	0	0	0	1	1	1	0	0	0	1
1	1	1	1	1	0	0	0	0	1	1	1	1	1	0
1	1	1	0	1	1	0	0	0	1	1	1	0	1	1
1	1	1	0	0	1	1	0	0	0	1	1	1	1	0
1	1	1	0	0	0	1	1	0	0	0	0	1	1	1
1	1	1	0	0	0	0	1	1	1	1	0	0	1	1
1	1	1	1	0	1	0	0	0	0	1	1	1	1	0
1	1	1	0	1	0	1	0	0	1	0	1	1	0	1
1	1	1	0	0	1	0	1	0	0	1	0	1	0	1
1	1	1	0	0	0	1	0	1	1	1	1	1	1	1
1	1	1	1	1	1	0	0	0	0	1	1	1	1	1
1	1	1	0	1	1	1	0	0	0	0	1	1	1	1
1	1	1	0	0	1	1	1	0	0	0	0	1	1	1
1	1	1	0	0	0	1	1	1	0	1	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
0	1	0	0	0	1	0	0	0	0	1	0	0	1	0
0	1	0	0	0	0	1	0	0	0	1	0	0	0	1
0	1	0	0	0	0	0	1	0	1	1	0	1	0	0

(*Contd.*)

Error vectors									Syndromes					
0	1	0	0	0	0	0	1	0	0	0	0	0	1	0
0	1	0	0	0	0	0	0	1	0	1	1	0	0	1
0	1	0	0	1	1	0	0	0	0	1	0	0	1	1
0	1	0	0	0	1	1	0	0	1	1	0	1	0	1
0	1	0	0	0	0	1	1	0	1	0	0	1	1	0
0	1	0	0	0	0	0	1	1	0	0	1	0	1	1
0	1	0	0	1	0	1	0	0	1	1	0	1	1	0
0	1	0	0	0	1	0	1	0	0	0	0	0	1	1
0	1	0	0	0	0	1	0	1	1	1	1	1	0	1
0	1	0	0	1	1	1	0	0	1	1	0	1	1	1
0	1	0	0	0	1	1	1	0	1	0	0	1	1	1
0	1	0	0	0	0	1	1	1	1	0	1	1	1	1
0	1	1	0	0	0	0	0	0	0	1	1	0	0	0
0	1	1	0	1	0	0	0	0	0	1	1	0	1	0
0	1	1	0	0	1	0	0	0	0	1	1	0	0	1
0	1	1	0	0	0	1	0	0	1	1	1	1	0	0
0	1	1	0	0	0	0	1	0	0	0	1	0	1	0
0	1	1	0	0	0	0	0	1	0	1	0	0	0	1
0	1	1	0	1	1	0	0	0	0	1	1	0	1	1
0	1	1	0	0	1	1	0	0	1	1	1	1	0	1
0	1	1	0	0	0	1	1	0	1	0	1	1	1	0
0	1	1	0	0	0	0	1	1	0	0	0	0	1	1
0	1	1	0	1	0	1	0	0	1	1	1	1	1	0
0	1	1	0	0	1	0	1	0	0	0	1	1	0	1
0	1	1	0	0	0	1	0	1	1	1	0	1	0	1
0	1	1	0	1	1	1	0	0	1	1	1	1	1	1
0	1	1	0	0	1	1	1	0	1	0	1	1	1	1
0	1	1	0	0	0	1	1	1	1	0	0	1	1	1
0	1	0	1	0	0	0	0	0	0	1	0	1	0	0
0	1	0	1	1	0	0	0	0	0	1	0	1	1	0
0	1	0	1	0	1	0	0	0	0	1	0	1	0	1
0	1	0	1	0	0	1	0	0	1	1	0	0	0	0
0	1	0	1	0	0	0	1	0	0	0	0	1	1	0
0	1	0	1	0	0	0	0	1	0	1	1	1	0	1
0	1	0	1	1	1	0	0	0	0	1	0	1	1	1
0	1	0	1	0	1	1	0	0	1	1	0	0	0	1
0	1	0	1	0	0	1	1	0	1	0	0	0	1	0
0	1	0	1	0	0	0	1	1	0	0	1	1	1	1
0	1	0	1	1	0	1	0	0	1	1	0	0	1	0
0	1	0	1	0	1	0	1	0	0	0	0	1	1	1
0	1	0	1	0	0	1	0	1	1	1	0	0	1	1
0	1	0	1	1	1	1	0	0	1	1	0	0	1	1
0	1	0	1	0	1	1	1	0	1	0	0	0	1	1

(*Contd.*)

Error vectors									Syndromes					
0	1	0	1	0	0	1	1	1	1	0	1	0	1	1
0	1	1	1	0	0	0	0	0	0	1	1	1	0	0
0	1	1	1	1	0	0	0	0	0	1	1	1	1	0
0	1	1	1	0	1	0	0	0	0	1	1	1	0	1
0	1	1	1	0	0	1	0	0	1	1	1	0	0	0
0	1	1	1	0	0	0	1	0	0	0	1	1	1	0
0	1	1	1	0	0	0	0	1	0	1	0	1	0	1
0	1	1	1	1	1	0	0	0	0	1	1	1	1	1
0	1	1	1	0	1	1	0	0	1	1	1	0	0	1
0	1	1	1	0	0	1	1	0	1	0	1	0	1	0
0	1	1	1	0	0	0	1	1	0	0	0	1	1	1
0	1	1	1	1	0	1	0	0	1	1	1	0	1	0
0	1	1	1	0	1	0	1	0	0	0	1	1	1	1
0	1	1	1	0	0	1	0	1	1	1	0	0	0	1
0	1	1	1	1	1	1	0	0	1	1	1	0	1	1
0	1	1	1	0	1	1	1	0	1	0	1	0	1	1
0	1	1	1	0	0	1	1	1	1	0	0	0	1	1
0	0	1	0	0	0	0	0	0	0	0	1	0	0	0
0	0	1	0	0	1	0	0	0	0	0	1	0	0	1
0	0	1	0	0	0	1	0	0	1	0	1	1	0	0
0	0	1	0	0	0	0	1	0	0	1	1	0	1	0
0	0	1	0	0	0	0	0	1	0	0	0	0	0	1
0	0	1	0	0	1	1	0	0	1	0	1	1	0	1
0	0	1	0	0	0	1	1	0	1	1	1	1	1	0
0	0	1	0	0	0	0	1	1	0	1	0	0	1	1
0	0	1	0	0	1	0	1	0	0	1	1	0	1	1
0	0	1	0	0	0	1	0	1	1	0	0	0	1	1
0	0	1	0	0	1	1	1	0	1	1	1	1	1	1
0	0	1	0	0	0	1	1	1	1	1	0	1	1	1
0	0	1	1	0	0	0	0	0	0	0	0	1	1	0
0	0	1	1	0	1	0	0	0	0	0	0	1	1	1
0	0	1	1	0	0	1	0	0	1	0	1	0	0	0
0	0	1	1	0	0	0	1	0	0	1	1	1	1	0
0	0	1	1	0	0	0	0	1	0	0	0	0	1	1
0	0	1	1	0	1	1	0	0	1	0	1	0	0	1
0	0	1	1	0	0	1	1	0	1	1	1	0	1	0
0	0	1	1	0	0	0	1	1	0	1	0	1	1	1
0	0	1	1	0	1	0	1	0	1	0	0	0	0	1
0	0	1	1	0	0	1	0	1	1	1	1	0	1	1
0	0	1	1	0	1	1	1	0	1	1	0	0	1	1
0	0	1	0	1	0	0	0	0	0	0	1	0	1	0
0	0	1	0	1	1	0	0	0	0	0	1	0	1	1

(Contd.)

Error vectors									Syndromes					
0	0	1	0	1	0	1	0	0	1	0	1	1	1	0
0	0	1	0	1	0	0	1	0	0	1	1	0	0	0
0	0	1	0	1	0	0	0	1	0	0	0	0	1	1
0	0	1	0	1	1	1	0	0	1	0	1	1	1	1
0	0	1	0	1	0	1	1	0	1	1	1	1	0	0
0	0	1	0	1	0	0	1	1	0	1	0	0	0	1
0	0	1	0	1	1	0	1	0	0	1	1	0	0	1
0	0	1	0	1	0	1	0	1	1	0	0	1	1	1
0	0	1	0	1	1	1	1	0	1	1	1	1	0	1
0	0	1	0	1	0	1	1	1	1	1	0	1	0	1
0	0	1	1	1	0	0	0	0	0	0	1	1	1	0
0	0	1	1	1	1	0	0	0	0	0	1	1	1	1
0	0	1	1	1	0	1	0	0	1	0	1	0	1	0
0	0	1	1	1	0	0	1	0	0	1	1	1	0	0
0	0	1	1	1	0	0	0	1	0	0	0	1	1	1
0	0	1	1	1	1	1	0	0	1	0	1	0	1	1
0	0	1	1	1	0	1	1	0	1	1	1	0	0	0
0	0	1	1	1	0	0	1	1	0	1	0	1	0	1
0	0	1	1	1	1	0	1	0	0	1	1	1	0	1
0	0	1	1	1	0	1	0	1	1	0	0	0	1	1
0	0	1	1	1	1	1	1	0	1	1	1	0	0	1
0	0	1	1	1	0	1	1	1	1	1	0	0	0	1
0	0	0	1	0	0	0	0	0	0	0	0	1	0	0
0	0	0	1	0	0	1	0	0	1	0	0	0	0	0
0	0	0	1	0	0	0	1	0	0	1	0	1	1	0
0	0	0	1	0	0	0	0	1	0	0	1	1	0	1
0	0	0	1	0	0	1	1	0	1	1	0	0	1	0
0	0	0	1	0	0	0	1	1	0	1	1	1	1	1
0	0	0	1	0	0	1	0	1	1	0	1	0	0	1
0	0	0	1	0	0	1	1	1	1	1	1	0	1	1
0	0	0	1	1	0	0	0	0	0	0	0	1	1	0
0	0	0	1	1	0	1	0	0	1	0	0	0	1	0
0	0	0	1	1	0	0	1	0	0	1	0	1	0	0
0	0	0	1	1	0	0	0	1	0	0	1	1	1	1
0	0	0	1	1	0	1	1	0	1	1	0	0	0	0
0	0	0	1	1	0	0	1	1	0	1	1	0	1	1
0	0	0	1	1	0	1	0	1	1	0	1	0	1	1
0	0	0	1	1	0	1	1	1	1	1	1	0	0	1
0	0	0	1	0	1	0	0	0	0	0	0	1	0	1
0	0	0	1	0	1	1	0	0	1	0	0	0	0	1
0	0	0	1	0	1	0	1	0	0	1	0	1	1	1
0	0	0	1	0	1	0	0	1	0	0	1	1	0	0
0	0	0	1	0	1	1	1	0	1	1	0	0	1	1
0	0	0	1	0	1	0	1	1	0	1	1	1	1	0

(*Contd.*)

Error vectors									Syndromes					
0	0	0	1	0	1	1	0	1	1	0	1	0	0	0
0	0	0	1	0	1	1	1	1	1	1	1	0	1	0
0	0	0	1	1	1	0	0	0	0	0	0	1	1	1
0	0	0	1	1	1	1	0	0	1	0	0	0	1	1
0	0	0	1	1	1	0	1	0	0	1	0	1	0	1
0	0	0	1	1	1	0	0	1	0	0	1	1	1	0
0	0	0	1	1	1	1	1	0	1	1	0	0	0	1
0	0	0	1	1	1	0	1	1	0	1	1	1	0	0
0	0	0	1	1	1	1	0	1	1	0	1	0	1	0
0	0	0	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	1	0	0
0	0	0	0	1	1	1	1	0	1	1	0	1	0	1
0	0	0	0	1	1	1	0	1	1	0	1	1	1	0
0	0	0	0	1	1	1	0	0	1	0	0	1	1	1
0	0	0	0	1	1	0	1	1	0	1	1	0	0	0
0	0	0	0	1	1	0	1	0	0	1	0	0	0	1
0	0	0	0	1	1	0	0	1	0	0	1	0	1	0
0	0	0	0	1	1	0	0	0	0	0	0	0	1	1
0	0	0	0	1	0	1	1	1	1	1	1	1	0	1
0	0	0	0	1	0	1	1	0	1	1	0	1	0	0
0	0	0	0	1	0	1	0	1	1	0	1	1	1	1
0	0	0	0	1	0	1	0	0	1	0	0	1	1	0
0	0	0	0	1	0	0	1	1	0	1	1	0	0	1
0	0	0	0	1	0	0	1	0	0	1	0	0	0	0
0	0	0	0	1	0	0	0	1	0	0	1	0	1	1
0	0	0	0	1	0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	1	1	1	1	1	1	1	1	1	0
0	0	0	0	0	1	1	1	0	1	1	0	1	1	1
0	0	0	0	0	1	1	0	1	1	0	1	1	0	0
0	0	0	0	0	1	1	0	0	1	0	0	1	0	1
0	0	0	0	0	1	0	1	1	0	1	1	0	1	0
0	0	0	0	0	1	0	1	0	0	1	0	0	1	1
0	0	0	0	0	1	0	0	1	0	0	1	0	0	0
0	0	0	0	0	1	0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
0	0	0	0	0	0	1	1	0	1	1	0	1	1	0
0	0	0	0	0	0	1	0	1	1	0	1	1	0	1
0	0	0	0	0	0	1	0	0	1	0	0	1	0	0
0	0	0	0	0	0	0	1	1	0	1	1	0	1	1
0	0	0	0	0	0	0	1	0	0	1	0	0	1	0
0	0	0	0	0	0	0	0	1	0	0	1	0	0	1

Acknowledgements

The authors thank the anonymous referees for their critical examination which has helped us in revising the paper in the present form.

References

Abramson, N.M., 1959. A class of systematic codes for non-independent errors. *IRE Trans. on Information Theory*, IT-5 (4), 150–157.

Bridwell, J.D., Wolf, J.K., 1970. Burst distance and multiple-burst correction. *Bell System Tech. J.*, 49, 889–909.

Campopiano C.N. (1962), Bounds on burst error correcting codes, *IRE Trans.*, IT-8, pp. 257–259.

Chien, R.T., Tang, D.T., 1965, On definitions of a burst. *IBM Journal of Research and Development*, 9 (4) (July), 292–293.

Dass, B. K., Garg, P., Zannetti, M., 2008. Some combinatorial aspects of *m*-repeated burst error detecting codes. *Journal of Statistical Theory and Practice*, 2 (4) (December), 707–711.

Fire, P., 1959. A class of multiple-error-correcting binary codes for non-independent errors. *Sylvania Report RSL-E-2*, Sylvania Reconnaissance Systems Laboratory, Mountain View, Calif.

Hamming, R.W., 1950. Error-detecting and error-correcting codes, *Bell System Technical Journal*, 29 (April), 147–160.

Peterson, W.W., Weldon, E.J., Jr., 1972. *Error-Correcting Codes*. 2nd edition, The MIT Press, Mass.

Reiger, S.H., 1960. Codes for the correction of "clustered errors". *IRE Trans. Inform. Theory*, IT-6 (March), 16–21.

Stone, J.J., 1961. Multiple burst error correction. *Information and Control*, 4, 324–331.