Detection of Storage Errors in Mass Memories Using Low-Cost Arithmetic Error Codes

BEHROOZ PARHAMI, MEMBER, IEEE, AND ALGIRDAS AVIŽIENIS, FELLOW, IEEE

Abstract—**Arithmetic error codes constitute a class of error codes that are preserved during most arithmetic operations. Effectiveness studies for arithmetic error codes have shown their value for concurrent detection of faults in arithmetic processors, data transmission subsystems, and main storage units in fault-tolerant computers. In this paper, it is shown that the same class of codes is also quite effective for detecting storage errors in both shift-register and magnetic-recording mass memories. Some of the results are more general and deal with properties of arithmetic error codes in detecting unidirectional failures. For example, it is shown that a low-cost arithmetic error code with check modulus $A = 2^N - 1$ can detect any unidirectional failure which affects fewer than N bits. The use of arithmetic error codes for checking of mass memories is further justified since it eliminates the need for hard-core or self-checking code translators and reduces the number of different types of code checkers required.**

Index Terms—**Arithmetic error codes, fault-tolerant memories, low-cost arithmetic codes, magnetic-recording memories, mass memories, product codes, residue codes, shift-register memories, storage errors, two-dimensional burst errors, unidirectional failures.**

I. Background and Assumptions

ARITHMETIC error codes are of interest in designing fault-tolerant computer systems since they permit arithmetic operations to be performed on the coded operands by simple algorithms [1]. These algorithms are either identical to those used for noncoded operands or are slightly modified versions of them. Even though arithmetic error codes were originally developed to detect failures within arithmetic processors, they are also capable of detecting transmission and storage errors.

Arithmetic error codes that have been considered use a check modulus A and can be divided into *separate* and *nonseparate* classes [2]. *Residue codes* [3] are separate arithmetic error codes which attach to an arithmetic value X a check value X' computed as the modulo-A residue of X;

$$X' = A \mid X.$$

Inverse residue codes [4] attach to an arithmetic value X a check value

$$X'' = A - A \mid X$$

and have significant advantages in the detection of repeated-use faults. In both residue and inverse residue codes, the check symbol (X' or X'') is separate from the data X; hence, the name "separate codes." "*An*" (*product*) *codes* [5] are nonseparate arithmetic error codes which use the value

$$A \times X$$

to represent the arithmetic value X. The data and check parts cannot be readily separated; hence, the name "nonseparate codes." If the check modulus is of the form

$$A = 2^N - 1,$$

the checking algorithms for error-coded operands become very simple and the resulting codes are referred to as *low-cost arithmetic error codes* [6].

Cost and effectiveness studies for arithmetic error codes [7] have shown their value for concurrent detection of faults in arithmetic processor, data transmission subsystem, and main storage units in a fault-tolerant computer [8]. In this paper, some results are presented which show the effectiveness of arithmetic error codes for detecting storage errors in mass memories. The use of a single arithmetic error code throughout a fault-tolerant computer system is attractive since it eliminates the need for hardcore or self-checking code translators and reduces the number of different types of code checkers required.

In what follows we will assume the organization of Fig. 1. Data are transferred byte-serially between the two stores in blocks of W words, each consisting of B N-bit bytes, as depicted in Fig. 2. Each word is encoded in a product or inverse residue code with check modulus

$$A = 2^N - 1.$$

We do not consider conventional residue encoding since the inverse residue encoding is known to be more effective when byte-serial operations are used [7]. In either case, the checking algorithm performed by the bus checker consists of the modulo-A addition of B bytes in each word and testing the results for zero. The information in a block is assumed to be correct only if each of its W words produces a zero check sum. Otherwise, the bus checker indicates an error.

The mass memory failures can be divided in two categories according to their effects.

1) Some failures result in an error in the selection of the

Manuscript received April 19, 1974; revised July 1, 1977. This work was supported by the National Science Foundation under Grant MCS72-03633 A04.

B. Parhami was with the Department of Computer Science, University of California, Los Angeles, CA 90024. He is now with the Department of Mathematics and Computer Science, Arya-Mehr University of Technology, Tehran, Iran.

A. Avižienis is with the Department of Computer Science, University of California, Los Angeles, CA 90024.

0018-9340/78/0400-0302$00.75

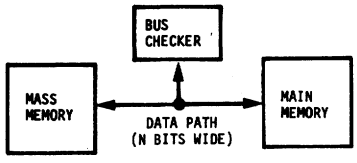


Fig. 1. The system model.

desired block of words. For example, a failure in the decoding circuitry may result in the selection of a different block. A failure in the timing mechanism of a circulating mass memory may result in the selection of W words, some of which do not belong to the desired block. Errors caused by these failures are referred to as *selection errors* and are clearly undetectable by the coding scheme under consideration. We will, therefore, assume that selection errors are avoided through internal redundancy in the selection circuits and/or stored-address redundancy in data.

2) The second class of failures are those which cause incorrect information to be written into or read from the selected block. Errors caused by these failures are referred to as *storage errors*. Storage errors are manifested by the addition of some error magnitude to each byte in a word. The error is undetectable if the B error magnitudes in each word of a block sum to a multiple of the check modulus A.

We will consider two types of mass storage devices; shift-register memories and magnetic-recording memories. The effectiveness of arithmetic error codes for checking these two classes of mass memories will be evaluated by determining what fraction or types of failures result in undetectable error magnitudes. We will restrict our attention to permanent failures of the type stuck-at-1 (s-a-1) or stuck-at-0 (s-a-0). We will assume that all of the 2^N possible byte patterns are equally likely to be stored in any given byte of storage.

II. Checking of Shift-Register Mass Memories

Let us assume that the $W \times B \times N$ bits of each block are stored in N shift registers each containing one bit from each of the $W \times B$ bytes, as shown in Fig. 3. We will find the probability that a permanent failure of one or more shift registers results in an undetectable error. As a result of such a failure, the output of a subset F of the set of N shift registers will assume constant values (0 or 1). Transient failure of a subset F of shift registers is equivalent to having a two-dimensional burst error on a magnetic-recording mass memory, which will be discussed in Section III.

Denoting the probability of failure of a subset F of the shift registers by prob [failure pattern F], we can write

$$\text{prob [undetected error in one block]} = \sum_F \text{prob [failure pattern } F] \times (\text{prob [undetected error in one word} \mid \text{failure pattern } F])^W, \quad (1)$$

where prob $[E_1 \mid E_2]$ is the conditional probability of event E_1 given that event E_2 has occurred. The value of prob [failure pattern F] is implementation-dependent and is assumed to be known for any subset F in (1).

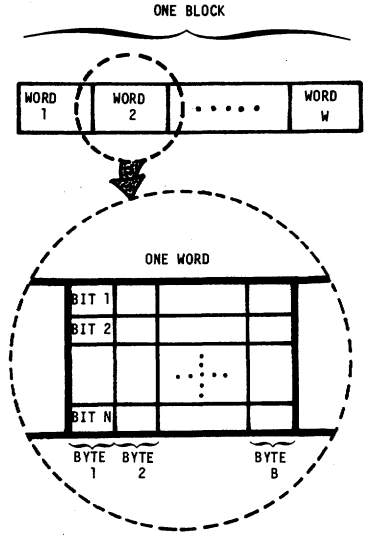


Fig. 2. Organization of data.

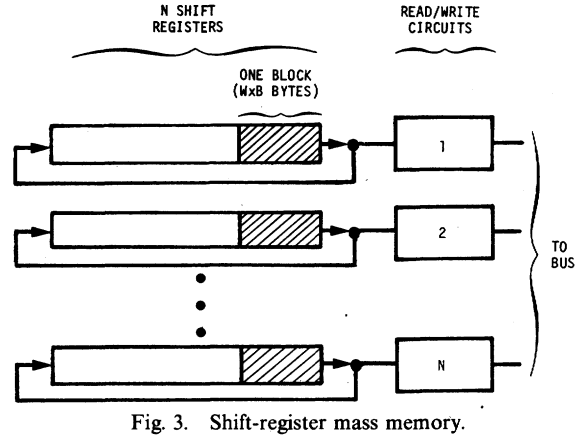


Fig. 3. Shift-register mass memory.

If the failure probability for each subset F is not known, one can use the following inequality to find an upper bound for the probability of an undetected error:

$$\text{prob [undetected error in one block]} \leq \left(\sum_F \text{prob [failure pattern } F]\right) \times (\max_F \text{prob [undetected error in one word} \mid \text{failure pattern } F])^W. \quad (2)$$

On the right-hand side of (2), the first factor is the probability of a failure of any type, and the second factor is an upper bound (over all possible failure patterns) on the probability that errors in all W words of a block go undetected.

To use (1) or (2) for finding the probability of an undetected block error, one needs to know for each failure pattern F the conditional probability that it results in an undetectable word error given that it has occurred. In many cases, this conditional probability can be determined by an enumeration of all possible error magnitudes. The following three steps are needed for this purpose.

Step 1: First find the set of all possible error magnitudes which may be added to each byte.

Step 2: From this set, find all combinations of B error magnitudes that sum to a multiple of the check modulus A.

Step 3: For each such combination, find the probability of its occurrence and add all these probabilities.

Example 1: If we have $N = 4$, $B = 5$, and F consisting of the lowest order and the second highest order shift registers stuck-at-1 (error weights 1 and 4), then the set of all possible error magnitudes added to each byte is

$$\{0, 1, 4, 5\}.$$

When arranged in decreasing order of values, there are only three combinations of 5 error magnitudes from the above set which sum to a multiple of

$$A = 2^N - 1 = 15.$$

These combinations are (5, 5, 5, 0, 0), (5, 5, 4, 1, 0), and (5, 4, 4, 1, 1). The probability of an undetected error is found by summing the probabilities of occurrence for these three combinations:

$$\text{prob [undetected error in one word} \mid \text{failure pattern } F] = \left(\frac{1}{4}\right)^5 \left[\binom{5}{3} + \binom{5}{2} 3! + 5\binom{4}{2}\right] = 0.098.$$

That is, the given failure pattern is not detected in approximately 9.8 percent of the cases.

In some cases, however, such an exact calculation becomes impractical because of the large number of possible combinations. In such cases, the following theorem can be used to find an upper bound for the conditional probability of an undetected word error given that the failure pattern F has occurred.

Theorem 1: Let $S = \{A_1, A_2, \cdots, A_n\}$ be a set of integers, $|A_i| \leq A$, $i = 1, 2, \cdots, n$. Then if B integers are chosen at random (with possible repetition) from S, the probability p_B that their sum is divisible by A satisfies the inequality

$$p_B \leq \frac{2}{n-k+2} + \frac{n-k}{n-k+2}\left(\frac{k-2}{n}\right)^B, \tag{3}$$

where k is the number of integers in S that are divisible by A ($k = 0$, 1, 2, or 3).

Proof: Let p_i denote the probability that the sum of i integers chosen at random from S is divisible by A. We can write

$$p_i \leq p_{i-1}\frac{k}{n} + (1 - p_{i-1})\frac{2}{n}.$$

This recursive inequality is based on the fact that if the sum of i integers is divisible by A, then we must have one of the following two cases: 1) the first $i - 1$ integers selected sum to a multiple of A (probability p_{i-1}) in which case the last one must also be a multiple of A (probability k/n); and 2) the first $i - 1$ integers selected do not sum to a multiple of A (probability $1 - p_{i-1}$) in which case there are at most two choices for the last integer to make the sum a multiple of A (probability $2/n$). Using this inequality and the fact that $p_1 = k/n$, we find the desired result. Q.E.D.

Theorem 1 is a generalized version of a previously published result [9].

Example 2: Assuming $N = 4$, $B = 5$, and F consisting of the two low-order shift registers s-a-1 and the highest order shift register s-a-0, we have the following set of error magnitudes:

$$\{-8, -7, -6, -5, 0, 1, 2, 3\}.$$

We have $k = 1$ and (3) yields

$$p_B < 2/(n+1) = 2/9 = 0.222.$$

That is at most 22.2 percent of the errors will go undetected. Direct computation shows that this bound is highly pessimistic.

We now want to consider a complete example to illustrate the concepts discussed so far. To simplify our subsequent discussion, we introduce the concept of *canonical form* for a set of error magnitudes. A set of error magnitudes is said to be in canonical form if it has the following two properties: 1) the largest error magnitude in the set is not less than the absolute value of the smallest error magnitude; and 2) all the error magnitudes in the set are not divisible by a common power of 2. For any given set of error magnitudes, we can derive a corresponding canonical set by dividing all the members of the set by a power of 2 and changing their signs if necessary. Clearly, neither of these two steps has an effect on the probability that the sum of some such error magnitudes is a multiple of $A = 2^N - 1$. It is also convenient to have a way of specifying failure patterns in shift registers. For this reason, we identify an s-a-1 failure by 1, an s-a-0 failure by 0, and a fault-free shift register by X. Then, $X1X1$ and $0X11$ denote the failure patterns assumed in Examples 1 and 2, respectively.

Example 3: Let us consider a shift-register memory with $N = 4$, $B = 5$, and $W = 20$. The probability of an undetected word error for different failure patterns is given in Table I. Two of the table entries were obtained in Examples 1 and 2 (Lines 6 and 11). Other entries have been computed similarly. There are a total of $3^4 = 81$ possible failure patterns which result in 27 different canonical sets of error magnitudes. From Table I, we have

$$\max_F \text{prob [undetected error in one word} \mid \text{failure pattern } F] \leq 0.222.$$

Using (2), we find

$$\text{prob [undetected error in one block]} \leq (0.222)^{20} \sum_F \text{prob [failure pattern } F] < 10^{-13} \sum_F \text{prob [failure pattern } F].$$

That is, given that a failure of the type considered has occurred, the probability is less than 10^{-13} that it is not detected by the bus checker.

Note that for using Theorem 1 in the computations of Example 3, we need not know the exact set S of error magnitudes. All that we need to know is the number of error magnitudes in the set S and whether or not $\pm 15 \in S$. (Note that we always have $0 \in S$.) Also, one must note that any partial information about the failure patterns may result in

TABLE I
PROBABILITIES OF UNDETECTED FAILURES FOR EXAMPLE 3

No.	Failure Patterns and Their Corresponding Canonical Sets of Error Magnitudes	Probability of Undetected Word Error
1	XXXO, XXX1, XXOX, XX1X, XOXX, X1XX, OXXX, 1XXX 0,1	= 0.000
2	XXO1, XX1O, XO1X, X1OX, O1XX, 1OXX -1,0,1,2	= 0.098
3	XOX1, X1XO, OX1X, 1XOX -1,0,3,4	= 0.050
4	OXX1, 1XXO -1,0,7,8	= 0.068
5	XXOO, XX11, XOOX, X11X, OOXX, 11XX 0,1,2,3	= 0.001
6	XOXO, X1X1, OXOX, 1X1X 0,1,4,5	= 0.098
7	OXXO, 1XX1 0,1,8,9	= 0.001
8	O1X1, 1OXO -5,-4,-1,0,3,4,7,8	≤ 0.222
9	O1XO, 1OX1 -4,-3,0,1,4,5,8,9	≤ 0.222
10	XO11, X1OO, O11X, 1OOX -3,-2,-1,0,1,2,3,4	≤ 0.222
11	OX11, 1XOO -3,-2,-1,0,5,6,7,8	≤ 0.222
12	XO1O, X1O1, O1OX, 1O1X -2,-1,0,1,2,3,4,5	≤ 0.222
13	OX1O, 1XO1 -2,-1,0,1,6,7,8,9	≤ 0.222
14	XOO1, X11O, OO1X, 11OX -1,0,1,2,3,4,5,6	≤ 0.222
15	OXO1, 1X1O -1,0,1,2,7,8,9,10	≤ 0.222
16	OOX1, 11XO -1,0,3,4,7,8,11,12	≤ 0.222
17	XOOO, X111, OOOX, 111X 0,1,2,3,4,5,6,7	≤ 0.222
18	OXOO, 1X11 0,1,2,3,8,9,10,11	≤ 0.222
19	OOXO, 11X1 0,1,4,5,8,9,12,13	≤ 0.222
20	O111, 1OOO -7,-6,-5,-4,-3,-2,-1,0,1,2,3,4,5,6,7,8	< 0.118
21	O11O, 1OO1 -6,-5,-4,-3,-2,-1,0,1,2,3,4,5,6,7,8,9	< 0.118
22	O1O1, 1O1O -5,-4,-3,-2,-1,0,1,2,3,4,5,6,7,8,9,10	< 0.118
23	O1OO, 1O11 -4,-3,-2,-1,0,1,2,3,4,5,6,7,8,9,10,11	< 0.118
24	OO11, 11OO -3,-2,-1,0,1,2,3,4,5,6,7,8,9,10,11,12	< 0.118
25	OO1O, 11O1 -2,-1,0,1,2,3,4,5,6,7,8,9,10,11,12,13	< 0.118
26	OOO1, 1111 -1,0,1,2,3,4,5,6,7,8,9,10,11,12,13,14	< 0.118
27	OOOO, 1111 0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15	≤ 0.125

sharper bounds for the probability values as illustrated by the following example.

Example 4: If in Example 3 we know that about 90 percent of the failures involve only a single shift register or two adjacent shift registers stuck at the same value, we can write

$$\begin{aligned}&\text{prob [undetected error in one block]}\\ &\le [0.9 \times (0.001)^{20} + 0.1 \times (0.222)^{20}]\\ &\quad \times \sum_F \text{prob [failure pattern } F]\\ &< 10^{-14} \times \sum_F \text{prob [failure pattern } F].\end{aligned}$$

It is seen that the bound for the probability of not detecting a failure is decreased by an order of magnitude.

In general, if we partition the set of all possible failure patterns into disjoint subsets $F_1, F_2, \cdots, F_m$, then we have

$$\begin{aligned}&\text{prob [undetected error in one block]}\\ &\le \sum_{k=1}^{m} \left\{ \left(\sum_{F \in F_k} \text{prob [failure pattern } F] \right) \right.\\ &\quad \times \left. \left(\max_{F \in F_k} \text{prob [undetected error in one word} \mid \text{failure pattern } F] \right)^W \right\}.\end{aligned}$$

The inverse problem of finding a lower bound for W to achieve a desired error detection coverage is also of some interest. As illustrated in the following example, this can be found directly from (2).

Example 5: Suppose in a system with $N = 4$ and $B = 5$ we want the probability of not detecting an error given that a failure has occurred to be less than 10^{-9}. Using (2) and the results of Example 3, we have

$$10^{-9} > (0.222)^W$$

or

$$W > (-9) \div \log 0.222 = 13.77.$$

Therefore, to achieve the desired error detection coverage, each block must consist of at least 14 words.

From the preceding discussion, we conclude that arithmetic error codes are extremely effective for detecting permanent shift-register failures in shift-register mass memories. Clearly, the results were based on some simplifying assumptions about the failures. However, we note that even if the probability of detecting a word error through external checking is as low as 0.5, for Example 3 the probability of an undetected error in a block of 20 words is still $(0.5)^{20}$ which is less than 10^{-6} and the error detection coverage of Example 5 can be achieved for $W \ge 30$.

III. CHECKING OF MAGNETIC-RECORDING MASS MEMORIES

We distinguish two classes of failures in mass memories which use magnetic recording.

1) Failures of the type s-a-1 and s-a-0 of the read/write circuitry whose effects are identical to shift-register failures in shift-register mass memories discussed in Section II.

2) Bursts on the magnetic recording surface due to dust particles, minute scratches, and defects in the coating, which will be discussed subsequently.

We will assume that a burst error of this type results in unidirectional multiple failures; i.e., multiple failures that are all of the same type s-a-1 or s-a-0, but not both. Let us assume that on the recording surface, the N bits of each byte are stored in parallel while the $W \times B$ bytes of each block are stored serially. As shown in Fig. 4, a two-dimensional burst may affect a number of adjacent bit positions each of which contributes an error magnitude to the check sum. For example, the burst pattern of Fig. 4 can generate any even error magnitude between 0 and 80 (inclusive). If the check constant is $A = 2^N - 1 = 63$, all such errors are detectable.

The above constitutes a generalization of the concept of one-dimensional bursts. Just as a one-dimensional burst is characterized by its length (in bits), a two-dimensional burst may be characterized by its area (in bits). The effectiveness of an error code for detecting one-dimensional burst errors is measured in terms of the shortest undetectable burst. Similarly, we can measure the effectiveness of arithmetic error codes for detecting two-dimensional burst errors in terms of the smallest undetectable burst area.

Finding the smallest possible burst area which may result in an undetectable error for an arithmetic error code is an interesting problem. Clearly, an upper bound for this minimum area is N since if all bits in a byte are changed from 0 to 1 by s-a-1 failures, the resulting error magnitude will be $2^N - 1 = A$ which corresponds to an undetectable error. For $N = 2, 3, 4, 5$, and 6, simple enumerations show that the minimum area is in fact equal to N; i.e., any burst with an area smaller than N bits is detectable. We now prove some results that give us lower bounds for the size of the smallest burst area resulting in an undetectable error.

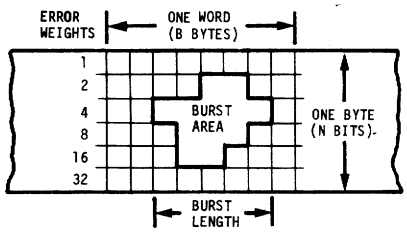


Fig. 4. A two-dimensional burst error pattern.

Theorem 2: For $N \geq j$, at least

$$2^{N-j+1} + j - 2$$

integers from the set

$$T_j = \{1, 2, 4, \cdots, 2^{j-1}\}$$

are needed to form a sum which is a multiple of $2^N - 1$.

Proof: Let $K_{i,j}^N$ be the minimum number of integers from T_j which sum to $i \times (2^N - 1)$. Clearly, in such a minimal collection of $K_{i,j}^N$ integers, no element of T_j other than 2^{j-1} can be repeated since, otherwise, a smaller collection can be obtained by replacing two occurrences of $a < 2^{j-1}$ by $2 \times a \in T_j$. Let $M_{i,j}^N$ be the number of occurrences of 2^{j-1} in the minimal collection of $K_{i,j}^N$ integers. Then, we have by assumption

$$i \times (2^N - 1) = b + M_{i,j}^N \times 2^{j-1}, \tag{5}$$

where

$$0 \leq b < 2^{j-1}.$$

Clearly

$$K_{i,j}^N = M_{i,j}^N + W(b), \tag{6}$$

where $W(b)$ is the number of 1's in the binary representation of b. For $i = 1$, we have

$$\begin{aligned} M_{1,j}^N &= 2^{N-j+1} - 1, \\ b &= 2^{j-1} - 1, \\ K_{1,j}^N &= M_{1,j}^N + W(b) = 2^{N-j+1} + j - 2. \end{aligned} \tag{7}$$

If $i \geq 2$, we can subtract $2^N - 1$ from both sides of (5) to obtain

$$(i-1) \times (2^N - 1) = b + 1 + (M_{i,j}^N - 2^{N-j+1}) \times 2^{j-1}. \tag{8}$$

From (8) and (6), we conclude that

$$\begin{aligned} K_{i-1,j}^N &\leq W(b) + 1 + M_{i,j}^N - 2^{N-j+1} \\ &= K_{i,j}^N - (2^{N-j+1} - 1) \\ &\leq K_{i,j}^N - 1. \end{aligned} \tag{9}$$

Using (9) recursively, we find

$$K_{i,j}^N \geq K_{1,j}^N + i - 1.$$

From (7), we conclude that in any case

$$K_{i,j}^N \geq 2^{N-j+1} + j - 2. \qquad \text{Q.E.D.}$$

Corollary 1: At least N integers from the set

$$T_N = \{1, 2, 4, \cdots, 2^{N-1}\}$$

are needed to form a sum which is a multiple of $2^N - 1$.

Proof: Let $j = N$ in Theorem 2. Q.E.D.

Corollary 1 states that any burst whose area is smaller than N is detectable by an arithmetic error code with check modulus $A = 2^N - 1$. In fact, the result is more general in the sense that the bit positions affected by the failure do not have to be adjacent. *Any* unidirectional failure that affects fewer than N bits is detectable by a low-cost arithmetic error code with check modulus $A = 2^N - 1$.

Note that we have not put any restriction on the shape of the burst and have considered only its area. The fact that on magnetic-recording memories the bit density is much higher than the track density (an order of magnitude for disk memories [10], for example) makes the occurrence of some burst patterns highly improbable. This generally results in improved error detection capability.

Theorem 2 states that if a burst error affects only j adjacent tracks on the magnetic recording memory, the smallest undetectable burst is of size

$$2^{N-j+1} + j - 2 \geq N.$$

If the j affected tracks are those with error weights $1, 2, \cdots$, and $2^j - 1$, this conclusion is immediate. Otherwise, we can divide all error weights by the smallest one to obtain the above set of error weights. Since any power of 2 is relatively prime with respect to $2^N - 1$, this change in error weights does not affect the detectability of any burst error.

Example 6: Let us assume that $N = 8$. Then by Corollary 1, any burst of size 7 or less is detectable. However, if we can assume that a burst affects at most three adjacent tracks, then by Theorem 2, the smallest undetectable burst has an area of

$$2^{N-j+1} + j - 2 = 2^{8-3+1} + 3 - 2 = 65.$$

Hence, any burst of size 64 or less is detectable. This represents an improvement by a factor of about 9 in the error detection capability over the case where we have no restriction on the shape of a burst error.

It is also interesting to find a lower bound for the length of the minimum-length undetectable burst. The definition of burst length should be clear from Fig. 4. Such a result will give us an upper bound on the number of bytes B in each word in terms of N and j if all single bursts affecting j or fewer tracks are to be detectable. It also gives us an upper bound on the longest allowable (in terms of memory cycles) unidirectional transient failure affecting j adjacent shift registers in a shift-register mass memory if it is to be detectable.

Theorem 3: For $N \geq j$, at least

$$\lceil (2^N - 1) \div (2^j - 1) \rceil$$

integers from the set

$$S_j = \{1, 2, 3, 4, \cdots, 2^j - 1\}$$

are needed to form a sum which is a multiple of $2^N - 1$. $\lceil Y \rceil$ denotes the smallest integer which is not less than Y.

Proof: Let $K_{i,j}^N$ be the minimum number of integers

from S_j which sum to $i \times (2^N - 1)$. Clearly, there exists one such minimal collection of $K_{i,j}^N$ integers in which there is only one element not equal to $2^j - 1$. For if the collection contains $u < 2^j - 1$ and $v < 2^j - 1$, we can obtain another collection of size $K_{i,j}^N - 1$ or $K_{i,j}^N$ by replacing u and v by

$$(u + v) \in S_j, \qquad \text{if } u + v < 2^j,$$

$$2^j - 1 \quad \text{and} \quad (u + v - 2^j + 1) \in S_j, \qquad \text{otherwise.}$$

By repeating this procedure, we can always find a minimal collection with the desired property. Such a minimal collection will contain

$$\lfloor i \times (2^N - 1) \div (2^j - 1) \rfloor$$

occurrences of $2^j - 1$ and one other integer if $i \times (2^N - 1)$ is not divisible by $2^j - 1$. Therefore, in any case

$$K_{i,j}^N = \lceil i \times (2^N - 1) \div (2^j - 1) \rceil \geq \lceil (2^N - 1) \div (2^j - 1) \rceil. \qquad \text{Q.E.D.}$$

Example 7: Consider a system with $N = 8$. If it can be assumed that a burst affects at most three adjacent tracks, no single burst error will go undetected for word lengths of up to $B = 36$ bytes, since

$$\lceil (2^N - 1) \div (2^j - 1) \rceil = 37.$$

With four tracks affected, word lengths of up to $B = 16$ bytes are safe. These results also indicate that if a unidirectional transient failure affects three (four) adjacent shift registers in a shift-register mass memory, then it is detectable if its duration is at most 36 (16) memory shift cycles.

Example 8: Let us consider an associative processor which uses a magnetic-recording memory device (disk) as its storage medium [11]. Let us assume that each byte consists of 9 bits (8 bits for storing a symbol and 1 control bit). The check modulus is therefore $A = 2^9 - 1 = 511$. Since in the system under consideration we deal with variable-length records, it is convenient to choose $W = 1$ and regard the entire contents of one disk cylinder as one word. The redundancy introduced is remarkably low (1 bit per track) but results in the detection of all bursts of size 8 or less. With additional restrictions on the number of tracks affected by a burst, larger burst errors become detectable. A lower bound for the smallest undetectable burst error as a function of the number of tracks affected by a burst is given in Table II. The values given in the second and third columns of Table II have been obtained from Theorems 2 and 3, respectively. From Table II, it is seen that very effective error detection is provided with remarkably low redundancy. The design modifications required to incorporate the checking and encoding consist of the provision of temporary storage for check results and the introduction of additional operation steps to perform the checking addition.

From the preceding discussion, we conclude that arithmetic error codes are very effective for detecting two-dimensional burst errors in magnetic-recording mass memories. In particular, when the number of tracks affected is small with respect to N, very large burst errors become detectable. It may be argued that one can force each burst error to affect only a single track of a data record by storing the data on nonadjacent tracks. However, it is more convenient for parallel readout and processing operations to be performed on data on adjacent tracks.

TABLE II
EFFECTIVENESS OF AN ARITHMETIC ERROR CODE FOR EXAMPLE 8

Number of adjacent tracks affected by burst j	Lower bound for minimum area of an undetectable burst error	Lower bound for minimum length of an undetectable burst error
1	511	511
2	256	171
3	129	73
4	66	35
5	35	17
6	20	9
7	13	5
8	10	3
9	9	1

IV. CONCLUSION

To summarize, we have shown that arithmetic error codes can effectively detect a large class of failures in mass memories. Some of the results obtained are more general and deal with properties of arithmetic error codes in detecting unidirectional failures. We have shown that very high error detection coverage can be achieved. These results, combined with known properties of arithmetic error codes and the fact that a reasonable storage efficiency can be achieved if long words are used, justify their application for checking of mass memories. The use of a single code throughout a fault-tolerant computer system has the additional advantage of eliminating the need for hardcore or self-checking code translators and reducing the number of different types of code checkers required.

Further research in this area may proceed in several directions.

1) The restriction of byte-serial transmission is not absolutely necessary and was imposed for convenience. The same analysis techniques can be used for the case where several bytes are transmitted in parallel from shift-register mass memories. For magnetic-recording mass memories, a single burst with parallel byte transmission may be equivalent to two smaller bursts in the case of byte-serial transmission. However, the result of Corollary 1 is still valid since it does not depend on the relative positions of faulty bits.

2) Another area for further investigation is the characterization of burst patterns for magnetic-recording mass memories and failure patterns for shift-register mass memories. Additional information in this area may enable us to derive tighter bounds for the effectiveness of arithmetic error codes in checking of mass memories.

3) We have not addressed the problem of storage efficiency as measured by the amount of redundant information that needs to be stored. For short word lengths, the additional storage requirements due to redundancy is generally higher for arithmetic error codes than for some other ones such as block codes [12]. However, one must note that

this is the only additional expense for the detection of storage errors if the bus checker is already present for other reasons.

4) Another useful extension of this work would be to consider burst patterns consisting of s-a-1 and s-a-0 failures. To deal with such burst errors, more sophisticated analysis techniques are needed which take into account the relative probabilities of s-a-1 and s-a-0 failures.

Acknowledgment

The authors would like to thank Dr. Ying-Wah Ng for proving Corollary 1 which was stated as an assertion in an earlier version of the paper. His proof technique was used by the authors to prove the more general result of Theorem 2. In the time it took to process this paper for publication, some of the results presented here (and in [9], without proof) appeared in a note by Wakerly [13]. Also, some related results were brought to our attention by Mandelbaum [14]. In particular, our Corollary 1 corresponds to his Lemma 2.

References

[1] A. Avižienis, "Arithmetic algorithms for error-coded operands," *IEEE Trans. Comput.*, vol. C-22, pp. 567–572, June 1973.

[2] H. L. Garner, "Error codes for arithmetic operations," *IEEE Trans. Electron. Comput.*, vol. EC-15, pp. 763–770, Oct. 1966.

[3] W. W. Peterson, "On checking an adder," *IBM J. Res. Develop.*, vol. 2, pp. 166–168, Apr. 1958.

[4] A. Avižienis, "Concurrent diagnosis of arithmetic processors," in *Dig. 1st Annu. IEEE Comput. Conf.*, pp. 34–37, Sept. 1967.

[5] D. T. Brown, "Error detecting and correcting binary codes for arithmetic operations," *IRE Trans. Electron. Comput.*, vol. EC-9, pp. 333–337, Sept. 1960.

[6] A. Avižienis, "A set of algorithms for a diagnosable arithmetic unit," Jet Propulsion Lab., Pasadena, CA, Tech. Rep. 32-546, Mar. 1964.

[7] ——, "Arithmetic error codes: Cost and effectiveness studies for application in digital system design," *IEEE Trans. Comput.*, vol. C-20, pp. 1322–1331, Nov. 1971.

[8] A. Avižienis, G. C. Gilley, F. P. Mathur, D. A. Rennels, J. A. Rohr, and D. K. Rubin, "The STAR (self-testing-and-repairing) computer: An investigation of the theory and practice of fault-tolerant computer design," *IEEE Trans. Comput.*, vol. C-20, pp. 1312–1321, Nov. 1971.

[9] B. Parhami and A. Avižienis, "Application of arithmetic error codes for checking of mass memories," in *Dig. Int. Symp. Fault-Tolerant Computing*, pp. 47–51, June 1973.

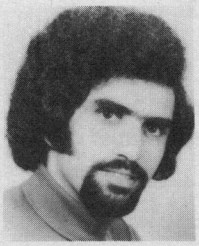
[10] R. E. Matick, "Review of current proposed technologies for mass storage systems," *Proc. IEEE*, vol. 60, pp. 266–289, Mar. 1972.

[11] B. Parhami, "A highly parallel computing system for information retrieval," in *Fall Joint Comput. Conf., AFIPS Conf. Proc.*, vol. 41. Montvale, NJ: AFIPS Press, 1972, pp. 681–690.

[12] P. R. Daher, "Automatic correction of multiple errors originating in a computer memory," *IBM J. Res. Develop.*, vol. 7, pp. 317–324, Oct. 1963.

[13] J. F. Wakerly, "Detection of unidirectional multiple errors using low-cost arithmetic codes," *IEEE Trans. Comput.*, vol. C-24, pp. 210–212, Feb. 1975.

[14] D. Mandelbaum, "Arithmetic error detecting codes for communications links involving computers," *IEEE Trans. Commun. Technology*, vol. COM-13, pp. 165–171, June 1965.



Behrooz Parhami (S'70–M'73) was born in Tehran, Iran, on February 1, 1947. He received the B.S. degree in electrical engineering from University of Tehran, Tehran, in 1968, and the M.S. and Ph.D. degrees in computer science from Oregon State University, Corvallis, and the University of California, Los Angeles, in 1970 and 1973, respectively.

From 1968 to 1969, he was an instructor at the School of Engineering, University of Tehran. He was a Research Assistant in 1970–1971 and a Post-Graduate Research Engineer from 1971 to 1973 in the Department of Computer Science, University of California, Los Angeles. From 1973 to 1974, he served as Acting Assistant Professor at UCLA. Since 1974, he has been an Assistant Professor of Computer Science at Arya-Mehr University of Technology, Tehran, Iran, where he is currently Vice-Chairman of the Department of Mathematics and Computer Science. His research interests are in the area of computer architecture; in particular, parallel processing and fault-tolerant computing. He is the author or co-author of 25 technical papers.

Dr. Parhami is a member of the Association for Computing Machinery, the British Computer Society, and the Iranian Mathematical Society. He is also the Chairman of IEEE Iran Section.

Algirdas Avižienis (S'55–M'56–F'73) was born in Kaunas, Lithuania, in 1932. He received the B.S., M.S., and Ph.D. degrees in electrical engineering from the University of Illinois, Urbana.

While at the University of Illinois he was associated with the Digital Computer Laboratory as a Research Assistant and Fellow from 1954 to 1960, participating in the design of the Illiac II computer. In 1960 he joined the Technical Staff of the Jet Propulsion Laboratory (JPL), California Institute of Technology, Pasadena, where he initiated and directed the JPL Self-Testing And Repairing (STAR) computer research project. In 1962 he became a member of the faculty of the School of Engineering and Applied Science at the University of California, Los Angeles. Presently he is a Professor in the Department of Computer Science at UCLA, where he teaches and conducts research in computer system architecture and fault-tolerant computing and is the Principal Investigator of a five-year research grant on fault-tolerance sponsored by the National Science Foundation. He has also remained associated with the Jet Propulsion Laboratory as an Academic Member of the Technical Staff, and he has served as the Principal Consultant on fault-tolerant computer research projects with Ultrasystems, Inc., Irvine, CA.

Dr. Avižienis has served as the first Chairman of the IEEE Computer Society's Technical Committee on Fault-Tolerant Computing (1969–1973), and as Chairman of the First International Symposium on Fault-Tolerant Computing in 1971.