Hamming Codes Are Rate-Efficient Array Codes

Esteban L. Vallés[1], Andres I. Vila Casado[1], Mario Blaum[2], J. Villasenor[1] and Richard D. Wesel[1]
[1]Department of Electrical Engineering, University of California, Los Angeles, CA 90095-1594,USA.
Email: {evalles, avila, wesel}@ee.ucla.edu, villa@icsl.ucla.edu
[2]Hitachi Global Storage Technologies.650 Harry Road. San Jose, CA, 95120,USA.
Email: Mario.Blaum@hitachigst.com.

ABSTRACT

Array codes are error-correcting codes of very low complexity that were initially used for burst and erasure correction in Redundant Arrays of Inexpensive Disks (RAID) architectures and other storage applications. The structure of these codes allows a very simple encoding and decoding mechanism. Although they are very high-rate codes, they do not achieve the maximum possible rate given their design constraints. In fact Hamming codes maximize the possible rate given these design constraints. This paper compares the rate and complexity of array codes when compared to Hamming codes.

Keywords: Array codes, Low-density parity-check codes, Hamming codes, burst correction, RAID architectures, disk arrays.

I. INTRODUCTION

Burst error correcting codes are used in many fields such as multi-track storage, satellite communications and disk arrays. Array codes [1], Fire codes [2] and Reed-Solomon [3] codes are well-known codes that have good burst-error-correcting capabilities. If an error-correcting code requires operations over a finite field (as in the case of Reed-Solomon codes) complexity in the encoder and decoder architecture is increased. Array code encoding and decoding only requires the use of simple bit manipulation, reducing the overall complexity. Therefore when implementation simplicity is an issue and hardware efficient encoders and decoders are needed, array codes can be the most attractive option.

In this paper a method for increasing the rate of an array code is presented based on the relation between Hamming codes over $GF(q = 2^{(m-1)})$ and array codes. This is equivalent to adding columns to the parity check matrix of a traditional array code. The total number of rows of an array code, $[n-k]$, is unchanged. The increase in rate is significant for short code lengths. For a code length $n = 42$ (that corresponds to $m = 7$) the rate gain is 35.6%. As n increases and the code rate approaches one, the gain in rate becomes much smaller. For $n = 506$, which corresponding to $m = 23$, the rate gain falls to 9.52%.

In the next section an introduction to array codes is presented. A description of a method for obtaining higher-rate codes appears in Section III. The encoding and decoding algorithms are presented in Section IV. Concluding remarks are made in Section V.

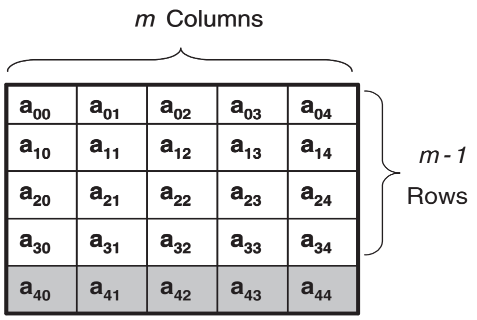

Fig. 1. Array code for $m = 5$. The grey shaded row in the array is also known as the “Imaginary row”.

$$H = \begin{array}{cccc|cccc|cccc|cccc|cccc}
a_{00} & a_{10} & a_{20} & a_{30} & a_{01} & a_{11} & a_{21} & a_{31} & a_{02} & a_{12} & a_{22} & a_{32} & a_{03} & a_{13} & a_{23} & a_{33} & a_{04} & a_{14} & a_{24} & a_{34} \\
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0 \\
0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0 \\
0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0 \\
0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1 \\
1&0&0&0&0&0&0&0&0&0&0&1&0&0&1&0&0&1&0&0 \\
0&1&0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&0&1&0 \\
0&0&1&0&0&1&0&0&1&0&0&0&0&0&0&0&0&0&0&1 \\
0&0&0&1&0&0&1&0&0&1&0&0&1&0&0&0&0&0&0&0
\end{array}$$

Fig. 2. Parity-check matrix for the array code of Fig.1.

II. ARRAY CODES

Array codes are systematic codes whose codewords are rectangular arrays of dimension $[(m-1) \times m]$ where m is a prime number. An example array with $m = 5$ is depicted in Fig. 1. These codes have minimum (column) distance equal to 3 if and only if m is a prime number (i.e., It can either correct two erased columns or one column in error). Error-correcting codes are defined in terms of the total number of information bits k, and the codeword length n. The parity-check matrix H of a block code has dimensions $[(n-k) \times n]$. For an array code both $n = (m) \cdot (m-1)$ and $k = (m-2) \cdot (m-1)$ are uniquely determined by m and so are the dimensions of H. The resulting rate of an array code is $R = k/n = (m-2)/m$. Fig. 2 shows a parity-check matrix for the case of $m = 5$. The parity-check matrix is sparse in general, which allows array codes to be described as low-density parity-check (LDPC) codes [4], [5], [6].

A. Burst Error Correction

Two different types of burst errors can be corrected using array codes. The first case consists of correcting bursts of length $L \leq (m-1)$ that occur within one column of the array. This type of error is a *phased error burst* since the burst cannot spread across columns of the array. Array codes can correct all error bursts of this type [7]. The second type of burst is a *non-phased error burst*. In this case the only constraint is that the length of the burst is $L \leq (m-1)$. As the name indicates, errors can start on one column of the array and propagate to a neighboring column. In [8] it is shown that although array codes cannot correct all non-phased bursts, the probability of finding a burst that cannot be corrected decreases with m.

Array codes were originally designed to be used in Redundant Arrays of Inexpensive Disks (RAID) architectures [9] or multi-track tape recording for fixing errors or erasures occurring in one or more disks of the array. In RAID-3 or RAID-4 type of architectures, a number of disks carry data and one or more disks carry parity information.

We assume, for illustrative purposes, that each column in the array represents a disk. One disk stores the column-wise parity of the data disks and the rest carry data information. Let us further assume that even parity is used. If one disk is *erased*, the lost information can be recovered using the remaining disks and knowing that the horizontal parity of the array has to be even. On the other hand, when one disk is read in *error*, more parity information is required in order to recover our original data. In general a code that can correct s errors can correct $2s$ erasures.

The ways in which the additional parity columns are encoded define different types of array codes like the ones presented by Blaum and Roth in [7] or the later EvenOdd codes [10]. This family of codes uses one column of the array to store horizontal (slope=0) parity and uses an additional column to store parity bits computed along diagonals (slope=1). It is also possible to create a lower rate code and add an additional diagonal parity column with slope=2 [11]. However, in this paper we will only consider array codes with two parity columns of slope 0 and 1.

In Blaum-Roth codes [7] the two parity columns are dependent. On the other hand EvenOdd codes have independent horizontal and diagonal parities. Horizontal parity is always even while diagonal parity can be even or odd depending on the data stored in the array. When data bits are updated, because of the dependence between parity columns, Blaum-Roth codes require a higher number of updates in the parity columns of the array than EvenOdd codes. From this point forward when referencing array codes, it will be assumed that the code has a Blaum-Roth structure [7].

Having parity equations among diagonals of different slopes is a very important characteristic of array codes, since it defines the encoding and decoding algorithms. We assume an $(m-1) \cdot m$ array A such that two columns carry parity information, like the one shown in Fig. 1. The code has even parity on rows and on diagonals (with a toroidal topology). There is an extra row added at the bottom of the array,that is often referred to as the *imaginary row*. This row does not carry any information and is assumed to have all zeros. As will be shown shortly, the reason for its existence is to solve an indexing problem on the diagonal parity equations (1) and (2). These equations are as follows:

- Horizontal Parity:

$$\sum_{j=0}^{m-1} a_{i,j} = 0 \qquad 0 \leq i \leq (m-2), \tag{1}$$

- Diagonal Parity:

$$\sum_{l=0}^{m-1} a_{\langle j-l \rangle_m, l} = 0 \qquad 0 \leq j \leq (m-1). \tag{2}$$

where $\langle k \rangle_n$ denotes k modulo n. In order to have a better understanding of the diagonal parity equation and the reason to define the imaginary row, the equation for j=1 for the array in Fig. 1 will be expanded:

$$a_{\langle 1-0 \rangle_5,0} + a_{\langle 1-1 \rangle_5,1} + a_{\langle 1-2 \rangle_5,2} + a_{\langle 1-3 \rangle_5,3} + a_{\langle 1-4 \rangle_5,4} =$$
$$a_{1,0} + a_{0,1} + a_{4,2} + a_{3,3} + a_{2,4} = 0. \tag{3}$$

Note that if the imaginary row was not defined, there would be an indexing problem since the entry $a_{4,2}$ does not belong to the original array.

B. Syndrome Decoding Analysis

A syndrome-decoding analysis provides insight into the efficiency of array codes. Syndrome decoding of a block code consists of multiplying the received vector r by the parity check matrix. In the event of a phased error burst, the resulting syndrome vector $s = r \cdot H^T$ is unique and contains all the available information of the error event. For the case of non-phased errors, unique syndrome decoding of array codes cannot always be accomplished, since bursts that occur on different positions of the array sometimes yield the same syndrome.

The dimensions of r, H and s are as follows:

- $dim(r) = n = m \cdot (m-1)$
- $dim(H) = (n-k, n) = (2 \cdot (m-1), m \cdot (m-1))$
- $dim(s) = n - k = 2 \cdot (m-1)$.

Thus there are $2^{2(m-1)}$ distinct syndromes, which implies that at most $2^{2(m-1)}$ distinct error events (including the "no-error" error event) can be corrected for array codes. However, for array codes there are only $m \cdot (2^{(m-1)})$ phased errors possible. Hence array codes have some syndromes (and hence some error correction capability) left over.

Raphaeli pointed out in [8] that array codes can correct some non-phased error bursts. This is possible by making use of left over syndromes. Correcting non-phased error bursts might be beneficial. However, if the specific purpose of the code is to correct only phased error bursts, the non-phased correction capability introduces a needless reduction in rate.

C. Decoding Algorithm

Observe that the parity check matrix can be divided into an upper and a lower part, both of dimension $((m-1), m \cdot (m-1))$. The upper part consists of sub-matrices that are equal to identity matrices with columns being repeated every m positions. When syndrome decoding is performed, the first $m-1$ bits of the syndrome are the exact received burst error pattern.

The decoding algorithm uses the remaining $m-1$ bits to determine the position of the error burst within the packet. The bottom part of the parity check matrix, is formed by m square sub-matrices of dimension $m-1$. When referring to a particular sub-matrix i in the bottom half of H, notation $H_{BOT}(i)$ will be used. The traditional method for decoding array codes consists of extracting the first $m-1$ bits from the syndrome, create a vector with these $m-1$ bits plus a zero added at the end,

$$T = \left[\begin{array}{ccccc} s(1) & s(2) & \ldots & s(m-1) & 0 \end{array} \right], \tag{4}$$

and apply left cyclic shifts to this vector until the first $m-1$ bits of the shifted vector match the last $m-1$ bits of the syndrome s. The number of cyclic shifts applied is equal to the column of the array code where the error burst occurred. Let $T^{(i)}$ indicate T left-shifted i times. The bit position where the detected error burst starts is equal to $i \cdot (m-1)$. Algorithm 1 summarizes the procedures mentioned above.

Algorithm 1 Array Code Decoding Algorithm

```
1: $s = r \cdot H^T$
2: Burst-Type=$s(1 : m-1)$
3: $T =$ [Burst-Type 0]
4: for $i = 1$ to $m$ do
5:     if $T^{(i-1)}[1 : m-1] = s[m : end]$ then
6:         Position =i
7:         break
8:     end if
9: end for
```

III. Hamming Codes as Array codes

Hamming codes were the first major class of linear binary codes designed for error correction [12]. The parameters for the family of binary Hamming codes are typically expressed as a function of a single integer $m_h \geq 2$, not necessarily prime. A Hamming code on $GF(2)$ has code length $n = 2^{m_h} - 1$, message length $k = 2^{m_h} - 1 - m_h$, redundancy $n - k = m_h$ and error correcting capability $t = 1$ bit. The parity-check matrices for binary Hamming codes has all nonzero binary m_h-tuples as columns.

To construct non-binary Hamming codes the same approach is used. There are exactly $(q^{m_h} - 1)$ distinct nonzero $q-$ary m_h-tuples, but not all pairs of these m_h-tuples are linearly independent. For each $q-$ary m_h-tuple there are $(q-1)$ distinct nonzero m_h-tuples that are multiples of that m_h-tuple, pairs of which are clearly dependent. The $q-$ary Hamming code parity-check matrix H is constructed by selecting exactly one m_h-tuple from each set of multiples. This can be done by selecting as columns of H all distinct $q-$ary m_h-tuples for which the uppermost nonzero element is 1. The parity-check matrix thus has $n = (q^{\tilde{m}} - 1)/(q-1)$ columns and defines a $q-$ary Hamming code with $k = (q^{m_h} - 1)/(q-1) - m_h$ and $t = 1$ symbol of m_h bits.

TABLE I
Dimensions of highest-rate phased-error-burst correcting codes for burst size m and redundancy $2(m-1)$. Second row is expressed in units of bits and the third row has units of sub-matrices.

m	3	5	7	11
(k, n)	$(6, 10)$	$(60, 68)$	$(378, 390)$	$(10230, 10250)$
$(M_{SB} - 2, M_{SB})$	$(3, 5)$	$(15, 17)$	$(63, 65)$	$(1023, 1025)$

Careful analysis of the 4x4 sub-matrices in the bottom half of Fig.2 reveals that other possible 4x4 sub-matrices will yield as yet unused values of $s[m : end]$ for a given error burst. This means that more sub-matrices can be added to the parity-check matrix, obtaining a higher rate code, while still being able to uniquely decode phased error bursts. The maximum number of phased error bursts of length L that can be corrected is

$$Ep = m \cdot (2^L - 1) \leq m \cdot (2^{(m-1)} - 1) \tag{5}$$

which is clearly much smaller than the $2^{2(m-1)} - 1$ possible non-zero syndromes that can be generated with $n-k$ bits. Let M_{SB} be the maximum number of sub-matrices that a parity check matrix can have, then

$$M_{SB} = \left(\frac{2^{2(m-1)} - 1}{2^{m-1} - 1} \right). \tag{6}$$

Table I shows the dimensions of array codes for different values of m. The second row shows the relationship between input and output bits, while the third row expresses the same relation in units of sub-matrices instead of bits. Since each sub-matrix in an array code is a square matrix of size $(m-1)$, the third row is obtained by diving the second row by $(m-1)$.

Consider the case of $m = 5$ as an example to describe the modified array codes as Hamming codes. For this example the field where the Hamming code is defined is $GF(q = 2^{(m-1)} = 16)$. In $GF(16)$, every nonzero element has order that divides 15. An element may have order 1, 3, 5, or 15. An element with order 15 is primitive. We can construct $GF(16)$ with the polynomial $p(z) = z^4 + z + 1$, and the element α as primitive. The elements of $GF(16)$are shown on Table II.

The parity check matrix for this Hamming code on $GF(16)$ is

$$H_{GF(16)} = \left[\begin{array}{cccccc} 0 & 1 & 1 & 1 & \cdots & 1 \\ 1 & 0 & \alpha & \alpha^2 & \cdots & \alpha^{15} \end{array} \right]. \tag{7}$$

TABLE II

REPRESENTATION OF ELEMENTS OF $GF(16)$. V_{α^i} INDICATES THE BINARY VECTOR REPRESENTATION OF THE ELEMENT α^i

Element		Polynomial in α		V_{α^i}
α	=	α	=	0 0 1 0
α^2	=	α^2	=	0 1 0 0
α^3	=	α^3	=	1 0 0 0
α^4	=	$\alpha + 1$	=	0 0 1 1
α^5	=	$\alpha^2 \quad \alpha$	=	0 1 1 0
α^6	=	$\alpha^3 + \alpha^2$	=	1 1 0 0
α^7	=	$\alpha^3 + \alpha + 1$	=	1 0 1 1
α^8	=	$\alpha^2 + 1$	=	0 1 0 1
α^9	=	$\alpha^3 + \alpha$	=	1 0 1 0
α^{10}	=	$\alpha^2 + \alpha + 1$	=	0 1 1 1
α^{11}	=	$\alpha^3 + \alpha^2 + \alpha$	=	1 1 1 0
α^{12}	=	$\alpha^3 + \alpha^2 + \alpha + 1$	=	1 1 1 1
α^{13}	=	$\alpha^3 + \alpha^2 + 1$	=	1 1 0 1
α^{14}	=	$\alpha^3 + 1$	=	1 0 0 1
α^{15}	=	1	=	0 0 0 1

An equivalent multiplication operation to the ones appearing on Table II needs to be defined for $GF(2)$. To accomplish this a binary matrix A_α of dimension $m-1$ will be used. A_α is such that multiplying by a binary vector V_{α^i} on the right by A_α, is equivalent to multiplying the corresponding element α^i in $GF(2^{m-1})$ by α. This matrix needs to satisfy the following:

$$A_{\alpha^i} = (A_\alpha)^i \leftrightarrow \alpha^i = (\alpha)^i, \tag{8}$$

$$V_{\alpha^i} \cdot A_\alpha = V_{\alpha^{i+1}} \leftrightarrow \alpha^i \cdot \alpha = \alpha^{i+1}. \tag{9}$$

Using the vector representation shown in Table II, the matrix A_α can be derived and is equal to the first four vectors in Table II in reverse order :

$$A_\alpha = \begin{bmatrix} 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{10}$$

By means of this primitive sub-matrix we can now generate the parity check matrix of a rate-efficient array code by computing the different powers of A_α. Since α is a primitive element in the Galois field, all the $2^{m-1}-1$ powers of A_α are different from each other. (It has to have order 15.) The binary equivalent of (7) is

$$H_{GF(2)} = \begin{bmatrix} 0 & I & I & I & \cdots & I \\ I & 0 & A_\alpha & A_{\alpha^2} & \cdots & A_{\alpha^{15}} \end{bmatrix}, \tag{11}$$

where all the square sub-matrices above are of size $(m-1) = 4$. In $GF(16)$ some of the matrix-elements are :

$$A_{\alpha^2} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \quad A_{\alpha^3} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \tag{12}$$

TABLE III

DIMENSIONS OF HAMMING CODES OVER $GF(q)$ AS A FUNCTION OF m, FOR $m_h = n-k = 2$

m	3	5	7	11
$q = 2^{m-1}$	4	16	64	1024
(k,n)	(3,5)	(15,17)	(63,65)	(1023,1025)

The procedure to obtain rate-efficient array codes by describing them as Hamming codes has now been completely defined. This shows that rate-efficient array codes are Hamming codes defined over $GF(q = 2^{(m-1)})$. In general A_α is equal to a $\left(2^{m-1}-1\right)^{\text{th}}$ root of the identity matrix. The general form of (11) has square sub-matrices of size $m-1$ and can be expressed as:

$$H_{GF(q=2^{(m-1)})} = \begin{bmatrix} 0 & I & I & I & \cdots & I \\ I & 0 & A_\alpha & A_{\alpha^2} & \cdots & A_{\alpha^{(q-1)}} \end{bmatrix}.$$

The requirement of m being prime is only necessary for array codes to have a cyclic-shift decoder. Out of all possible Hamming codes defined over a Galois field, the ones with $m_h = n-k = 2$ are considered since this is the relationship between input and output symbols in the case of array codes with two parity columns. The dimensions of Hamming codes as a function of m is shown on Table III where it can be seen that the values of k and n for Hamming codes match the $(M_{SB}-2, M_{SB})$ values of array codes.

The rate of a Hamming Code code is

$$R_{\text{Hamming}} = \frac{2^{m-1}-1}{2^{m-1}+1}. \tag{13}$$

The increase in rate compared to original array codes is shown in Fig. 3. As was mentioned in Section I, there is a great increase in code rate for short code lengths but the gain is reduced as the code length grows. Fig. 3 shows that for a code length $n = 42$ ($m = 7$), the rate increase is 35.6% but that gain falls to 9.52% when the code length is increased to $n = 506$ ($m = 23$).

IV. ENCODING AND DECODING ALGORITHMS

The systematic parity-check matrix of Hamming codes allows a very simple encoder implementation. If the first and second sub-matrices of $H_{GF(2)}$ in (11) are interchanged, a systematic parity-check matrix is obtained

$$H_{SYS} = \begin{bmatrix} I & 0 & I & I & \cdots & I \\ 0 & I & A_\alpha & A_{\alpha^2} & \cdots & A_{\alpha^{15}} \end{bmatrix}. \tag{14}$$

For the binary parity-check matrix $H_{SYS} = \begin{bmatrix} I_{n-k} & P^T \end{bmatrix}$ the corresponding generator matrix is $G_{SYS} = \begin{bmatrix} P & I_k \end{bmatrix}$ that allows linear-time encoding.

The decoding algorithm for array codes was presented in Section II-C. The burst appeared on the first $m-1$ bits of the syndrome. The position of the burst was derived by using the last $m-1$ bits of the syndrome by means of shift and compare

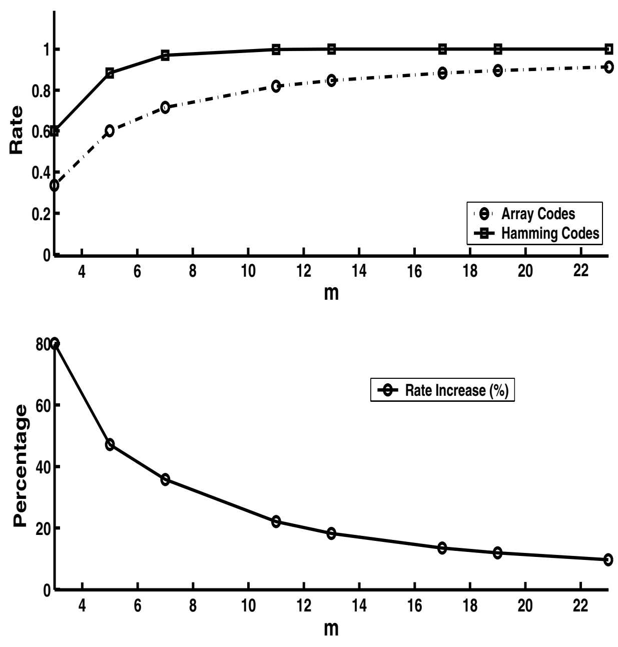

Fig. 3. Comparison of original array codes and rate efficient array codes

operations. Since Hamming codes modify the structure of the bottom half of H, a new decoding algorithm needs to be defined.

Algorithm 2 Hamming Code Decoding Algorithm

```
1:  $s = r \cdot H^T$
2:  BurstType=$s\,(1 : m - 1)$
3:  if Burst-Type $\neq \vec{0}$ then
4:      for i=1 to $M_{SB}$ do
5:          if (Burst-Type $\cdot H_{BOT}(i)) = s(m : end)$ then
6:              Position=i
7:              break
8:          end if
9:      end for
10: else
11:     Burst-Type =$s(m : end)$
12: end if
```

Algorithm 2 is a universal decoding algorithm that can be applied to any type of array code. It is actually an extension of Algorithm 1 for the general case when the bottom half of the parity-check matrix is unconstrained. The simple bit shifts of the original array codes are replaced by XOR and adding operations required for vector multiplication. The close relation between both algorithms can be seen by observing the similarities of their pseudo-codes.

V. Conclusions

The most rate-efficient codes that meet the original burst-error-correction requirement of array codes with two parity columns are simply Hamming codes. These codes can correct all phased error bursts of length $L \leq (m-1)$. Unlike the original array codes, m need not be prime. An encoding algorithm was presented along with a decoding procedure which is a generalization of the decoding algorithm for traditional array codes. Both encoding and decoding procedures consist of simple additions and XOR operations. Still this is more complex than what the original array codes require. For very short code lengths the gain in rate obtained is significant, but as the code length increases and the code rate approaches one, the rate increase introduced by expressing array codes as Hamming codes is greatly reduced. For a length $n = 42$ code, the rate increase is 35.6% but that gain falls to 9.52% when the code length is increased to $n = 506$.

References

[1] M. Blaum, P. Farrell, and H. van Tilborg., *Handbook of Coding Theory*. V.S.Press and W.C.Huffman. Elsevier Science B.V., 1998, ch. 22.
[2] R. Blahut, *Theory and Practice of Error Control Codes*. Addison-Wesley., 1983.
[3] S. Wicker, *Error Control Systems for Digital Communication and Storage*. Prentice Hall., 1995.
[4] J. Fan., "Array codes as low-density parity-check codes," *Proceedings of Second Int. Symposium on Turbo Codes (ISTC 2000)*, pp. 423–426, Brest, France, Sept 4–7, 2000.
[5] K. Yang and T. Helleseth., "On the minimum distance of array codes as LDPC codes," *IEEE Trans. on Information Theory*, vol. 49, no. 12, pp. 3268–3271, Dec. 2003.
[6] R. G. Gallager, "Low-density parity-check codes," *IRE Trans. Inform. Theory*, vol. IT-8, pp. 21–28, Jan. 1962.
[7] M. Blaum and R. Roth., "New array codes for multiple phased burst correction," *IEEE Trans. on Information Theory*, vol. 39, no. 1, pp. 66–67, Jan. 1993.
[8] D. Raphaeli., "The burst error correcting capabilities of a simple array code," *IEEE Trans. on Information Theory*, vol. 51, no. 2, pp. 722–728, Feb. 2005.
[9] D. A. Patterson, P. Chen, G. Gibson, and R. H. Katz., "Introduction to redundant arrays of inexpensive disks (RAID)," *Proceedings of IEEE COMPCON Spring '89*, pp. 112–117, March 1989.
[10] M. Blaum, J. Brady, J. Bruck, and J. Menon., "Evenodd: An efficient scheme for tolerating double disk failures in raid architectures," *IEEE Trans. on Computers*, vol. 42, no. 2, pp. 192–202, Feb 1995.
[11] M. Blaum and A. Vardy., "Array combinatorial decoding with multiple error and erasure detection and location using cyclic equivalence testing," *US Patent 5,644,695*, July 1997.
[12] R. Hamming, "Error detecting and error correcting codes," *Bell Syst. Tech. J.*, vol. 42, pp. 79–74, April 1950.