Array Codes with Error Detection For Efficient Storage Systems

A Kh. Al Jabri (Prof), Al Shahrani A.
Elect. Eng. Dept., King Saud University,
P.O.Box 800, Riyadh 11421, Saudi Arabia
e-mail: aljabri@ksu.edu.sa

Abstract- Increasing the efficiency of data storage systems for a given degree of reliability is of extreme practical importance. In this paper we propose a technique for increasing such efficiency. The idea is based on generating side information about the erroneous symbol positions and exploit the capability of linear codes for correcting erasures. A simple hash function is proposed for checking the integrity of the data and hence generating such information. The overhead for such improvement is negligible. This will result in almost 50 % saving in the redundancy requirements.

1. Introduction

Increasing the reliability of transmitted or stored data is of extreme practical importance. For storage systems, the stored data are subjected to possible disk failure or errors resulting, therefore, in erroneous data upon retrieval. Such a situation is modeled as a communication channel where errors are introduced in time (now to then). Error control codes are used to increase the reliability of such channels.

Different codes for this channel are proposed in the literature. A common class of codes for this channel is the array codes. These codes have wide applications in data storage systems such as magnetic tapes and RAID [1, 2, 3]. Array codes are two-dimensional codes but can be MDS in one of their format. For these codes the encoding and decoding operations are performed on the binary filed, $GF(2)$. Such structure allows for reduced complexity and high encoding and decoding speeds.

In [2], Blaum, *et. al.* proposed MDS array codes for efficient and reliable storage and retrieval data systems. The code is an (n, k) MDS array codes that can correct up to $(n-k)/2$ errors for the selected values of k and n. In such systems, k disks or columns of the array are reserved for the data while the remaining $(n-k)$ are for parities. The parity columns are simple **XOR** function of the information column bits. It is important to note that the size of each column is typically large; few Gbytes [3].

The code can correct both errors and erasures. Errors, however, need more redundancy (parities) than erasures; this is natural since erasures provide more information than errors since the location of the erasures are known. For array codes with large columns size, it may not be obvious how to obtain side information about the location of the erroneous columns. One may, however, divide each column into sub blocks and add simple parity to check every block. This will add a large overhead and reduce the storage efficiency. A suitable technique is to use one of the data integrity mechanism such as hashing functions typically used in cryptographic applications. In our case, however, simple hash functions may be sufficient. Before discussing the approach, we first review some basics of burst correcting array codes.

2. Preliminaries Of Array Codes

Of interest is the class of array codes for multiple phased burst errors introduced by Blaum, et al.. Let the code be denoted by $B_r(p)$. It is shown in [2] that if p is a prime, then $B_r(p)$ is an $(p, p-r)$ MDS code that can correct up to r erasures. In this code, the first $p-r$ columns are for information while the remaining r columns are for parities. Therefore, we can regard the encoding procedure as a special case of the erasure decoding when the last r columns are erased. $B_r(p)$ is defined as a set of arrays $(a_{i,j})\ \ 0 \leq i, j \leq p-1$ with $a_{p-1,j} = 0$ for all j. satisfying the following constraints:

$$\sum_{l=0}^{p-1} a_{<j-tl>_{p,l}} = 0,$$

$$0 \leq j \leq p-1, 0 \leq t \leq r-1.$$

Algebraically, this code can be viewed as conventional Reed-Solomon code, except that it is defined over the ring of polynomials modulo $M_p(x) = x^{p-1} + x^{p-2} + \ldots + x + 1$. $B_r(p)$ is characterized by the following check matrix:

$$H = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{p-1} \\ 1 & \alpha^2 & \alpha^4 & \cdots & \alpha^{2(p-1)} \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 1 & \alpha^{r-1} & \alpha^{2(r-1)} & \cdots & \alpha^{(p-1)(r-1)} \end{bmatrix}$$

In [2], two algorithms are presented for all erasure decoding and single-error multiple-erasure decoding. For multiple error case [2], the $B_r(p)$ is defined over the extended field and the Berlekamp-Massey algorithm is proposed for decoding. Of course, complexity increases due to defining the code over the extended field. Next, we summarize the erasure-decoding algorithm. More details about this algorithm are presented in [2]. Assume $\rho\ (\leq r)$ erasures have occurred in the positions $j_0, j_1, \ldots, j_{\rho-1}$. Let $\mathbf{b} = [b_0, b_1, \ldots, b_{p-1}]$ be the received (retrieved) vector that equals the transmitted code word in at all coordinates (columns) except the erased ones, in which it is zero. Also, let $\alpha_i = \alpha^{j_i}$ and $e_i = -a_{j_i}$ denote locator and erasure value, respectively, for any $j_i,\ 0 < i \leq \rho - 1$. First, we compute the syndrome values as [2]:

$$S_l(\mathbf{b}H^T)_l = \sum_{j=0}^{p-1} b_j \alpha^{jl} = \sum_{i=0}^{\rho-1} e_i \alpha_i^l.$$

Having found the syndrome, the decoder has to solve the linear system

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ \alpha_0 & \alpha_1 & \cdots & \alpha_{\rho-1} \\ \alpha_0^2 & \alpha_1^2 & \cdots & \alpha_{\rho-1}^2 \\ \vdots & \vdots & \cdots & \vdots \\ \alpha_0^{\rho-1} & \alpha_1^{\rho-1} & \cdots & \alpha_{\rho-1}^{\rho-1} \end{bmatrix} \begin{bmatrix} e_0 \\ e_1 \\ \vdots \\ e_{\rho-1} \end{bmatrix}$$

$$= \begin{bmatrix} s_0 \\ s_1 \\ \vdots \\ s_{\rho-1} \end{bmatrix}$$

The solution of the above equation is given by

$$\prod_{s=0, s\neq i}^{\rho-1} (\alpha_i - \alpha_s) e_i = \sum_{l=0}^{\rho-1} \gamma_{i,\rho-1-l} S_l,$$

$$0 \leq i \leq \rho - 1,$$

where

$$\gamma_i(z) = \prod_{s=0, s\neq i}^{\rho-1} (1 - \alpha_s z) = \sum_{l=0}^{\rho-1} \gamma_{i,l} z^l,$$

$$0 \leq i \leq \rho - 1.$$

Here all the operations are performed mod $M_p(x)$. An efficient algorithm is presented in [2] to extract the value of e_i from the above equations.

3. The Proposed Technique

Here, we propose the usage of a simple XOR hash function typically used in some applications to test the integrity of the stored data [5]. In particular, each column information $\mathbf{c}_j$ is hashed by a function $h(\mathbf{c}_j)\ \ j := 0, 2, \ldots, p-1$ and the result is padded to the column or stored in the storage controller. The size of such hash

is usually small (128 bits, 256 bits, or 512 bits). Therefore, it is reasonable to assume that such data will always be retrieved correctly.

In the retrieval process, the retrieved j^{th} column data, $\mathbf{y}_j$, will be

$$\mathbf{y}_j = \mathbf{c}_j \oplus \mathbf{e}_j,$$

where $\mathbf{e}_j$ is the j^{th} column error. Now suppose a simple XOR hash function is used. In this case, the column data is first divided into blocks of size m bits and the blocks are then XORed. If m does not divide the size of the column, then one can pad the remaining with zeros.

Upon retrieval, the system check if $h(\mathbf{y}_j)$ is equal to $h(\mathbf{c}_j)$. If equal, then the j^{th} column is accepted as error free data otherwise the column is erased. From this it follows that

$$h(\mathbf{y}_j) = h(\mathbf{c}_j \oplus \mathbf{e}_j) = h(\mathbf{c}_j) \oplus h(\mathbf{e}_j).$$

An undetected error will occur if

$$h(\mathbf{e}_j) = 0,$$

while the data contain errors. That is a column will be accepted although it contains an error. Let $h(\mathbf{e}_j) = (h_1^j, h_2^j, \ldots, h_m^j)$. Since m is relatively large, the i^{th}, $i = 1, 2, \ldots, m$ bit of $h(\mathbf{e}_j)$ will be the XOR sum of a large number of random bits. Even if the probability, p, of a bit being in error is very small, the sequence $h_1^j, h_2^j, \ldots, h_m^j$, for large m, can be approximated as independent identically distributed Bernoulli random variables with $Pr(h_i^j = 0) = 0.5$, $i := 1, 2, \ldots, m$. This follows from the central limit theorem.

The next theorem shows the probability of such an event.

Theorem 1:

The probability that $h(\mathbf{e}_j) = 0$, $j = 1, 2, \ldots, m$ is 2^{-m}.

■

If m is chosen sufficiently large, this probability will be negligible. It is important to note that the technique will be effective against random errors. For other kind of errors such as a disk failure, we assume that the storage system will be able to detect such failure and pass the erasure information to the decoder.

4. Discussion

Typical storage systems have more than one disk with a capacity of few Gbytes per disk. If, for example, 17 disks are used with 3 Gbytes/disk of which 2 disks are redundant, then the proposed technique suggests that only 16 disks are needed to obtain the same amount of reliability. The hash overhead (256 bits or 32 bytes), in this case, will be $16 \times 32 = 512$ bytes. This is negligible compared to 48 Gbytes of the storage capacity.

In general, for a data storage system with p disks of which r disks are redundant, the proposed technique suggests reducing the storage size to $p - \frac{r}{2}$ (r is assumed to be even) and can still have the same level of reliability.

5. Conclusions

This paper has proposed a new technique for increasing the capacity of conventional data storage systems while keeping the same level of data reliability. The technique is based on using hash function for checking the data integrity within these disks. The hash function will provide error detection and hence erasure information that can be passed to the decoder for error correction. Since the amount of redundancy needed to correct erasures is almost 50% less than that required for errors, this will result in increasing the efficiency of the storage system while having the same amount of reliability. We expect that this technique will have a significant impact on the design of such systems.

References

[1] M. Blaum, J. Bruck, and A. Vardy, "MDS

Array Codes with Independent Parity Symbols," IEEE Trans. Inform. Theory, Vol. 32, pp. 529-542, 1996.

[2] M. Blaum, and R. Roth, "New array codes for multiple phased burst correction," IEEE Trans. Inform. Theory, Vol. 39, pp. 66-77, 1993.

[3] M. Blaum, J. Brady, J. Bruck, and J. Menon, "EVENODD: An Efficient Scheme for Tolerating Double Disk Failures in RAID Architectures," IEEE Trans, Comput., Vol. 44, pp. 192-202, 1995.

[4] F.J. McWilliams and N.J. Sloane,"The Theory of Error Correcting Codes," North Publishing Co. 3rd ed.,North Mathematical Library , Vol. 16, Netherlands 1983.

[5] A. Menezes, P. van Oorschot, S. Vanstone, "Handbook of Applied Cryptography," CRC Press, 1997.