ERROR-CORRECTING CODES FOR COMPUTER MEMORIES

Thesis by

Mario Blaum

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California

1985

(Submitted December 10, 1984)

ACKNOWLEDGMENT

I am deeply indebted to my advisor, Prof. Bob McEliece, for his guidance and encouragement. The results in chapter III were obtained during a two-week stay at the University of Manchester, England, under a kind invitation of Prof. Paddy Farrell. Other professors that had a great influence in my mathematical training during my stay at Caltech were R. Wilson, J. van Lint, H. van Tilborg, R. Goodman and G. Lorden, just to mention a few.

Special thanks to Jeffrey Pugh for his help in the numerical integration at the end of chapter II, and to Phil Merkey and Li-Fung for their patience while teaching me TEX.

The secretaries in the Departments of Mathematics and Electrical Engineering have always been very helpful and patient with me.

The friends with whom I have shared these years are many. We have had fun together and they have made my stay at Caltech a most enjoyable experience.

My dear wife Batia constantly supported me with her love and care, stoically putting up with my absent-mindness.

My mother-in-law always encouraged me. In our last conversation, she told me how much she wanted to be present at my graduation ceremony. It couldn't be. She died two months before this thesis was submitted. I remember her with deep affection.

My family in Los Angeles and in Argentina gave me always great support. In particular, I want to thank my father for more things than can be listed here. With all my love and affection, this thesis is dedicated to him.

ABSTRACT

This thesis is divided into four independent chapters and two appendices.

Chapter I deals with the following generalization of the birthday surprise problem: how many people we need to interview on the average until either r birthdays occur k times each or one birthday occurs $k+1$ times. If $r = 1$, we obtain the usual "birthday surprise" number. We verify that our formula generalizes previous known results. We give asymptotic estimates for the birthday surprise number using a theorem proved in appendix I.

In chapter II, we present accurate and easily evaluated estimates for the average lifetime of a semiconductor RAM memory protected by a single error correcting, doubly error detecting (SEC-DED) code. This problem is somehow related to the one in chapter I. As an application, we give an analysis of the benefits of soft error "scrubbing" when both hard and soft errors are present. We also discuss two methods for increasing the lifetime of a computer memory: adding s rows of spare chips and implementing 2-ECC. We close the chapter by comparing the two methods.

In chapter III, we describe a class of burst error correcting array codes. We prove the fundamental properties of these codes.

Patel and Hong have constructed a code that can correct any track error or two track erasures in a 9-track magnetic tape. In chapter IV, we extend the construction to codes that can correct higher numbers of track errors and erasures. The result is a new family of codes, the $\hat{B}(n, m)$-codes.

In appendix I, we prove an important theorem used for asymptotic estimates of integrals. This theorem is used in chapters I and II.

In appendix II, we give a discussion of the Poisson approximation used in chapter II assuming a simplified situation.

Each chapter is an independent entity with introduction and references.

Being conscious of the logical contradictions of the term, let's point out that this thesis is self-contained.

CONTENTS

CHAPTER I

EXTENSIONS OF THE BIRTHDAY SURPRISE PROBLEM

1. Introduction

The classical birthday surprise problem deals with the following question: Suppose you interview a sequence of randomly selected people, making a note of their birthdays, until some birthday has occurred twice. How many people will you interview on the average? This number turns out to be 24.62, and if you wait until the same birthday has occurred three times, the number is 88.74 (not "about 83," as reported in [4]).

In this chapter we solve the following generalization: given $r \geq 1, k \geq 2$, how many people do we have to interview on the average until r birthdays occur k times each. However, there is a small problem that has to be taken into account. It might happen that r birthdays k times each will never occur. In effect, suppose that some birthday occurs $k+1$ times *before* r birthdays occur k times each. In that case, we have two options: either stop, or continue until r birthdays occur *at least* k times each. We adopt the first point of view. However, both averages are very close. Moreover, it can be proved that, for planets with a very large number of days in the year, the two averages are asymptotically equal (see [1]).

In particular, when $r = 1, k = 2$, we obtain the usual birthday surprise.

Our asymptotic estimates will be more precise than the ones given in [4].

2. A generalization of the "birthday surprise" problem

Let us formulate the birthday problem in the following way: suppose we place randomly and independently balls in M cells. We wish to compute the average number of balls we have to place until either r cells contain k balls each, or one cell contains $k+1$ balls. To solve this problem, we shall introduce an apparent artificiality: We make the times between the placing of the balls independent exponentially distributed random variables. However, we shall see that this artifice actually simplifies the calculation of the expectations (the key is Wald's identity).

2.1. Definition

Assume at each of the arrivals W_i, $i \geq 1$, of a Poisson process of rate 1, a ball is placed at random into one of M cells. Then, $T_r(M,k)$ denotes the first time that, either r cells contain k balls each, or one cell contains $k+1$ balls. $N_r(M,k)$ denotes the number of balls placed by time $T_r(M,k)$, and $B_r(M,k)$ is the expected number of balls placed by time $T_r(M,k)$, i.e., $B_r(M,k) = E(N_r(M,k))$ $(r \geq 1, k \geq 2)$.

2.2. Theorem

Let $r \geq 0, k \geq 2$, and $T_0(M,k) = 0$. Then,

$$E(T_{r+1}(M,k)) = E(T_r(M,k)) + \binom{M}{r} \frac{M}{(k!)^r} \int_0^\infty e^{-Mx} (S_k(x))^{M-r} x^{kr}\, dx \quad (1)$$

where

$$S_k(x) = \sum_{j=0}^{k-1} \frac{x^j}{j!}$$

Proof: The arrival of balls in each cell is a Poisson process of intensity $\frac{1}{M}$. Whenever either r cells contain k balls each or one cell contains $k+1$ balls, we shall say that an (M,r,k)-*success* has occurred. Let $R_r(M,k,t)$ denote the probability that, by time t, an (M,r,k)-success has *not* occurred. Hence, an $(M,r+1,k)$-success will not occur, if either an (M,r,k)-success has not occurred, or *exactly* r cells contain k balls each. Thus,

$$R_{r+1}(M,k,t) = R_r(M,k,t) + \binom{M}{r}\left(\sum_{j=0}^{k-1}\left(\frac{t}{M}\right)^j \frac{e^{-t/M}}{j!}\right)^{M-r}\left(\left(\frac{t}{M}\right)^k \frac{e^{-t/M}}{k!}\right)^r$$

so,

$$R_{r+1}(M,k,t) = R_r(M,k,t) + \binom{M}{r}\frac{e^{-t}}{(k!)^r}\left[S_k\left(\frac{t}{M}\right)\right]^{M-r}\left(\frac{t}{M}\right)^{kr} \qquad (2)$$

From (2), by induction and the fact that we have a polynomial times a decaying exponential,

$$\lim_{t\to\infty} tR_r(M,k,t) = 0 \qquad (3)$$

for all r.

The mean time until an (M,r,k)-success occurs is then given by

$$E(T_r(M,k)) = -\int_0^\infty tR_r'(M,k,t)\,dt = \int_0^\infty R_r(M,k,t)\,dt \qquad (4)$$

The last equality is obtained integrating by parts and using (3). From (2) and (4), making the change of variable $\frac{t}{M} = x$, we obtain (1). ∎

2.3. Corollary

Let $B_0(M,k) = 0$, *then*

$$B_{r+1}(M,k) = B_r(M,k) + \binom{M}{r}\frac{M}{(k!)^r}\int_0^\infty e^{-Mx}[S_k(x)]^{m-r}x^{kr}\,dx \qquad (5)$$

Proof: Call T_i the time between the $(i-1)$-th and the i-th arrival, $i \geq 1$. Then $T_1, T_2, \ldots\ldots$ are independent random variables with common distribution $1 - e^{-t}$ and $T_r(M,k) = T_1 + T_2 + \ldots + T_{N_r(M,k)}$.

According to Wald's identity ([3], page 217), we have

$$E(T_r(M,k)) = E(T_1)E(N_r(M,k)) = E(T_1)B_r(M,k)$$

But $E(T_1) = 1$, hence,

$$E(T_r(M,k)) = B_r(M,k) \tag{6}$$

From (1) and (6), (5) follows. ∎

Notice that in the particular case $r = 0$, (5) gives

$$B_1(M,k) = M \int_0^\infty e^{-Mx}[S_k(x)]^M \, dx \tag{7}$$

Formula (7) is the usual birthday surprise number ([2],[4]).

3.Asymptotic estimates

The next theorem will be proved in appendix I. We shall use it to obtain asymptotic estimates for $B_r(M,k)$, M a large number.

3.1. Theorem

Let $F(M) = \int_0^\infty g(x)e^{Mh(x)}\,dx$, *where* g *is continuous and positive when* $x > 0$, h *is infinitely differentiable for* $x \geq 0$, $h(x) < h(0)$ *for all* $x > 0$, $h'(0) = h''(0) = \cdots = h^{(k-1)}(0) = 0$ *for some* $k \geq 1$, $h^{(k)}(0) < 0$,

$\lim_{x\to\infty} h(x) = -\infty$, $\int_0^\infty g(x)e^{h(x)}\,dx$ *converges, and let* $h(x) = a_0 + \sum_{j=k}^\infty a_j x^j$, $g(x) = \sum_{j=0}^\infty b_j x^j$ *for* $0 \le x \le \delta$ *for some* $\delta > 0$, *then,*

$$F(M) \sim \left(\sum_{\nu=0}^{\infty} d_\nu M^{-\frac{\nu+1}{k}}\right) e^{Mh(0)} \tag{8}$$

where

$$d_\nu = \frac{(-a_k)^{-\frac{\nu+1}{k}}}{k} \sum_{j=0}^{\nu} c_{j,\nu-j}(-a_k)^{-j}\Gamma\left(j + \frac{\nu+1}{k}\right) \tag{9}$$

and

$$g(v)\exp\left(u\sum_{s=0}^{\infty} a_{k+1+s}v^s\right) = \sum_{i,j} c_{i,j}u^i v^j \tag{10}$$

3.2. Corollary

$$B_1(M,k) = \sum_{j=1}^{k} R_j(k)(k!)^{j/k}\Gamma\left(1+\frac{j}{k}\right)M^{1-\frac{j}{k}} + O\left(M^{-\frac{1}{k}}\right) \tag{11}$$

where the terms $R_j(k)$ *are rational functions of* k.

Proof: Equation (7) can be written as

$$B_1(M,k) = M\int_0^\infty e^{Mh_k(x)}\,dx$$

where $h_k(x) = \log(S_k(x)) - x$.

We are in the conditions of theorem 3.1., with

$$h_k(0) = h_k'(0) = \cdots = h_k^{(k-1)}(0) = 0 \quad , \quad h_k^{(k)}(0) = -1.$$

Hence, from equation (8), we obtain

$$B_1(M,k) = \sum_{\nu=0}^{k-1} d_\nu M^{1-\frac{\nu+1}{k}} + O\left(M^{-\frac{1}{k}}\right) \tag{12}$$

where, using (9), (10), and the fact that $a_k = (-k!)^{-1}$,

$$d_\nu = \frac{(k!)^{\frac{\nu+1}{k}}}{k} \sum_{s=0}^{\nu} c_{s,\nu-s}(k!)^s \Gamma\left(s + \frac{\nu+1}{k}\right) \tag{13}$$

and

$$\exp\left(u \sum_{s=0}^{\infty} a_{k+1+s} v^s\right) = \sum_{i,j} c_{ij} u^i v^j \tag{14}$$

Notice that $d_0 = (k!)^{\frac{1}{k}} \Gamma\left(1 + \frac{1}{k}\right)$, so $R_1(k) = 1$.

Now let $\nu \geq 1$. Since $c_{0,\nu} = 0$ from (14), by properties of the Γ function, (13) becomes

$$d_\nu = \frac{(k!)^{\frac{\nu+1}{k}}}{k} \Gamma\left(1 + \frac{\nu+1}{k}\right) \sum_{s=1}^{\nu} c_{s,\nu-s}(k!)^s \prod_{i=1}^{s-1} \left(i + \frac{\nu+1}{k}\right) \tag{15}$$

where $\prod_{i=1}^{0} \left(i + \frac{\nu+1}{k}\right) = 1$.

Replacing d_0 and (15) in (12), we obtain (11), where, calling $j = \nu + 1$, $R_1(k) = 1$ and for $j \geq 2$,

$$R_j(k) = \sum_{s=1}^{j-1} c_{s,j-1-s}[(k-1)!]^s \prod_{i=1}^{s-1} (ik + j) \tag{16}$$

This completes the proof. ∎

Approximating by the first term in (11), since $R_1(k) = 1$, as $M \to \infty$, we obtain

$$B_1(M,k) \sim (k!)^{\frac{1}{k}} \Gamma\left(1 + \frac{1}{k}\right) M^{1-\frac{1}{k}} \tag{17}$$

Estimate (17) was first obtained by Klamkin and Newman ([4]). Using this estimate in the case $M = 365$ and $k = 3$, they concluded that the

triple birthday surprise number (ie., $B_1(365,3)$) is "about 83." However, the correct value is 88.739 ([1]). The problem is that the asymptotic estimate (17) does not contain enough terms. A more precise estimate is given by (11). Thus, we need to calculate the coefficients $R_j(k)$ for $2 \le j \le k$, which are given by (16). Using (14), the problem is reduced to evaluate the numbers a_j, where $h_k(x) = \sum_{j=0}^{\infty} a_j x^j$. An explicit formula for the relevant numbers a_j is given by the following lemma.

3.3. Lemma

For $0 \le i \le k-1$,

$$a_i = 0 \quad \textit{and} \quad a_{k+i} = \frac{(-1)^{i+1}}{i!(k+i)(k-1)!} \tag{18}$$

Proof: We had $h_k(x) = -x + \log S_k(x) = \sum_{i=0}^{\infty} a_i x^i$.

Let

$$y_k(x) = \sum_{j=0}^{k-2} \frac{x^j}{(j+1)!} \quad , \quad z_k = \sum_{j=0}^{\infty} \frac{x^j}{(k+j)!},$$

then,

$$h_k(x) = -x + \log(1 + x y_k) = -x + \sum_{i=1}^{\infty} \frac{(-1)^{i+1}}{i} x^i y_k^i \tag{19}$$

Notice,

$$\begin{aligned} 0 &= -x + \log e^x \\ &= -x + \log\left(\sum_{j=0}^{\infty} \frac{x^j}{j!}\right) \end{aligned}$$

$$= -x + \log(1 + xy_k + x^k z_k)$$
$$= -x + \sum_{i=1}^{\infty} \frac{(-1)^{i+1}}{i} (xy_k + x^k z_k)^i$$
$$= -x + \sum_{i=1}^{\infty} \frac{(-1)^{i+1}}{i} \sum_{j=0}^{i} \binom{i}{j} (xy_k)^{i-j} (x^k z_k)^j$$
$$= -x + \sum_{i=1}^{\infty} \frac{(-1)^{i+1}}{i} x^i y_k^i + \sum_{i=1}^{\infty} \frac{(-1)^{i+1}}{i} \sum_{j=1}^{i} \binom{i}{j} y_k^{i-j} z_k^j x^{i-j+kj}$$

so, using (19), we obtain

$$h_k(x) = \sum_{i=1}^{\infty} \frac{(-1)^i}{i} x^i \sum_{j=1}^{i} \binom{i}{j} y_k^{i-j} z_k^j x^{(k-1)j} \tag{20}$$

The smallest power of x in the right-hand side of (20) is k, hence $a_i = 0$ for $0 \leq i \leq k-1$, proving the first part of the lemma.

Consider now the function $p_k(x) = \sum_{i=0}^{k-1} (-1)^{i+1} x^i y_k^i z_k$.

Notice that, for $0 \leq j \leq k-1$,

$$a_{k+j} = \mathrm{coef}_j\,(p_k(x)) = \sum_{i=0}^{j} (-1)^{i+1} \mathrm{coef}_{j-i}(y_k^i z_k) \tag{21}$$

and

$$\mathrm{coef}_{j-i}(y_k^i z_k) = \sum_{\nu=0}^{j-i} \mathrm{coef}_\nu(y_k^i) \mathrm{coef}_{j-i-\nu}(z_k) = \sum_{\nu=0}^{j-i} \frac{\mathrm{coef}_\nu(y_k^i)}{(k+j-i-\nu)!} \tag{22}$$

Let $b_{i,\nu} = \mathrm{coef}_\nu(y_k^i)$, then, by (21) and (22),

$$a_{k+j} = \sum_{i=0}^{j} (-1)^{i+1} \sum_{\nu=0}^{j-i} \frac{b_{i,\nu}}{(k+j-i-\nu)!}$$
$$= \sum_{\nu=0}^{j} \frac{1}{(k+j-\nu)!} \sum_{i=0}^{\nu} (-1)^{i+1} b_{i,\nu-i} \tag{23}$$

Claim:

$$\sum_{i=0}^{m}(-1)^{i+1}b_{i,m-i} = \frac{(-1)^{m+1}}{m!} \tag{24}$$

Induction on m. If $m = 0$, then $-b_{0,0} = -1$, so (24) is true.

Assume (24) is true for $t \leq m-1, m \geq 1$. Note that, for $t \geq 1$,

$$b_{0,t} = \text{coef}_t(y_k^0) = 0,$$

so (24) can also be written as $\sum_{i=1}^{m}(-1)^{i+1}b_{i,m-i} = \frac{(-1)^{m+1}}{m!}$.

For $i \geq 1$,

$$b_{i,m-i} = \sum_{l=0}^{m-i}\text{coef}_l(y_k)\text{coef}_{m-i-l}(y_k^{i-1}) = \sum_{l=0}^{m-i}\frac{b_{i-1,m-i-l}}{(l+1)!} \tag{25}$$

Thus

$$\begin{aligned}\sum_{i=0}^{m}(-1)^{i+1}b_{i,m-1} &= \sum_{i=1}^{m}(-1)^{i+1}\sum_{l=0}^{m-i}\frac{b_{i-1,m-i-l}}{(l+1)!} \\ &= \frac{1}{m!} + \sum_{i=2}^{m}(-1)^{i+1}\sum_{l=0}^{m-i}\frac{b_{i-1,m-i-l}}{(l+1)!} \\ &= \frac{1}{m!} + \sum_{l=1}^{m-1}\frac{1}{l!}\sum_{i=1}^{m-l}(-1)^i b_{i,m-l-i} \end{aligned} \tag{26}$$

By induction, $\sum_{i=1}^{m-l}(-1)^i b_{i,m-l-i} = \frac{(-1)^{m-l}}{(m-l)!}$, so, replacing in (26), we get

$$\sum_{i=0}^{m}(-1)^{i+1}b_{i,m-i} = \frac{1}{m!} + \sum_{l=1}^{m-1}\frac{(-1)^{m-l}}{l!(m-l)!} = \sum_{l=0}^{m-1}\frac{(-1)^l}{l!(m-l)!}$$

hence,

$$(m!)\sum_{i=0}^{m}(-1)^{i+1}b_{i,m-i} + (-1)^m = \sum_{l=0}^{m}(-1)^l\binom{m}{l} = (1-1)^m = 0$$

and solving for $\sum_{i=0}^{m}(-1)^{i+1}b_{i,m-i}$ we obtain (24).

Replacing (24) in (23), we have

$$a_{k+j} = \sum_{\nu=0}^{j} \frac{(-1)^{\nu+1}}{\nu!(k+j-\nu)!} = \frac{\sum_{\nu=0}^{j}(-1)^{\nu+1}\frac{j!(k+j)!}{(\nu!)(k+j-\nu)!}}{j!(k+j)!}$$

Calling

$$f(x) = j!\left[-1 + \sum_{\nu=1}^{j} \frac{(-1)^{\nu+1}}{\nu!}(x+j)(x+j-1)\cdots(x+j-\nu+1)\right]$$

this becomes

$$a_{k+j} = \frac{f(k)}{j!(k+j)!} \tag{27}$$

Claim: $(-1)^{j+1}f(x) = x(x+1)\cdots(x+j-1)$.

Since $(-1)^{j+1}f(x)$ is monic, suffices to show that $0, -1, \cdots, -(j-1)$ are roots of $f(x)$. Notice, for $0 \le i \le j-1$,

$$\begin{aligned} -\frac{f(-i)}{j!} &= 1 + \sum_{\nu=1}^{j-i} \frac{(-1)^{\nu}}{\nu!}(j-i)(j-i-1)\cdots(j-i-\nu+1) \\ &= \sum_{\nu=0}^{j-i}(-1)^{\nu}\binom{j-i}{\nu} = (1-1)^{j-i} = 0 \end{aligned}$$

In particular, $(-1)^{j+1}f(k) = k(k+1)\cdots(k+j-1)$.

Replacing in (27), we obtain

$$a_{k+j} = (-1)^{j+1}\frac{k(k+1)\cdots(k+j-1)}{j!(k+j)!} = \frac{(-1)^{j+1}}{j!(k+j)(k-1)!}$$

This completes the proof. ∎

Using (14),(16) and (18), we can now calculate $R_j(k)$. The first four values of $R_j(k)$ are:

$$R_1(k) = 1$$
$$R_2(k) = \frac{1}{k+1}$$
$$R_3(k) = \frac{3k+5}{2(k+1)^2(k+2)}$$
$$R_4(k) = \frac{16k^2+66k+62}{6(k+1)^3(k+2)(k+3)}$$

For example, we obtain

$$B_1(M,2) = \sqrt{\frac{\pi M}{2}} + \frac{2}{3} + O\left(M^{-\frac{1}{2}}\right) \tag{28}$$

$$B_1(M,3) = 6^{\frac{1}{3}}\Gamma\left(\frac{4}{3}\right)M^{\frac{2}{3}} + \frac{6^{\frac{2}{3}}}{4}\Gamma\left(\frac{5}{3}\right)M^{\frac{1}{3}} + \frac{21}{40} + O\left(M^{-\frac{1}{3}}\right) \tag{29}$$

With $M = 365$, (28) yields $B_1(365,2) \cong 24.611$, whereas the right answer obtained by exact integration is 24.617 ([2],[5]). Similarly, (29) yields $B_1(365,3) \cong 88.725$ instead of the correct value 88.739 ([1]).

We give next asymptotic estimates for $B_r(M,k)$ when $r \geq 2$.

3.4. Theorem

For $r \geq 1$,

$$B_{r+1}(M,k) = B_r(M,k) + \frac{1}{M^r}\binom{M}{r}\sum_{j=1}^{k} f_j(k)\Gamma\left(1+\frac{j}{k}\right) (k!)^{\frac{j}{k}} M^{1-\frac{j}{k}} + O\left(M^{-\frac{1}{k}}\right) \tag{30}$$

and

$$B_{r+1}(M,k) \sim B_r(M,k) + (k!)^{\frac{1}{k}} \left(\prod_{i=1}^{r} \frac{(i-1)k+1}{ik} \right) \Gamma\left(1+\frac{1}{k}\right) M^{1-\frac{1}{k}} \quad (31)$$

Proof: We need an asymptotic estimate for the integral in (1) when $r \geq 1$. Let us write

$$I = \int_0^\infty e^{-Mx} [S_k(x)]^{M-r} x^{kr}\, dx = \int_0^\infty g(x) e^{M h_k(x)}\, dx \quad (32)$$

where $g(x) = \left(\frac{x^k}{S_k(x)}\right)^r$ and $h_k(x) = -x + \log S_k(x)$.

We satisfy the conditions of theorem 3.1. Applying it to (32) and replacing in (1), we obtain (30). If we approximate the sum in (30) by its first term, since $\lim_{M\to\infty} \frac{1}{M^r}\binom{M}{r} = \frac{1}{r!}$, we get (31). ∎

Using (17) and (31) repeatedly, we obtain

$$B_{r+1}(M,k) \sim \left(\sum_{j=0}^{r} \prod_{i=1}^{j} \frac{(i-1)k+1}{ik} \right) (k!) \frac{1}{k} \Gamma\left(1+\frac{1}{k}\right) M^{1-\frac{1}{k}} \quad (33)$$

where $\prod_{i=1}^{0} \frac{(i-1)k+1}{ik} = 1$.

When $k = 2$, (31) gives the aesthetic formula

$$B_{r+1}(M,2) \sim B_r(M,2) + \frac{1.3.5 \cdots (2r-1)}{2.4.6 \cdots (2r)} B_1(M,2) \quad (34)$$

Estimate (33) gives a good idea of the size of $B_{r+1}(M,k)$, but has to be handled with care since the error also tends to infinity in general. In order to obtain a more precise estimate, we need to use (30). For example,

$$B_{r+1}(M,2) = B_r(M,2) + \frac{(M-1)(M-2)\cdots(M-r+1)}{M^{r-1}} \left(\frac{1.3\cdots(2r-1)}{2.4\cdots 2r}\sqrt{\frac{\pi M}{2}} + \frac{-r+2}{3} \right) + O\left(M^{-\frac{1}{2}}\right) \quad (35)$$

and

$$B_{r+1}(M,3) = B_r(M,3) + \frac{(M-1)(M-2)\cdots(M-r+1)}{M^{r-1}} \left[6^{\frac{1}{3}} \frac{1.4.7\cdots(3r-2)}{3.6.9\cdots(3r)} \Gamma\left(\frac{4}{3}\right) M^{\frac{2}{3}} + 36^{\frac{1}{3}} \frac{5.8\cdots(3r-1)}{3.6.9\cdots(3r)} \left(\frac{-r+2}{4}\right) \Gamma\left(\frac{5}{3}\right) M^{\frac{1}{3}} + \frac{5r^2-33r+42}{80} \right] + O\left(M^{-\frac{1}{3}}\right) \quad (36)$$

When $r = 1$, (34) gives $B_2(M,2) \sim \frac{3}{2}B_1(M,2)$. Surprisingly, we have in fact that $B_2(M,2) = \frac{3}{2}B_1(M,2)$ for all values of M. We conclude this chapter by proving this result.

3.5. Theorem

$$B_2(M,2) = \frac{3}{2}B_1(M,2) \qquad \text{for all} \quad M \geq 1 \quad (37)$$

Proof: From (5) we have

$$B_2(M,2) = B_1(M,2) + \frac{M^2}{2}\int_0^\infty e^{-Mx}(1+x)^{M-1}x^2\,dx \quad (38)$$

Let $I = \int_0^\infty e^{-Mx}(1+x)^{M-1}x^2\,dx$. I can be written as

$$I = -2I_0 + I_1 + I_{-1} \quad (39)$$

where

$$I_0 = \int_0^\infty e^{-Mx}(1+x)^M\,dx$$
$$I_1 = \int_0^\infty e^{-Mx}(1+x)^{M+1}\,dx$$
$$I_{-1} = \int_0^\infty e^{-Mx}(1+x)^{M-1}\,dx$$

Integrating by parts,

$$I_1 = \frac{1}{M} + \frac{M+1}{M} I_0 \tag{40}$$

and

$$I_{-1} = -\frac{1}{M} + I_0 \tag{41}$$

Replacing (40) and (41) in (39), we obtain

$$I = \frac{I_0}{M} \tag{42}$$

so (38) becomes

$$B_2(M,2) = B_1(M,2) + \frac{M}{2}\int_0^\infty e^{-Mx}(1+x)^M\,dx \tag{43}$$

Since $M\int_0^\infty e^{-Mx}(1+x)^M\,dx = B_1(M,2)$ by (5), (43) yields (37). ∎

For instance, $B_2(365,2) = \frac{3}{2}B_1(365,2) = 36.93$, i.e., we need to interview around 37 people on the average in order to obtain either two double birthdays or one triple birthday.

Theorem 3.5. is too good to be true in general. In fact, we would like to have equality in (34), but that does not occur.

In effect, consider $B_3(M,2)$. From (5),

$$B_3(M,2) = B_2(M,2) + \frac{M^2(M-1)}{8}\int_0^\infty e^{-Mx}(1+x)^{M-2}x^4\,dx \tag{44}$$

Now, let $I = \int_0^\infty e^{-Mx}(1+x)^{M-2}x^4\,dx$.

Since $x^4 = (x+1)^4 - 4(x+1)^3 + 6(x+1)^2 - 4(x+1) + 1$,

$$I = I_2 - 4I_1 + 6I_0 - 4I_{-1} + I_{-2} \tag{45}$$

where $I_j = \int_0^\infty e^{-Mx}(1+x)^{M+j}\,dx, \quad -2 \le j \le 2.$

Integrating by parts and using (40),

$$I_2 = \frac{1}{M} + \frac{M+2}{M}I_1 = \frac{1}{M} + \frac{M+2}{M^2} + \frac{(M+2)(M+1)}{M^2}I_0 \tag{46}$$

Similarly, using (41),

$$I_{-2} = -\frac{1}{M-1} + \frac{M}{M-1}I_{-1} = -\frac{2}{M-1} + \frac{M}{M-1}I_0 \tag{47}$$

Replacing (46), (47), (40) and (41) in (45), we obtain

$$I = -\frac{2}{M^2(M-1)} + \frac{3M-2}{M^2(M-1)}I_0 \tag{48}$$

and replacing (48) in (44),

$$\begin{aligned} B_3(M,2) &= B_2(M,2) - \frac{1}{4} + \frac{3}{8}MI_0 - 2I_0 \\ &= B_2(M,2) + \frac{3}{8}B_1(M,2) - \frac{1}{4} - 2I_0 \end{aligned} \tag{49}$$

Clearly, as $M \to \infty$ in (49), (34) holds, but we do not have equality. Life is hard!

References

[1] M. Blaum, I. Eisenberger, G. Lorden and R. J. McEliece, "More about the birthday surprise," to appear.

[2] R. M. F. Goodman and R. J. McEliece, "Hamming Codes, Computer Memories and the Birthday Surprise," Proc. 20th Allerton Conference on Communication, Control and Computing, pp. 672-679.

[3] P. G. Hoel, S. C. Port and C. J. Stone, "Introduction to Probability Theory," Boston, Houghton-Mifflin,1971.

[4] M. S. Klamkin and D. J. Newman, "Extensions of the Birthday Surprise," J. Combinatorial Theory 3(1967), pp. 279-282.

[5] D. E. Knuth, "Sorting and Searching" (Vol. 3 in "The Art of Computer Programming"), Reading, Mass. Addison-Wesley, 1973.

CHAPTER II

AVERAGE LIFETIMES OF COMPUTER MEMORIES

1. Introduction

All modern computers have memories built from VLSI RAM chips. Individually these devices are highly reliable; any single chip can be expected to function for decades before failing. However, when many of these chips are combined into a single large computer memory, the expected waiting time until one of the component chips fails can be as small as a few hours. For this reason, almost all large computer memories are protected by single-error-correcting and double-error-detecting (SEC-DED) codes. Mathematically, these codes are just shortened $d = 4$ Hamming codes; the shortening is usually done in a hardware-efficient manner devised by Hsiao ([5]). The recent survey article by Chen and Hsiao ([4]) gives a very good overview of SEC-DED memory coding; but we shall summarize the important features of the coding architecture here.

Normally the memory is organized into an $M \times n$ rectangular array of chips (figure 1).

The first k chips in each row are information-carrying chips, while the remaining $r = n - k$ chips are parity-check chips. A typical example is a one megabyte memory board used by the VAX 11/750, which consists of $M = 4$ rows of 64K RAM chips, each row containing $k = 32$ data chips and 7 parity chips, corresponding to a $(39, 32) d = 4$ SEC-DED code.

We assume that each chip is organized internally as an $l \times l$ square array

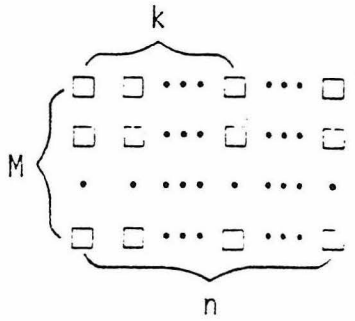

Figure 1

of bits as shown in figure 2 (for standard 4164 n-MOS 64K RAM chips, $l = 256$). Each n-bit codeword consists of one bit from each of the n chips in one row (figure 3).

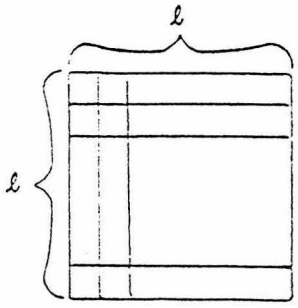

Figure 2

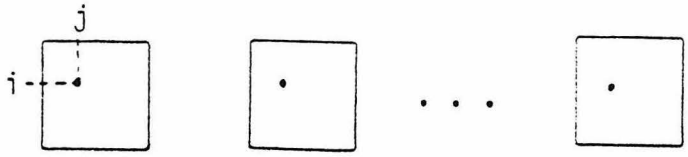

Figure 3: The (i, j)th codeword in one row of chips

In the following discussion, a chip *failure* will be taken to mean a situation in which one or more of the bits written on a chip cannot be reliably recovered.

These failures are traditionally classified as either "hard" (meaning that

the memory cells involved are permanently damaged, e.g., "stuck at" faults), or "soft" (meaning that a given bit has been somehow complemented but that the chip itself has suffered no structural damage).

Observation of real memories ([6]) shows that the single most common type of cell failure is a soft error affecting only one cell in one chip. These errors are caused by stray alpha-particles which can, under the right circumstances, change a logical "1" to a logical "0" without damaging the chip. However, several kinds of hard failures are observed to occur. A *single-cell failure*, which, as we have seen, can occur as a soft error, can also occur as a hard error. There are also several kinds or hard chip failures which cause bursts of errors in a chip. A *row- failure* occurs when all l cells in one row fail (this can be caused by a failure of one of the chip's row drivers). A *column-failure* occurs when all l cells in one column fail (this can be caused by a failure of one of the chip's column amplifiers). A short-circuit at a memory cell can cause a *row-column failure*, in which all the cells in either the same row or the same column as the affected cell fail. All four of these errors are illustrated in figure 4.

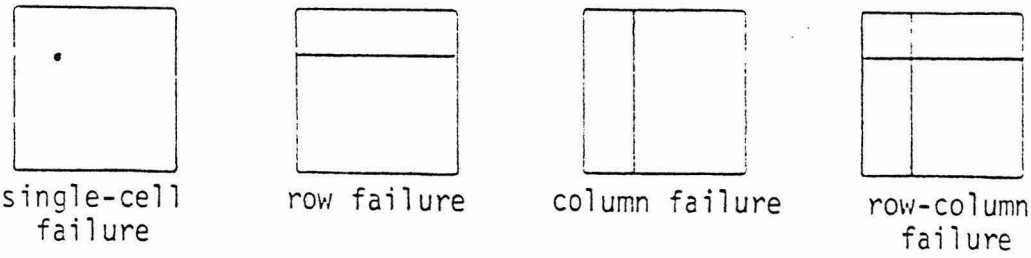

Figure 4

Also, a *catastrophic chip-failure* may occur, in which all cells in a chip

fail. This kind of failure, being very unusual, will be ignored in this work.

Of course the organization of the SEC-DED code guarantees that no failure, however catastrophic, which is confined to a single chip can cause two errors in any codeword, and so the code will correct any single chip failure. In fact, there are many combinations of chip failures that can be corrected by the code. Eventually, however, it is to be expected that enough chip failures will have occurred so that some codeword will have suffered two errors, at which point we have a *memory failure.* It is our object in this chapter to give accurate and easily evaluated estimates for the mean time between (memory) failures (MTBF) for memories protected by SEC-DED coding. In the next section we will present a model for the occurrences of the various types of chip failure, and use it to derive an estimate, based on the Poisson approximation, of the MTBF. In section 3, we shall give a simple asymptotic approximation to MTBF, when the number M of rows is large (e.g., which is the case in the CRAY-1 computer). In section 4, we shall give several numerical examples, using data typical for real chips. There we shall show that for one representative set of data, soft error "scrubbing," a technique which can be used to periodically purge the memory of soft errors, can increase the MTBF in a SEC-DED protected memory by as much as 40%. In the next two sections, we discuss two methods to extend the lifetime of a computer memory. The first method, to be discussed in section 5, consists of adding s additional rows of spare chips. Each time a chip fails, it is replaced by a spare chip. We give an estimate of the increment in MTBF. In section 6 we estimate MTBF when 2-ECC is implemented, that is, a

doubly-error- correcting triply-error-detecting (DEC-TED) code. Although a hardware implementation of the methods in sections 5 and 6 is possible, we are not aware of any application for commercial use. We close the chapter by comparing the two methods.

2. Models. Formula for MTBF

The *reliability* of a given chip (probability of no failure of any kind after t hours) is given by $e^{-\lambda t}$, where λ is a constant found experimentally ([6]). We have to distinguish between the four types of errors in figure 4, and so for future reference, we use the following notation:

A: row failure

B: column failure

C: single-cell failure

D: row-column failure

Let a, b, c, d be the relative frequencies of these four events. We assume that in a given chip, these four events occur independently, and that failures in one chip are independent from failures in all other chips. Thus, for example, the probability that after t hours a given row in chip has not yet failed is $e^{-\lambda a t/l}$. The key to finding the MTBF is the calculation of the *row reliability function* $R(t)$, which is defined as follows:

$$R(t) = \Pr\{\text{an uncorrectable pattern of chip failures has } \textit{not} \text{ occurred in row } i \text{ at time } t\}$$

Since rows fail independently, the reliability of the entire array of M rows is $R(t)^M$, and so

$$MTBF = \int_0^\infty t\left(-\frac{d}{dt}(R(t))^M\right)\,dt = \int_0^\infty (R(t))^M\,dt \qquad (1)$$

All of our results are based on equation (1).

Consider now a row of chips protected by SEC-DED coding. An uncorrectable failure will *not* occur in each of the following events:

I. Only row or single cell failures occur such that there is no more than one failure in a codeword.

II. The same thing applies to column and single-cell failures.

III. Exactly one row-column failure and corresponding single-cell failures occur.

Notice that a row and a column failure will make the whole system fail. Some thought shows that the system will survive only under events I, II and III. Let us call $R_1(t)$, $R_2(t)$, $R_3(t)$ the probabilities of events I, II and III, respectively.

In order to find $R_1(t)$, we focus on a single row, say row i. The probability that no two cells (i,j) will fail, if we assume only events A or C are occurring, is then, assuming the Poisson approximation,

$$e^{-a\lambda nt/l}\left[e^{-c\lambda nt/l^2}\left(1+\frac{c}{l^2}\lambda nt\right)\right]^l + e^{-c\lambda nt/l}\left(e^{-a\lambda nt/l}\,\frac{a}{l}\lambda nt\right) \qquad (*)$$

But we have l rows, each one failing independently, so we must take expression $(*)$ to the power l and multiply this power by $e^{-(b+d)\lambda nt}$, the probability that neither column nor row-column errors occur. Notice that we are assuming that each row fails according to a Poisson process, i.e., the prob-

ability of exactly j failures in any row is $\frac{(\lambda nt)^j}{j!}e^{-\lambda nt}$. This approximation is very good for typical values of n (see [3]). After some easy manipulations,

$$R_1(t) = e^{-\lambda nt}\left[\left(1+\frac{c}{l^2}\lambda nt\right)^l + \frac{a}{l}\lambda nt\right]^l \tag{2}$$

Similarly, we get

$$R_2(t) = e^{-\lambda nt}\left[\left(1+\frac{c}{l^2}\lambda nt\right)^l + \frac{b}{l}\lambda nt\right]^l \tag{3}$$

In order to find $R_3(t)$, observe that when a row-column failure occurs, single-cell failures may occur in the corresponding $(l-1)^2$ cells left, thus,

$$R_3(t) = e^{-(a+b)\lambda nt}\left[(e^{-d\lambda nt}\, d\lambda nt)e^{-c\lambda nt}\left(1+\frac{c}{(l-1)^2}\lambda nt\right)^{(l-1)^2}\right]$$

or,

$$R_3(t) = e^{-\lambda nt}\left[\left(1+\frac{c}{(l-1)^2}\lambda nt\right)^{(l-1)^2} d\lambda nt\right] \tag{4}$$

Putting (2), (3) and (4) together, we get the intimidating expression

$$R(t) = e^{-\lambda nt}\left\{\left[\left(1+\frac{c}{l^2}\lambda nt\right)^l + \frac{a}{l}\lambda nt\right]^l + \left[\left(1+\frac{c}{l^2}\lambda nt\right)^l + \frac{b}{l}\lambda nt\right]^l + \right.$$
$$\left. + \left(1+\frac{c}{(l-1)^2}\lambda nt\right)^{(l-1)^2} d\lambda nt - \left(1+\frac{c}{l^2}\lambda nt\right)^{l^2}\right\} \tag{5}$$

The last term has to be substracted since events I and II are not disjoint, event C lies in the intersection.

Before proceeding further, we can see that formula (5) generalizes previously known results. If we take $c = 0$ as in [3], example 3, we get (now $a+b+d=1$)

$$R(t) = e^{-\lambda nt}\left[\left(1+\frac{a}{l}\lambda nt\right)^l + \left(1+\frac{b}{l}\lambda nt\right)^l + d\lambda nt - 1\right] \tag{6}$$

Making $l \to \infty$,

$$R(t) = e^{-\lambda nt}(e^{a\lambda nt} + e^{b\lambda nt} + d\lambda nt - 1) \tag{7}$$

which is exactly the formula obtained in [3], example 3. (7) is a good approximation of (6), since l is in general a large number.

Finally, replacing (5) in (1) and making the change of variable $\lambda nt = x$, we obtain

$$MTBF = \frac{1}{\lambda n}\int_0^\infty e^{-Mx}\left\{\left[\left(1+\frac{c}{l^2}x\right)^l + \frac{a}{l}x\right]^l + \left[\left(1+\frac{c}{l^2}x\right)^l + \frac{b}{l}x\right]^l + \right.$$
$$\left. + d\left(1+\frac{c}{(l-1)^2}x\right)^{(l-1)^2} x - \left(1+\frac{c}{l^2}x\right)^{l^2}\right\}^M dx \tag{8}$$

Although the integral in (8) is somewhat complicated, if M is small it can be easily evaluated using numerical methods. On the other hand, if M is large, we can use asymptotic methods to estimate it.

3. The case of large M. Asymptotic approximations.

Let

$$g(x) = \left[\left(1+\frac{c}{l^2}x\right)^l + \frac{a}{l}x\right]^l + \left[\left(1+\frac{c}{l^2}x\right)^l + \frac{b}{l}x\right]^l +$$
$$+ d\left(1+\frac{c}{(l-1)^2}x\right)^{(l-1)^2} x - \left(1+\frac{c}{l^2}x\right)^{l^2}$$

and $h(x) = -x + \log g(x)$, then (8) can be written as

$$MTBF = \frac{1}{\lambda n} \int_0^\infty e^{Mh(x)}\, dx \tag{9}$$

We are interested in finding the coding gain (CG) with respect to the unprotected memory. As the $MTBF$ of the unprotected memory is $1/\lambda kM$, dividing the expression in (9) by this value, we obtain

$$CG = \frac{k}{n} M \int_0^\infty e^{Mh(x)}\, dx \tag{10}$$

In many applications, M is very large. From now on, we shall assume that. We have to estimate the integral $\int_0^\infty e^{Mh(x)}\, dx$. Applying theorem 3.1. of chapter I, and approximating by the first term, since $h(0) = 0$, $h'(0) = 0$ and $h''(0) < 0$, then

$$\int_0^\infty e^{Mh(x)}\, dx \sim \left(\frac{\pi}{2M(-h''(0))} \right)^{\frac{1}{2}} \qquad (M \to \infty) \tag{11}$$

Hence, from (10),

$$CG \sim \frac{k}{n} \left(\frac{\pi M}{2(-h''(0))} \right)^{\frac{1}{2}} \qquad (M \to \infty) \tag{12}$$

Finding $h''(0)$ is arduous although straightforward. Doing the evaluation and some algebraic manipulations, we get $h(0) = 0$, $h'(0) = 0$ and

$$h''(0) = -(d^2 + 2ab + 2ad + 2bd) - \frac{a^2 + b^2 + 2ac + 2bc}{l} - \frac{c^2}{l^2} \tag{13}$$

Notice, $h''(0) < 0$ as required. Replacing in (12), we get

$$CG \sim \frac{k}{n}\left(\frac{\pi M}{2\left(d^2+2ab+2ad+2bd+\frac{a^2+b^2+2ac+2bc}{l}+\frac{c^2}{l^2}\right)}\right)^{\frac{1}{2}} \tag{14}$$

If we make $l \to \infty$, an approximation is

$$CG \sim \frac{k}{n}\left(\frac{\pi M}{2(d^2+2ab+2ad+2bd}\right)^{\frac{1}{2}} \tag{15}$$

Assume only events A, B and D occur, then $c = 0$ and $a + b + d = 1$. Hence, (15) becomes

$$CG \sim \frac{k}{n}\left(\frac{\pi M}{2(1-a^2-b^2)}\right)^{\frac{1}{2}} \tag{16}$$

This is exactly formula (15) in [3].

4. Numerical examples

In this section, we shall prove one of the assertions made at the beginning, that is, that "scrubbing" soft errors is useful when ECC is present.

Let us assume that $a = b$ and $d = 0$. This is not totally unrealistic since in general d is significantly smaller than a, b and c (see [6]).

Denote by $(MTBF)_A$ the mean time between failure if only row and column errors are taken into account, $(MTBF)_C$ if only single-cell errors are considered. As in general $MTBF = \frac{CG}{\lambda k M}$, using (14) with $2a\lambda$ instead of λ, $a^2 = b^2 = \frac{1}{4}$, $c = d = 0$, we get

$$(MTBF)_A \sim \frac{1}{2a\lambda n}\left(\frac{\pi}{M}\,\frac{l}{l+1}\right)^{\frac{1}{2}} \tag{17}$$

Similarly, taking $a = b = d = 0$ in (14), we obtain

$$(MTBF)_C \sim \frac{l}{c\lambda n}\left(\frac{\pi}{2M}\right)^{\frac{1}{2}} \tag{18}$$

Taking the ratio,

$$\frac{(MTBF)_A}{(MTBF)_C} \sim \frac{c}{al}\left(\frac{l}{2(l+1)}\right)^{\frac{1}{2}} \tag{19}$$

Referring to example 4 in [3], we have, $2a = .01$, $c = .99$, $l = 256$. Thus, replacing these values in (19),

$$\frac{(MTBF)_A}{(MTBF)_C} \sim .5$$

That is, row and column failures, although a lot less frequent than single cell failures, are roughly responsible twice as often for failures of the whole system in typical cases (observe that in example 4 of [3], $M = 4$ while here M is a big number).

However, we are interested in $MTBF$, the formula combining all kinds of errors (recall that we are assuming $a = b$, $d = 0$, $2a + c = 1$). Using (14) and $MTBF = \frac{CG}{\lambda kM}$, we get

$$MTBF \sim \frac{l}{\lambda n}\left(\frac{\pi}{2M[2a^2l^2 + (2a^2 + 4ac)l + c^2]}\right)^{\frac{1}{2}} \tag{20}$$

As $(MTBF)_A$ is smaller than $(MTBF)_C$ in general, it is a better approximation for $MTBF$. But how good an approximation? Taking the ratio,

$$\frac{MTBF}{(MTBF)_A} \sim \left[1 + \frac{(2a)^2 + 8ac}{(2a)^2 l} + \left(\frac{c}{2al}\right)^2\right]^{-\frac{1}{2}}\left(\frac{l+1}{l}\right)^{\frac{1}{2}} \tag{21}$$

Keeping the other variables fixed,

$$\lim_{l\to\infty} \frac{MTBF}{(MTBF)_A} = 1$$

That is, if l is a very large number, the approximation is good. As a check, observe that

$$\lim_{a\to 0} \frac{MTBF}{(MTBF)_A} = 0 \quad \text{and} \quad \lim_{a\to \frac{1}{2}} \frac{MTBF}{(MTBF)_A} = 1$$

as expected.

In our example, replacing $2a = .01$, $l = 256$ and $c = .99$ in (21), we obtain

$$\frac{MTBF}{(MTBF)_A} \sim .6$$

Hence the actual $MTBF$ is roughly 40% smaller than the one obtained ignoring single-cell errors. As most single-cell errors are soft errors (see [6]), techniques like scrubbing soft errors, combined with ECC, are useful in extending the lifetime of the system. The degree of usefulness is given by formula (21).

5. Error protection when s rows of spare chips are added

As usual, our memory is an $M \times n$ array of chips, the first k columns are information chips, but at the bottom s rows of spare chips are added (see figure 5).

The spare chips act as follows: each time a chip fails, a connection to a spare chip in the corresponding column is made. In practice, this means that

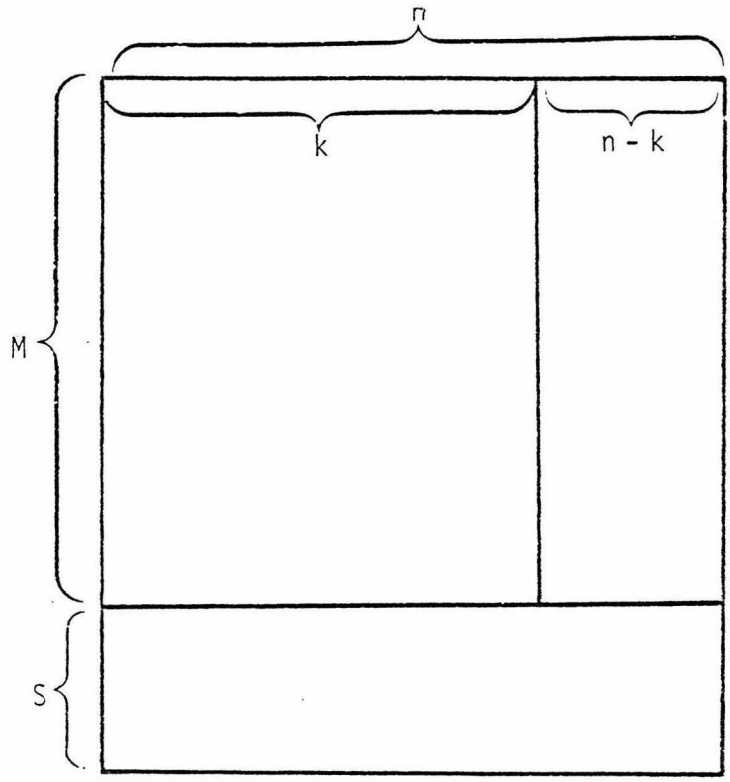

Figure 5: A computer memory with s rows of spare chips

the chip is replaced by one of the s spare chips in the corresponding column. Hence, s failures per column are tolerated before the code starts acting.

From (1) and the fact that the mean time between failure of the unprotected memory is $1/\lambda Mk$, we have

$$CG = \lambda Mk \int_0^\infty (R(t))^M \, dt \tag{22}$$

Denote by $(MTBF)_s$ the mean time between failure of a memory with error-coding-correction when s spare rows of chips are added. $(MTBF)_0$ denotes the usual case. Similarly, N_s is the number of failures that will make the whole memory fail and $(CG)_s$ is the coding gain with respect to the

unprotected $M \times k$ memory. $(CG)_0$ is given by equation (22) and was found in the previous sections when an SEC-DED code is implemented. We shall give upper and lower bounds on $(MTBF)_s$ and $(CG)_s$.

If we assume that spare chips do not fail when they are disconnected, by Wald's identity, we obtain

$$(MTBF)_s = \frac{1}{\lambda n M} E(N_s) \tag{23}$$

Since the best case occurs when all spare chips are used, and the worst case when a memory failure occurs when the first two nonreplaceable chips fail, we have

$$E(N_0) + B_2(n, s+1) \leq E(N_s) \leq E(N_0) + ns \tag{24}$$

For a definition of $B_2(n, s+1)$, see chapter I, definition 2.1. Using (23), we have

$$(MTBF)_0 + \frac{B_2(n, s+1)}{\lambda n M} \leq (MTBF)_s \leq (MTBF)_0 + \frac{s}{\lambda M} \tag{25}$$

Multiplying by $\lambda k M$, we obtain the following bounds for the coding gain:

$$\frac{k}{n} E(N_0) + \frac{k}{n} B_2(n, s+1) \leq (CG)_s \leq \frac{k}{n} E(N_0) + ks \tag{26}$$

Since $\frac{k}{n} E(N_0) = (CG)_0$, (26) becomes

$$(CG)_0 + \frac{k}{n} B_2(n, s+1) \leq (CG)_s \leq (CG)_0 + ks \tag{27}$$

We shall further discuss these bounds in section 7.

6. Doubly error protection

In section 2, we found the reliability $R(t)$ of a row of chips when a SEC-DED is implemented (equation (5)). This expression considers the general case in which events A, B, C and D occur.

Call $R^{(2)}(t)$ the reliability of a row of chips when a DEC-TED code is implemented. $R(t)$ has a complicated expression but $R^{(2)}(t)$ is even worse, since many patterns causing a triple error have to be considered. So, we shall assume that only events A and B occur, i.e., $c = d = 0$, $a + b = 1$. If most single-cell failures are soft errors and "scrubbing" is implemented, this is not an unrealistic assumption.

Denote by $(MTBF)'$, $(CG)'$ and n' the mean time between failure, coding gain and number of chips per row, respectively, for the memory with a DEC-TED code. An accurate model for the number of failures per row is a Poisson process of rate $\lambda n'$, as it was done in section 2.

A failure of the whole memory will occur if and only if in *any* row of chips one of the following four events occurs:

(i) Two A-failures in position i and a B-failure in position j, for some i, j, $1 \leq i, j \leq l$.

(ii) Two B-failures in position i and an A-failure in position j for some i, j.

(iii) Three A-failures in some position i.

(iv) Three B-failures in some position i.

However, we shall assume that two A-failures in some position i or two B-

failures in some position j are enough to make the whole memory fail. The value of $(MTBF)'$ found under these assumptions is very slightly smaller than the real one when a and b are close, as is the case in general.

Fix a position i for an A-failure. Then, the number of A-failures in this position i is a Poisson process of rate $a\lambda n'/l$, and as, at most, one failure is tolerated per position i, the reliability with respect to A-failures is $[\exp(-a\lambda n't/l)(1 + a\lambda n't/l)]^l$. Similarly for B-failures and, since events A and B are independent, we obtain

$$R^{(2)}(t) = e^{-\lambda n' t}\left[1 + \frac{\lambda n' t}{l} + ab\left(\frac{\lambda n' t}{l}\right)^2\right]^l \tag{28}$$

By (1) and an adequate change of variables,

$$(MTBF)' = \frac{l}{\lambda n'}\int_0^\infty e^{-Mlx}(1 + x + abx^2)^{Ml}\,dx \tag{29}$$

Applying asymptotics (theorem 3.1., chapter I), as $Ml \to \infty$, we obtain

$$(MTBF)' \sim \frac{1}{\lambda n'}\left(\frac{\pi l}{2M(1-2ab)}\right)^{\frac{1}{2}} \tag{30}$$

and hence

$$(CG)' \sim \frac{k}{n'}\left(\frac{\pi Ml}{2(1-2ab)}\right)^{\frac{1}{2}} \tag{31}$$

If $a = b = \frac{1}{2}$ as in most cases,

$$(CG)' \sim \frac{k}{n'}(\pi Ml)^{\frac{1}{2}} \tag{32}$$

Notice that the asymptotic estimates (11), (12) and (13) are valid even for small M, since the estimates are made on Ml which in general is a number large enough.

Now, what is the increment in coding gain with respect to the memory with SEC-DED? Using (16) and taking the quotient, we get (as $M \to \infty$)

$$\frac{(CG)'}{CG} \sim \frac{n}{n'} \left(\frac{2ab}{1-2ab} \right)^{\frac{1}{2}} \sqrt{l} \tag{33}$$

If $a = b = \frac{1}{2}$, this becomes

$$\frac{(CG)'}{CG} \sim \frac{n}{n'} \sqrt{l} \tag{34}$$

Example: $l = 256$, $k = 32$, $n = 39$, $n' = 45$, M a large number and only failures of type A or B occur with the same frequency. Then, the increment in coding gain when DEC-TED is implemented with respect to the memory with SEC-DED, applying (34), is roughly 14.

7. Comparison between the two methods

In this section, we shall give a discussion of the methods described in sections 5 and 6 to increase the lifetime of a computer memory. Of course, we cannot give a conclusive answer about which method is better, since the manufacturer must take into account hardware considerations.

Assume, as usual, that soft errors are "scrubbed," $a = b = \frac{1}{2}$, and M is a sufficiently large number such that we can use asymptotics.

Suppose we implement a DEC-TED code. How many spare rows do we need to add such that $(CG)_s \geq (CG)'$?

From (27), in particular, if

$$(CG)_0 + \frac{k}{n} B_2(n, s+1) \geq (CG)' \tag{35}$$

then $(CG)_s \geq (CG)'$.

Replacing (34) in (35), we have,

$$(CG)_0 + \frac{k}{n} B_2(n, s+1) \geq \frac{n}{n'} \sqrt{l}\, (CG)_0$$

or,

$$\left(\frac{n}{n'} \sqrt{l} - 1 \right) (CG)_0 \leq \frac{k}{n} B_2(n, s+1) \tag{36}$$

By (16), $(CG)_0 \sim \frac{k}{n} \sqrt{\pi} \sqrt{M}$, so (36) becomes, as M is large enough,

$$\left(\frac{n}{n'} \sqrt{l} - 1 \right) \sqrt{\pi} \sqrt{M} \leq B_2(n, s+1) \tag{37}$$

Using our typical example with $k = 32$, $n = 39$, $n' = 45$, and $l = 256$, we obtain

$$(22.8) \sqrt{M} \leq B_2(39, s+1) \tag{38}$$

Using equation (38) and fixing M (big enough so that the asymptotic approximation of $(CG)_0$ makes sense), we can find the minimum s that verifies the inequality. Let us call $s(M)$ this minimum s. If we add $s(M)$ spare rows, we are adding $39\, s(M)$ chips, while if we implement a DEC-TED code, we have to add $6M$ chips. If $39\, s(M) \leq 6M$, or , $s(M) \leq \frac{2}{13} M$, we conclude that adding spare rows is better than implementing a DEC-TED

code. The matter will be settled if we find $B_2(39, s+1)$ for various values of s.

From formula (5) of chapter I, we have

$$B_2(39, s+1) = 39 \int_0^\infty e^{-39x} \left(\sum_{i=0}^{s} \frac{x^i}{i!} \right)^{39} dx + \\ + \frac{(39)^2}{(s+1)!} \int_0^\infty e^{-39x} \left(\sum_{i=0}^{s} \frac{x^i}{i!} \right)^{38} x^{s+1}\, dx \qquad (39)$$

Performing numerical integration in (39), we obtain the following table for $B_2(39, s+1)$:

s	$B_2(39, s+1)$
1	12.8
2	29.0
3	48.3
4	69.6
5	92.4
6	116.3
7	141.2
8	166.8
9	193.1
10	219.9
11	247.2
12	275.0
13	303.1
14	331.6
15	360.4
16	389.5
17	418.9
18	448.5
19	478.4
20	508.4

Using these values, we obtain $s(M)$ for several values of M, and then we can compare $s(M)$ with $\frac{2}{13}M$, as shown in the following table:

M	$(22.8)\sqrt{M}$	$s(M)$	$\frac{2}{13}M$
50	161.2	8	7.7
60	176.6	9	9.2
80	203.9	10	12.3
100	228	11	15.4
150	279.2	13	23.1
200	322.4	14	30.8
400	456	19	61.5

The table shows that $s(M)$ is a function that grows very slowly. For M around 50, the result is inconclusive, but for $M \geq 60$, clearly $s(M) < \frac{2}{13}M$, showing that sparing is better than implementing 2-ECC.

References

[1] M. Blaum and R. J. McEliece, "Single-Error Protected Semiconductor RAM Memories," to appear.

[2] R. M. F. Goodman and R. J. McEliece, "Lifetime Analyses of Error-Control Coded Semiconductor RAM Systems," Proc. IEE, vol. 129, part E, No. 3, May 1982, pp. 81-85.

[3] R. M. F. Goodman and R. J. McEliece, "Hamming Codes, Computer Memories and the Birthday Surprise," Proc. 20th Allerton Conference on Communication, Control and Computing, pp. 672-679.

[4] C. L. Chen and M. Y. Hsiao, "Error-Correcting Codes for Semiconductor Memory Applications: A State-of-the-Art review," IBM J. Res. Develop. 28, 124-134 (March 1984).

[5] M. Y. Hsiao, "A Class of Optimal Minimum Odd-Weight-Column SEC-DED Codes," IBM J. Res. Develop. 14, 395-401 (July 1970).

[6] Intel Corp., "Memory System Reliability with ECC," in Intel Memory Applications Handbook, 1980.

CHAPTER III

A CLASS OF BURST-ERROR CORRECTING CODES

1. Introduction

Figure 1 shows a simple array-code in which the last row and the last column are parity-check bits.

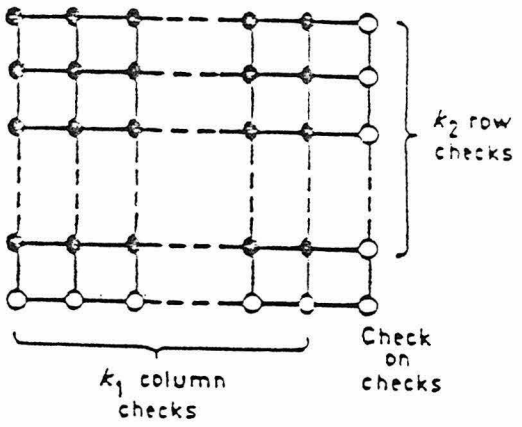

Figure 1: Two-dimensional code with single parity checks

This code has block length $(k_1+1)(k_2+1)$ and rate $k_1k_2/(k_1+1)(k_2+1)$. It is well known that it can correct a single random error.

It was recently shown that burst-error correction is possible if the digits are read diagonally ([1], [2], [3]).

An efficient way of diagonally reading the array is shown in figure 2.

Let us call b the burst-error correcting capability of the code. It has been conjectured ([1]) that, if $k_2 \geq 2(k_1-1)$, then $b = k_1$ (i.e., the code can correct any burst of length smaller or equal than k_1).

Our goal in this chapter is precisely proving this conjecture.

There exist efficient encoding and decoding algorithms for the code ([1],

0	17	14	11	h_0
4	1	18	15	h_1
8	5	2	19	h_2
12	9	6	3	h_3
16	13	10	7	h_4
v_0	v_1	v_2	v_3	

$k_1 = 3$

$k_2 = 4$

16, 13, 10, 11, 15, 19, 3, 7: checks
read-out order: $0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow \cdots \rightarrow 19$

Figure 2: $(20, 12)$ $b = 3$ array-code

[2]). These algorithms are based on the calculation of the horizontal and vertical syndromes.

The horizontal syndrome is represented by a vector $(h_0, h_1, \cdots h_{k_2})$, where h_i is the sum of the received bits in row i. Similarly, the vertical syndrome is represented by vector $(v_0, v_1, \cdots v_{k_1})$, where v_j is the sum of the received bits in column j.

We shall prove that, when the right conditions on k_1 and k_2 are met, then for each burst of length smaller or equal than k_1 corresponds a unique syndrome (vertical and horizontal).

2. Basic properties of the code

We need a precise mathematical description for the diagonal read-out. Consider a codeword $(a_{ij})_{\substack{0 \le i \le k_2 \\ 0 \le j \le k_1}}$. The read-out starts at entry (0,0). The set of pairs of indices (i, j) , $0 \le i \le k_2$, $0 \le j \le k_1$, will be considered as labels of vertices in a directed graph, where i is taken modulo $(k_2 + 1)$.

Every vertex has exactly one outgoing arrow, defined by the following law:

$$\left.\begin{aligned} (i,j) &\to (i+1, j+1) \quad \text{if} \quad j < k_1 \\ (i,k_1) &\to (i-k_1+1, 0) \end{aligned}\right\} \tag{1}$$

The diagonal read-out of the entries $a_{i,j}$ starts at $a_{0,0}$ and proceeds with the next entries in a directed path defined by law (1). For this read-out to make sense, we need the directed graph to be a directed cycle. Let us prove this result.

2.1. Lemma

The directed graph defined by (1) is a directed cycle.

Proof: Consider the set of integers modulo $(k_1+1)(k_2+1)$. Of course, they can be considered as the vertices of a directed cycle under the law $l \to l+1$. It is routinely verified that the following assignment f, from our directed graph to the integers modulo $(k_1+1)(k_2+1)$, is a graph isomorphism:

$$f(i,j) = (i-j)(k_1+1) + j \tag{2}$$

The lemma is proved. ∎

The diagonal read-out (1) and assignment (2) are illustrated in figure 3, with $k_1 = 4$ and $k_2 = 6$.

The directed cycle interpretation of the read-out allows us to associate with a burst of length b, a directed path of length (i.e., number of edges) $b-1$, where the first and last bits of the burst correspond, respectively, to the first and last vertices of the path.

(0,0)	(0,1)	(0,2)	(0,3)	(0,4)
(1,0)	(1,1)	(1,2)	(1,3)	(1,4)
(2,0)	(2,1)	(2,2)	(2,3)	(2,4)
(3,0)	(3,1)	(3,2)	(3,3)	(3,4)
(4,0)	(4,1)	(4,2)	(4,3)	(4,4)
(5,0)	(5,1)	(5,2)	(5,3)	(5,4)
(6,0)	(6,1)	(6,2)	(6,3)	(6,4)

0	31	27	23	19
5	1	32	28	24
10	6	2	33	29
15	11	7	3	34
20	16	12	8	4
25	21	17	13	9
30	26	22	18	14

Figure 3: The (35,24), $b = 4$ code

We also have a distance between two vertices (i.e., length of the shortest path connecting them). From (2),

$$\mathrm{d}((i_1, j_1), (i_2, j_2)) = \min \{ f(i_1, j_1) - f(i_2, j_2) \, , \, -[f(i_1, j_1) - f(i_2, j_2)] \} \qquad (3)$$

Of course, equation (3) is taken modulo $(k_1 + 1)(k_2 + 1)$.

Further properties are then easy to obtain. The next lemma is immediate using assignment (2).

2.2. Lemma

$$f(i+1, j) = f(i, j) + k_1 + 1 \qquad (4)$$

$$f(i, j+1) = f(i, j) - k_1 \quad , \quad 0 \leq j < k_1 \qquad (5)$$

Equation (4) tells us that there is a path of length $k_1 + 1$ from vertex (i, j) to vertex $(i + 1, j)$ (see figure 3). As index i is taken modulo $(k_2 + 1)$, this means, we have a cyclic structure on the rows of the array.

Equation (5) shows that there are bursts of length $k_1 + 1$ that are uncorrectable. In effect, the burst with 1's in entries (i, j) and $(i, j + 1)$, 0 in all the other entries, has the same syndrome that a burst with 1's in entries (l, j) and $(l, j + 1)$, 0 in all other entries, where $i \neq l$.

From now on, let $k_2 \geq k_1$. Consider a path

$$(i, j) \rightarrow (i + 1, j + 1) \rightarrow (i + 2, j + 2) \rightarrow \cdots$$

Whenever $j + t < k_1$, the path visits then the next row. However, when $j + t = k_1$, from (1), we have

$$(i + t, j + t) = (i + k_1 - j, k_1) \rightarrow (i - j + 1, 0)$$

and we shall say that rows

$$i + k_1 - j + 1, \quad i + k_1 - j + 2, \quad \cdots, i - j \mod(k_2 + 1)$$

are *skipped* by the path.

2.3 Lemma

Consider a path of length at most $k_1 - 1$ in the cycle defined by (1), with $k_2 \geq k_1$. Then any row and column are visited at most once by the path. If a row is skipped, then it will not be visited at all.

Proof: In each step we move cyclically to the right. Hence, a path of length at most $k_1 - 1$ will visit at most k_1 different columns. Since there are exactly $k_1 + 1$ columns, no column can be visited more than once.

Consider now rows. Without loss, assume the path has length $b = k_1 - 1$. Let (i, j) be the initial vertex. Given the cyclic structure on rows, also without loss, we may take $i = 0$.

Assume $j = 0$ or $j = 1$. After $k_1 - 1$ steps, the final vertex is $(k_1 - 1, k_1 - 1)$ or $(k_1 - 1, k_1)$. Since $k_1 \leq k_2$, the path visits exactly k_1 consecutive rows.

Assume $j \geq 2$. The path is (using (1))

$$(0, j) \rightarrow (1, j + 1) \rightarrow \cdots \rightarrow (k_1 - j + 2, 0) \rightarrow \cdots \rightarrow (k_2, j - 2)$$

Since $k_1 \leq k_2$, no row is visited more than once. Rows

$$k_1 - j + 1\,,\ k_1 - j + 2\,,\ \cdots\,,\ k_2 - j + 1$$

are skipped and the path never visits them. ∎

We can now prove our main result relating the burst-error capability of the code with conditions on k_1 and k_2. We shall do that in the next section.

3. The Main Result

3.1. Theorem

The code defined by the diagonal read-out (1) can correct any burst of length at most k_1 if and only if

$$k_2 \geq 2(k_1 - 1) \tag{6}$$

Proof: $k_1 = 1$ is a trivial case, so we are going to assume $k_1 \geq 2$.

$\Rightarrow$) Assume the burst of length k_1 whose associated path is

$$(0,0) \rightarrow (1,1) \rightarrow \cdots \rightarrow (k_1 - 1, k_1 - 1)$$

occurs, where entries (0,0) and $(k_1 - 1, k_1 - 1)$ correspond to 1's and the rest of the entries are 0's (figure 4).

	0	1	...	$k_1 - 1$	k_1
0	1	0	...	0	0
1	0	0	...	0	0
⋮	⋮	⋮	⋱	⋮	⋮
$k_1 - 1$	0	0	...	1	0
k_1	0	0	...	0	0
⋮	⋮	⋮	⋱	⋮	⋮
k_2	0	0	...	0	0

Figure 4

There is exactly one other vector having the same syndrome: the vector with 1's in entries $(0, k_1 - 1)$ and $(k_1 - 1, 0)$, 0 elsewhere. Since the code can correct any burst of length k_1,

$$\mathrm{d}((0, k_1 - 1), (k_1 - 1, 0)) \geq k_1 \tag{7}$$

From (3), in particular,

$$f(0, k_1 - 1) - f(k_1 - 1, 0) \geq k_1 \tag{8}$$

Applying (2) and taking into account that inequality (8) is taken modulo $(k_1 + 1)(k_2 + 1)$, we have

$$-2k_1^2 + k_1 + 1 + (k_1 + 1)(k_2 + 1) \geq k_1 \tag{9}$$

After some easy manipulations, (9) becomes

$$k_2 \geq 2(k_1 - 1) - \frac{k_1}{k_1 + 1} \tag{10}$$

Since all the involved numbers are integers, (6) follows from (10).

$\Leftarrow$) Assume a burst $\vec{a}$ of length $b \leq k_1$ occurs. We shall show that no other burst of length smaller or equal than k_1 has the same syndrome. Distinguish two cases:

(i) The path of length $b - 1$ associated with the burst does *not* skip rows.

(ii) The path associated with the burst does skip rows.

Assume there is another burst $\vec{b}$ of length at most k_1 with the same syndrome that $\vec{a}$ and $\vec{a} \neq \vec{b}$. By considering cases (i) and (ii) separately, we shall show that each leads to a contradiction.

Case (i)

The path of length $b - 1$ associated with the burst has the form

$$(i, j) \rightarrow (i + 1, j + 1) \rightarrow \cdots \rightarrow (i + b - 1, j + b - 1)$$

where $j + b - 1 \leq k_1$. Without loss, $i = 0$. Thus, the burst is a vector

$$\vec{a} = (0, \cdots, 0, a_{0,j}, a_{1,j+1}, \cdots, a_{b-1,j+b-1}, \cdots, 0, \cdots, 0)$$

where $a_{0,j} = a_{b-1,j+b-1} = 1$. The case is illustrated in figure 5.

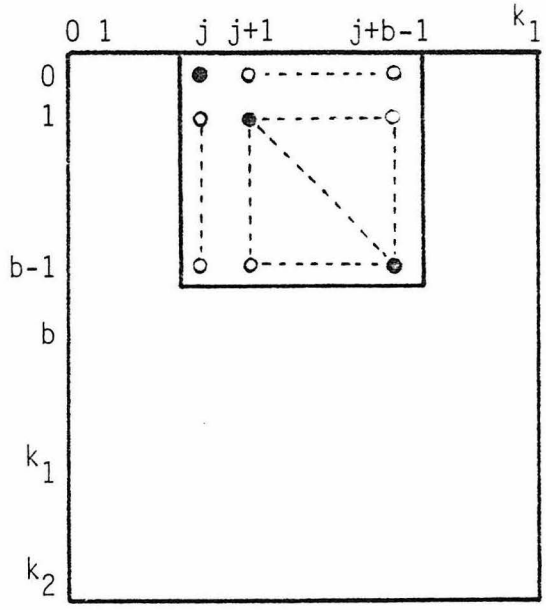

Figure 5: Case (i): The path (black points) does not skip rows

Let

$$\vec{b} = (0 \cdots 0, b_{\alpha,j+\beta}, b_{\alpha+1,j+\beta+1}, \cdots)$$

such that $b_{\alpha,j+\beta} = 1$ and $(\alpha, j+\beta)$ is the initial vertex in the associated path. As we are assuming that $\vec{a}$ and $\vec{b}$ have the same syndrome and $\vec{a} \neq \vec{b}$, the following conditions hold:

$$\left.\begin{array}{c} 0 \leq \alpha \leq b-1 \quad , \quad 0 \leq \beta \leq b-1 \quad , \quad (\alpha,\beta) \neq (0,0) \\ b_{s,t} = 0 \quad \text{for} \quad s > b-1 \quad , \quad 0 \leq t < j \quad \text{or} \quad j+b \leq t \leq k_1 \end{array}\right\} \qquad (11)$$

Consider the path associated with $\vec{b}$, say,

$$(\alpha, j+\beta) \rightarrow (\alpha+1, j+\beta+1) \rightarrow \cdots$$

This path has length at most $k_1 - 1$.

There are three possibilities for α and β, each one leading to contradiction.

(a) $\alpha = \beta$: Assume $j = 0$, then $(0,0)$ is the initial vertex in the path associated with $\vec{a}$ and (α, α) is the initial vertex in the path associated with $\vec{b}$. Since $(0,0) \neq (\alpha, \alpha)$, $\alpha \geq 1$. Then the path associated with β has the form

$$(\alpha, \alpha) \rightarrow (\alpha+1, \alpha+1) \rightarrow \cdots \rightarrow (k_1, k_1) \rightarrow (1,0) \rightarrow \cdots$$

Notice that row 0 is skipped, then, by lemma 3.3., it will never be visited. Hence, $b_{0,t} = 0$ for all t, but $a_{0,0} = 1$, thus, $\vec{a}$ and $\vec{b}$ cannot have the same syndrome.

Let $j > 0$, then, the path is

$$(\alpha, \alpha+j) \rightarrow (\alpha+1, \alpha+j+1) \rightarrow \cdots \rightarrow (k_1 - j, k_1) \rightarrow$$
$$\rightarrow (k_2 - j + 2, 0) \rightarrow \cdots \rightarrow (0, j-1)$$

Since $a_{0,j} = 1$ and $\vec{a}$ and $\vec{b}$ have the same syndrome, then $b_{0,j-1} = 1$. But this contradicts (11).

(b) $\alpha > \beta$: The path associated with $\vec{b}$ is

$$(\alpha, \beta + j) \to (\alpha + 1, j + \beta + 1) \to \cdots \to (\alpha - \beta + b - 1, j + b - 1) \to \cdots$$

Notice that $b - 1 < \alpha - \beta + b - 1 \leq 2(b-1) \leq 2(k_1 - 1) \leq k_2$, by (6). Since $a_{b-1,j+b-1} = 1$, then $b_{\alpha-\beta+b-1,j+b-1} = 1$, but this contradicts (11).

(c) $\alpha < \beta$: Distinguish two cases:

$$\beta - \alpha \leq k_1 - (j + b - 1) \quad \text{and} \quad \beta - \alpha > k_1 - (j + b - 1).$$

Assume $\beta - \alpha \leq k_1 - (j + b - 1)$. The path is then

$$(\alpha, \beta + j) \to (\alpha + 1, \beta + j + 1) \to \cdots \to (\alpha - \beta + b - 1, j + b - 1) \to$$
$$\to (\alpha - \beta + b, j + b) \to \cdots \to (b - 1, j + b - 1 + \beta - \alpha) \to \cdots$$

Our conditions imply $j + b - 1 < j + b - 1 + \beta - \alpha \leq k_1$. Since $a_{b-1,j+b-1} = 1$ then $b_{b-1,j+b-1+\beta-\alpha} = 1$. But this contradicts (11).

So, assume $\beta - \alpha > k_1 - (j + b - 1)$. Therefore, the path is

$$(\alpha, \beta + j) \to (\alpha + 1, \beta + j + 1) \to \cdots \to (\alpha - \beta + b - 1, j + b - 1) \to$$
$$\to (\alpha - \beta + b, j + b) \to \cdots \to (\alpha - \beta + k_1 - j, k_1) \to (k_2 + \alpha - \beta - j + 2, 0) \to \cdots$$

Our condition implies $\alpha - \beta + k_1 - j < b - 1$. Observe that $j + \beta - \alpha \leq k_1$. Thus, from (6),

$$k_2 \geq 2(k_1 - 1) \geq (b-1) + (k_1 - 1) > (b-1) + (k_1 - 2)$$
$$\geq (b-1) + (j + \beta - \alpha - 2)$$

so, $k_2 + \alpha - \beta - j + 2 > b - 1$.

This means, row $b-1$ is skipped by the path. This is a contradiction, since $a_{b-1,j+b-1} = 1$.

From (a), (b) and (c), the result is true for bursts whose associated path does not skip rows.

Case (ii)

Now the path associated with $\vec{a}$ has the form

$$(i,j) \to (i+1, j+1) \to \cdots \to (i + k_1 - j, k_1) \to$$
$$\to (i - j + 1, 0) \to \cdots \to (i - j + t + 1, t)$$

where $j - t \geq 2$. Without loss, $i - j + 1 = 0$ (see figure 6).

So,

$$\vec{a} = (0, \cdots, 0, a_{j-1,j}, \cdots, a_{t,t}, 0, \cdots, 0)$$

where $a_{j-1,j} = a_{t,t} = 1$; the associated path has endpoints $(j-1, j)$ and (t,t) and rows $k_1, k_1 + 1, \cdots k_2$ are skipped.

Let (α, β) be the initial point of the path associated with $\vec{b}$. Since $\vec{a}$ and $\vec{b}$ have the same syndrome, the following conditions hold:

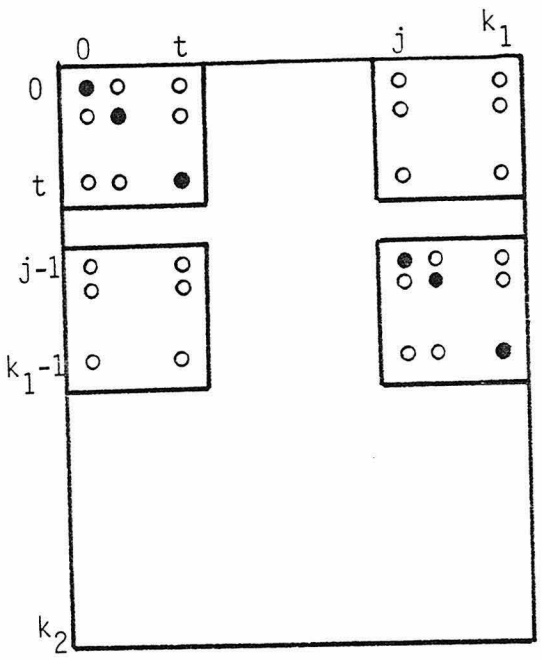

Figure 6:

The path with endpoints $(j-1, j)$ and (t, t) skips rows $k_1, k_1+1, \cdots k_2$

$$\left.\begin{array}{c} 0 \le \alpha \le t \quad \text{or} \quad j-1 \le \alpha \le k_1 - 1 \\ 0 \le \beta \le t \quad \text{or} \quad j \le \beta \le k \\ (\alpha, \beta) \ne (j-1, j) \quad , \quad b_{\alpha,\beta} = 1 \\ b_{u,v} = 0 \quad \text{for} \quad t < u < j-1 \quad , \quad k_1 \le u \le k_2 \quad \text{or} \quad t < v < j \end{array}\right\} \qquad (12)$$

As in case (i), we distinguish three subcases:

(a) $\beta = \alpha + 1$: Assume first $0 \le \alpha \le t$. Then the path associated with $\vec{b}$ is

$$(\alpha, \alpha+1) \to (\alpha+1, \alpha+2) \to \cdots \to (t, t+1) \to \cdots$$

Since $a_{t,t} = 1$, then $b_{t,t+1} = 1$, but this contradicts (12).

So, assume $\alpha \ge j$. Then, the path is

$$(\alpha, \alpha+1) \to (\alpha+1, \alpha+2) \to \cdots \to (k_1-1, k_1) \to$$

$$\to (0,0) \to (1,1) \to \cdots \to (j-1, j-1) \to \cdots$$

Since $a_{j-1,j} = 1$, then $b_{j-1,j-1} = 1$, again contradicting (12).

(b) $\beta < \alpha + 1$: Since $a_{j-1,j} = 1$, the path associated with $\vec{b}$ visits column j at a point (l, j) , $b_{l,j} = 1$ and $l = \alpha - \beta + j$ or $l = \alpha - \beta + j + 1$.

From (6), $j + 1 \leq l \leq k_2$. If $l \geq k_1$, we would contradict (12), hence $j \leq l \leq k_1 - 1$.

Define a finite subset T of nonnegative integers as follows:

$$T = \{i: \quad l + i(l-j+1) < k_1 \quad \text{and} \quad b_{l+i(l-j+1),l+(i-1)(l-j+1)+1} = 1\}$$

Since $b_{l,j} = 1$, $0 \in T$, hence, T is not empty.

Let $i_0 = \max \quad T$, then, as in particular, $i_0 \in T$,

$$b_{l+i_0(l-j+1),l+(i_0-1)(l-j+1)+1} = 1.$$

Since $\vec{a}$ and $\vec{b}$ have the same syndrome,

$$a_{l+i_0(l-j+1),l+i_0(l-j+1)+1} = 1,$$

and by the same token,

$$b_{l+(i_0+1)(l-j+1),l+i_0(l-j+1)+1} = 1 \tag{13}$$

By the maximality of i_0 , $l + (i_0 + 1)(l - j + 1) \geq k_1$.

Also, since $l + i_0(l - j + 1) \leq k_1 - 1$, then

$$l + (i_0 + 1)(l - j + 1) \leq k_1 - 1 + (l - j + 1) \leq 2(k_1 - 1) \leq k_2$$

by (6). Thus, (13) contradicts (12).

(c) $\beta > \alpha + 1$: We claim, the path associated with $\vec{b}$ visits the rectangle of entries (u, v) where $j - 1 \leq u \leq k_1 - 1$ and $j \leq v \leq k_1$ (in figure 6, this corresponds to the lower right rectangle). If $j - 1 \leq \alpha$ then the initial point (α, β) is in the rectangle, so assume $\alpha < j - 1$. Distinguish two cases:

$$\beta - \alpha > k_1 - j + 1 \quad \text{and} \quad \beta - \alpha \leq k_1 - j + 1.$$

If $\beta - \alpha > k_1 - j + 1$ the path associated with $\vec{b}$ is

$$(\alpha, \beta) \to (\alpha + 1, \beta + 1) \to \cdots \to (\alpha - \beta + k_1, k_1) \to (\alpha - \beta + k_2 + 2, 0) \to \cdots$$

Since $\beta - \alpha - 2 < k_1 - 1$ and $j - 1 \leq k_1 - 1$ then

$$(\beta - \alpha - 2) + (j - 1) < 2(k_1 - 1) \leq k_2$$

by (6). Therefore, since $\alpha - \beta + k_1 < j - 1$ and $\alpha - \beta + k_2 + 2 > j - 1$ row $j - 1$ is skipped. This is a contradiction because $a_{j-1,j} = 1$.

If $\beta - \alpha \leq k_1 - j + 1$ the path is

$$(\alpha,\beta) \to (\alpha+1,\beta+1) \to \cdots \to (j-1,\beta-\alpha+j-1) \to \cdots$$

and point $(j-1,\beta-\alpha+j-1)$ lies in the rectangle. So, the claim is true, and without loss, we may assume that the initial point (α,β) lies in the rectangle, i.e.,

$$b_{\alpha,\beta} = 1 \quad , \quad j-1 \le \alpha \le k_1 - 1 \quad , \quad j \le \beta \le k_1.$$

Define a finite subset W of nonnegative integers as follows:

$$W = \{i: \quad \beta + i(\beta-\alpha-1) \le k_1 \quad \text{and} \quad b_{\beta+(i-1)(\beta-\alpha-1)-1,\beta+i(\beta-\alpha-1)} = 1\}$$

Notice that W is nonempty since $0 \in W$. Let $i_0 = \max \quad W$. In particular, $b_{\beta+(i_0-1)(\beta-\alpha-1)-1,\beta+i_0(\beta-\alpha-1)} = 1$. Since $\vec{a}$ and $\vec{b}$ have the same syndrome, then $a_{\beta+i_0(\beta-\alpha-1)-1,\beta+i_0(\beta-\alpha-1)} = 1$.

If $\beta + (i_0+1)(\beta-\alpha-1) \le k_1$ then $b_{\beta+i_0(\beta-\alpha-1)-1,\beta+(i_0+1)(\beta-\alpha-1)} = 1$ contradicting the maximality of i_0. Therefore, $\beta + (i_0+1)(\beta-\alpha-1) > k_1$. So, the path associated with $\vec{b}$ after vertex

$$\Big(\beta + (i_0-1)(\beta-\alpha-1) - 1\,,\, \beta + i_0(\beta-\alpha-1)\Big)$$

is

$$\cdots \to \Big(\beta + (i_0-1)(\beta-\alpha-1) - 1\,,\, \beta + i_0(\beta-\alpha-1)\Big) \to$$
$$\to \Big(\beta + (i_0-1)(\beta-\alpha-1)\,,\, \beta + i_0(\beta-\alpha-1) + 1\Big) \to$$
$$\cdots \to (\alpha-\beta+k_1\,,\, k_1) \to (\alpha-\beta+k_2+2\,,\, 0) \to \cdots$$

As before, by (6),

$$\alpha - \beta + k_2 + 2 > k_1 - 1 \geq \beta + i_0(\beta - \alpha - 1) - 1,$$

and since $k_1 < \beta + (i_0 + 1)(\beta - \alpha - 1)$ we have,

$$\alpha - \beta + k_1 < \beta + i_0(\beta - \alpha - 1) - 1.$$

This means that row $\beta + i_0(\beta - \alpha - 1) - 1$ is skipped by the path, a contradiction since $a_{\beta+i_0(\beta-\alpha-1)-1,\beta+i_0(\beta-\alpha-1)} = 1$.

Subcases (a), (b) and (c) show that any burst $\vec{a}$ of type (ii) has a unique syndrome. This completes the proof. ∎

References

[1] P. G. Farrell and S. J. Hopkins, "Burst-error-correcting Array Codes," The Radio and Electronic Engineer, Vol. 52, No 4, April 1982, pp. 182-192.

[2] P. G. Farrell and S. J. Hopkins, "Decoding Algorithms for a Class of Burst-error-correcting Array Codes," IEEE Int. Symp. on Information Theory, Les Arcs, France, June 21-25, 1982.

[3] J. S. Daniel, "Array Codes for Error Control," M. Sc. Thesis, University of Manchester, 1983.

CHAPTER IV

A CLASS OF ERROR-CORRECTING CODES FOR MAGNETIC TAPES

1. Introduction

Patel and Hong ([2],[4]) devised an error-correcting scheme that was successfully used in the IBM 3420 series tape units with a recording density of 6250 bits per inch. This error-correcting scheme is capable of correcting any error pattern on a single track or any error patterns on two tracks provided that the erroneous tracks i and j are identified by some external pointers (that is, two track erasures).

Here, we shall describe in detail a family of codes that can correct higher numbers of track errors and erasures and contains previously known codes as particular cases ([1],[4]). These codes are *maximum distance separable* or MDS ([3]).

An IBM 3420 series tape unit writes characters in parallel across 9 tracks on a $\frac{1}{2}$-inch tape as shown in figure 1.

Each character consists of 8 information bits and one overall parity-check bit. The rows and the first 8 bits in each column will be considered as elements of the Galois field of order 2^8 , $GF(2^8)$.

Although $GF(2^8)$ can be defined using any irreducible polynomial of degree 8 over $GF(2)$, Patel and Hong used $g(x) = 1 + x^3 + x^4 + x^5 + x^8$ which is the irreducible polynomial of degree 8 with minimum exponent. This choice of $g(x)$ simplifies the decoding (see [2]).

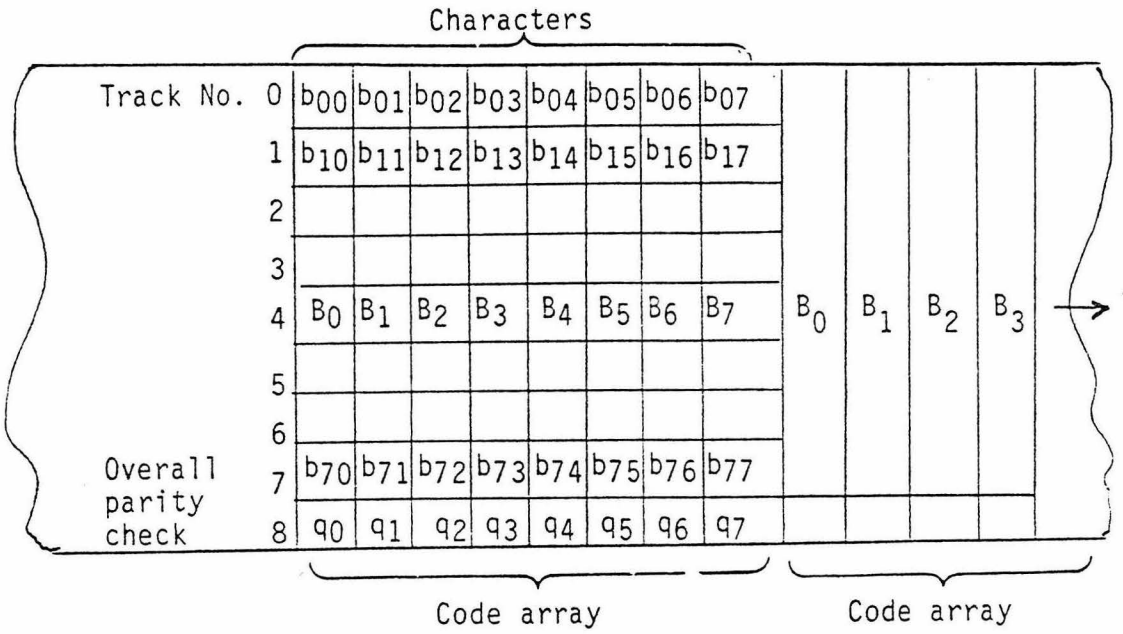

Figure 1

The construction can be generalized to an $(n+1) \times n$ array. Consider an $(n+1)$-track tape. Let $GF(2^n)$ be defined by $g(x)$, an irreducible polynomial of degree n over $GF(2)$. Denote by B_i the first n bits in each column, $0 \leq i \leq n-1$, and by Z_j each row, $0 \leq j \leq n$. Z_n is a parity-check row (also denoted Q in literature). B_i and Z_j are considered as elements in $GF(2^n)$.

The type of errors that occur are track-errors. In other words, since rows are elements in $GF(2^n)$ these are byte-errors. There are efficient byte-error-correcting codes, like Reed-Solomon codes. However, they do not work well for magnetic tapes, since the input and the output are read vertically. So, we need a procedure that permutes the action of rows and columns. This will be achieved by the family of codes $B(n, m)$ to be described in the next

section.

2. Construction and basic properties of B(n,m)-codes

Consider an $(n+1) \times n$ array $(b_{i,j})_{\substack{0 \le i \le n \\ 0 \le j \le n-1}}$, $b_{i,j} \in GF(2)$. Let $GF(2^n)$ be defined by $g(x)$, where $g(x)$ is an irreducible polynomial of degree n over $GF(2)$. Let α be a root of $g(x)$, $\alpha \in GF(2^n)$, i.e., $g(\alpha) = 0$.

As stated in the introduction, we consider the rows and the first n bits in each column as elements of $GF(2^n)$. Therefore,

$$\left.\begin{aligned} Z_i &= \sum_{k=0}^{n-1} b_{ik}\alpha^k \quad , \quad 0 \le i \le n \\ B_j &= \sum_{k=0}^{n-1} b_{kj}\alpha^k \quad , \quad 0 \le j \le n-1 \end{aligned}\right\} \tag{1}$$

Let $0 \le m \le n-1$. A $B(n,m)$-code is the set of vectors $(Z_0, Z_1, \cdots, Z_{n-1})$ satisfying the following equations in $GF(2^n)$:

$$\sum_{j=0}^{n-1} \alpha^j Z_j^{2^i} = 0 \quad , \quad 0 \le i \le m-1 \tag{2}$$

We see immediately that the m equations (2) define a linear code of length n over $GF(2^n)$. We want to prove that the code is an MDS-code and its minimum distance is $m+1$. We need first a technical lemma.

2.1. Lemma

Let α_0 , α_1 , $\cdots$, α_{m-1} be elements in $GF(2^n)$ and let

$$D(\alpha_0, \alpha_1, \cdots, \alpha_{m-1}) = \det \begin{pmatrix} \alpha_0 & \alpha_1 & \alpha_2 & \cdots & \alpha_{m-1} \\ \alpha_0^2 & \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_{m-1}^2 \\ \alpha_0^4 & \alpha_1^4 & \alpha_2^4 & \cdots & \alpha_{m-1}^4 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha_0^{2^{m-1}} & \alpha_1^{2^{m-1}} & \alpha_2^{2^{m-1}} & \cdots & \alpha_{m-1}^{2^{m-1}} \end{pmatrix}$$

Then

$$D(\alpha_0, \alpha_1, \cdots, \alpha_{m-1}) = \prod_{\{i_1, i_2, \cdots, i_k\} \subseteq \{0,1,\cdots,m-1\}} (\alpha_{i_1} + \alpha_{i_2} + \cdots + \alpha_{i_k})$$

i.e., $D(\alpha_0, \alpha_1, \cdots, \alpha_{m-1})$ *is the product of the sum of the elements of all possible subsets of* $\{\alpha_0, \alpha_1, \cdots, \alpha_{m-1}\}$.

Proof: Induction on m. If $m = 0$ the result is trivial. Assume it is true for $m \geq 1$. Consider the polynomial $f(x) = D(x, \alpha_1, \alpha_2, \cdots, \alpha_{m-1})$ of degree 2^{m-1}. Since our field has characteristic 2, all possible sums

$$\alpha_{i_1} + \alpha_{i_2} + \cdots + \alpha_{i_k} \quad , \quad \{i_1, i_2, \cdots, i_k\} \subseteq \{1, 2, \cdots m-1\}$$

are roots of $f(x)$. Since there are exactly $2^{m-1} - 1$ nonempty subsets of $\{\alpha_1, \alpha_2, \cdots \alpha_{m-1}\}$ and 0 is also a root, $f(x)$ admits a factorization

$$f(x) = C\, x \prod_{\{i_1, i_2, \cdots, i_k\} \subseteq \{1,2,\cdots,m-1\}} (x + \alpha_{i_1} + \alpha_{i_2} + \cdots + \alpha_{i_k}) \qquad (3)$$

where C is the coefficient of $x^{2^{m-1}}$, i.e.,

$$C = D(\alpha_1, \alpha_2, \cdots, \alpha_{m-1}).$$

By induction,

$$C = \prod_{\{i_1,i_2,\cdots i_k\}\subseteq\{1,2,\cdots,m-1\}} (\alpha_{i_1} + \alpha_{i_2} + \cdots + \alpha_{i_k})$$

so, replacing in (3) and making $x = \alpha_0$, the result follows. ∎

We can now prove our main result.

2.2. Theorem

A $B(n,m)$-code is a linear $(n, n-m, d = m+1)$ code. In particular, it is an MDS-code.

Proof: Taking the 2^i-th root in equation i of (2), observe that the parity check matrix of the code is given by $(\lambda_{ij})_{\substack{0\le i\le m-1\\ 0\le j\le n-1}}$ where $(\lambda_{ij})^{2^i} = \alpha^j$. Thus, the code will be MDS and will have dimension $n-m$ if and only if any m columns are linearly independent ([3], chapter 11), i.e.,

$$\det\left((\lambda_{ij_k})_{\substack{0\le i\le m-1\\ 0\le k\le m-1}}\right) \neq 0 \quad \text{for} \quad \text{any} \quad 0 \le j_0 < j_1 < \cdots < j_{m-1} \le n-1.$$

But

$$(\lambda_{ij_k})^{2^{m-1}} = \left[(\lambda_{ij_k})^{2^i}\right]^{2^{m-1-i}} = [\alpha^{j_k}]^{2^{m-1-i}} \quad \text{for} \quad \text{any} \quad 0 \le i \le m-1,$$

hence, it is equivalent to prove that $D(\alpha^{j_0}, \alpha^{j_1}, \cdots, \alpha^{j_{m-1}}) \neq 0$, where

$$D(\alpha^{j_0}, \alpha^{j_1}, \cdots, \alpha^{j_{m-1}}) = \det\begin{pmatrix} \alpha^{j_0} & \alpha^{j_1} & \cdots & \alpha^{j_{m-1}} \\ (\alpha^{j_0})^2 & (a^{j_1})^2 & \cdots & (\alpha^{j_{m-1}})^2 \\ \vdots & \vdots & \ddots & \vdots \\ (\alpha^{j_0})^{2^{m-1}} & (\alpha^{j_1})^{2^{m-1}} & \cdots & (\alpha^{j_{m-1}})^{2^{m-1}} \end{pmatrix}.$$

But the result follows from lemma 2.1., since $D(\alpha^{j_0}, \alpha^{j_1}, \cdots, \alpha^{j_{m-1}})$ is a product of polynomials in α of degree smaller than n. ∎

From $B(n,m)$-codes, we can obtain extended $B(n,m)$-codes denoted $\hat{B}(n,m)$, by adding a parity check byte $Z_n = \sum_{j=0}^{n-1} Z_j$, where $(Z_0, Z_1, \cdots, Z_{n-1})$ is in $B(n,m)$.

A $\hat{B}(n,m)$-code is still MDS. We prove this fact in the following theorem.

2.3. Theorem

A $\hat{B}(n,m)$-code is a linear $(n+1, n-m, d=m+2)$ code. In particular, it is an MDS-code.

Proof: By theorem 2.2., the minimum distance of a $\hat{B}(n,m)$-code is at least $m+1$. This minimum distance will be exactly $m+1$ if and only if there is a codeword of weight $m+1$ in $B(n,m)$, say $(Z_0, Z_1, \cdots, Z_{n-1})$, and $\sum_{j=0}^{n-1} Z_j = 0$. So, assume this is the case and let

$$Z_{j_0}, Z_{j_1}, Z_{j_2}, \cdots, Z_{j_m} \quad , \quad 0 \le j_0 < j_1 < \cdots < j_m \le n-1$$

be the nonzero entries. Then, from (2), these entries satisfy the following set of equations:

$$\left.\begin{aligned} \sum_{k=0}^{m} Z_{j_k} &= 0 \\ \sum_{k=0}^{m} \alpha^{j_k} (Z_{j_k})^{2^i} &= 0 \quad , \quad 0 \le i \le m-1 \end{aligned}\right\} \tag{4}$$

Replacing $Z_{j_m} = \sum_{k=0}^{m-1} Z_{j_k}$ in the last m equations, we obtain:

$$\sum_{k=0}^{m-1}(\alpha^{j_k}+\alpha^{j_m})(Z_{j_k})^{2^i}=0 \quad , \quad 0\le i\le m-1 \tag{5}$$

Taking the ith equation to the power 2^{m-1-i}, the system becomes

$$\sum_{k=0}^{m-1}(\alpha^{j_k}+\alpha^{j_m})^{2^i}(Z_{j_k})^{2^{m-1}}=0 \quad , \quad 0\le i\le m-1 \tag{6}$$

Let $\alpha_i=\alpha^{j_i}+\alpha^{j_m} \quad , \quad 0\le i\le m-1$. Then, system (6) admits a nontrivial solution if and only if $D(\alpha_0,\alpha_1,\cdots,\alpha_{m-1})=0$. But lemma 2.1. assures that $D(\alpha_0,\alpha_1,\cdots,\alpha_{m-1})$ is a product of polynomials in α of degree smaller than n. Hence $D(\alpha_0,\alpha_1,\cdots,\alpha_{m-1})\neq 0$ and (5) is not satisfied. Therefore, the minimum distance is $m+2$. ∎

Theorem 2.3. assures that, whenever $2s+t\le m+1$ a $\hat{B}(n,m)$-code can correct s track errors together with t track erasures. In the next section, we discuss encoding and decoding procedures.

3. Encoding and Decoding

Our code is an array $(b_{ij})_{\substack{0\le i\le n\\ 0\le j\le n-1}}$, where the rows Z_i, $0\le i\le n-1$ satisfy (2) and Z_n is a parity-check row. The input and output are read vertically, an $(n+1)$-bit column at a time. However, the errors occur in the horizontal tracks. Hence, we need a procedure to permute the action of rows and columns. The key property is given by the following lemma:

3.1. Lemma

$$\sum_{j=0}^{n-1} \alpha^j (Z_j)^{2^i} = \sum_{j=0}^{n-1} (\alpha^j)^{2^i} B_j \quad , \quad 0 \leq i \leq m-1 \tag{7}$$

Proof:

$$\begin{aligned}
\sum_{j=0}^{n-1} \alpha^j (Z_j)^{2^i} &= \sum_{j=0}^{n-1} \alpha^j \left(\sum_{k=0}^{n-1} b_{jk} \alpha^k \right)^{2^i} \\
&= \sum_{j=0}^{n-1} \alpha^j \sum_{k=0}^{n-1} b_{jk} (\alpha^k)^{2^i} \\
&= \sum_{k=0}^{n-1} (\alpha^k)^{2^i} \sum_{j=0}^{n-1} b_{jk} \alpha^j \\
&= \sum_{k=0}^{n-1} (\alpha^k)^{2^i} B_k \qquad \blacksquare
\end{aligned}$$

If we consider a $\hat{B}(n,m)$-code as an array code $(b_{ij})_{\substack{0 \leq i \leq n \\ 0 \leq j \leq n-1}}$ over $GF(2)$, by lemma 3.1., an equivalent definition is

$$\left.\begin{aligned}
\sum_{j=0}^{n} Z_j &= 0 \\
\sum_{j=0}^{n-1} (\alpha^j)^{2^i} B_j &= 0 \quad , \quad 0 \leq i \leq m-1
\end{aligned}\right\} \tag{8}$$

Let the parity-check bits be contained in columns $B_0, B_1, \cdots, B_{m-1}$ and in the parity-check row Z_n. Thus, the information symbols are contained in $B_m, B_{m+1}, \cdots, B_{n-1}$. Notice that a $\hat{B}(n,m)$-code has rate $n - m/n + 1$.

The encoding proceeds as follows: first B_{n-1} is received, then B_{n-2}, etc., up to B_m. For each B_j, $b_{nj} = \sum_{k=0}^{n-1} b_{kj}$ is immediately obtained. From (8), we have

$$\sum_{j=0}^{m-1} (\alpha^j)^{2^i} B_j = \sum_{j=m}^{n-1} (\alpha^j)^{2^i} B_j \quad , \quad 0 \leq i \leq m-1 \tag{9}$$

Circuits to obtain $\sum_{j=m}^{n-1}(\alpha^j)^{2^i}B_j$ are easy to implement ([2]). We finally need to solve the linear system (9) in order to obtain $B_0, B_1, \cdots, B_{m-1}$. Once the encoding is completed, the bytes $Z_0, Z_1, Z_2, \cdots, Z_n$ are sent. However, assume bytes $\hat{Z}_0, \hat{Z}_1, \cdots, \hat{Z}_n$ are received. We have to retrieve the original information. The syndrome $(S_p, S_0, S_1, \cdots, S_{m-1})$ is given by

$$\left.\begin{aligned} S_p &= \sum_{i=0}^{n} \hat{Z}_i \\ S_i &= \sum_{j=0}^{n-1} \alpha^j (\hat{Z}_j)^{2^i} \quad , \quad 0 \le i \le m-1 \end{aligned}\right\} \tag{10}$$

However, (10) is an inefficient way to calculate the syndrome, since the information is read vertically. As in the case of the encoding, using lemma 3.1., we obtain

$$\left.\begin{aligned} S_p &= \sum_{i=0}^{n} \hat{Z}_i \\ S_i &= \sum_{j=0}^{n-1} (\alpha^j)^{2^i} \hat{B}_j \quad , \quad 0 \le i \le m-1 \end{aligned}\right\} \tag{11}$$

S_p is easily obtained, one bit at a time, while circuits that find S_i for $0 \le i \le m-1$ are also implemented without difficulties ([2]). If the syndrome is the zero vector, we conclude that the codeword has been transmitted without errors.

Now, suppose s errors occur – say $e_{i_1}, e_{i_2}, \cdots, e_{i_s}$ at locations $i_1, i_2, \cdots, i_s$– together with t erasures –say $e_{j_1}, e_{j_2}, \cdots, e_{j_t}$ at locations $j_1, j_2, \cdots, j_t$– where $2s + t \le m + 1$. Hence, $\hat{Z}_{i_k} = Z_{i_k} + e_{i_k}$ for $1 \le k \le s$, $\hat{Z}_{j_k} = Z_{j_k} + e_{j_k}$ for $1 \le k \le t$, and $\hat{Z}_i = Z_i$ in all other locations. System (10) then becomes

$$\left.\begin{aligned} S_p &= \sum_{k=1}^{s} e_{i_k} + \sum_{k=1}^{t} e_{j_k} \\ S_i &= \sum_{k=1}^{s} \alpha^{i_k}(e_{i_k})^{2^l} + \sum_{k=1}^{t} \alpha^{j_k}(e_{j_k})^{2^l} \quad , \quad 0 \le l \le m-1 \end{aligned}\right\} \tag{12}$$

System (12) is a system of $m+1$ equations with $2s+t$ unknowns, $2s+t \le m+1$. We are assuming that no error or erasure occurs in track n. If it does, system (12) has to be slightly modified. The unknowns are the errors $e_{i_1}, e_{i_2}, \cdots, e_{i_s}$, their locations $i_1, i_2, \cdots, i_s$ and the erasures $e_{j_1}, e_{j_2}, \cdots, e_{j_t}$. Since a $\hat{B}(n,m)$-code has minimum distance $m+2$, a solution to system (12) exists and is unique. So, it is necessary to build circuits that will solve system (12) in order to complete the decoding. The decoder will have $\left\lfloor \frac{m+1}{2} \right\rfloor + 1$ decoding modes, according to the number $s = 0, 1, 2, \cdots, \left\lfloor \frac{m+1}{2} \right\rfloor$ of track errors that $\hat{B}(n,m)$ can correct. The strategy for choosing a decoding mode is then as follows: count the number t of erasures that have occurred, and then choose the maximum s such that $2s+t \le m+1$. Assume then that s errors have occurred, and this choice of s will determine the decoding mode. The examples in the next section will help to clarify this matter.

4. Examples

In all the examples we take $n = 8$, as in the IBM 3420 series tape unit with 9 tracks.

Example (i): $\hat{B}(8,0)$

This is a code defined by the parity check equation $\sum_{i=0}^{8} Z_i = 0$. Its minimum distance is 2, so it can correct only one track-erasure.

Example (ii): $\hat{B}(8,1)$

This is the well known Patel-Hong code ([2],[4]). The minimum distance of $\hat{B}(8,1)$ is 3, so it can correct either one track-error or two track-erasures. According to (2) and lemma 3.1., this code is defined by

$$\left.\begin{aligned} \sum_{j=0}^{8} Z_j &= 0 \\ \sum_{j=0}^{7} \alpha^j Z_j &= \sum_{j=0}^{7} \alpha^j B_j = 0 \end{aligned}\right\} \tag{13}$$

where α is a root of the irreducible polynomial over $GF(2)$ $g(x) = 1 + x^3 + x^4 + x^5 + x^8$. The redundant bits are in B_0 and in Z_8 (see figure 1). For the encoding, Z_8 is readily obtained, while $B_0 = \sum_{j=1}^{7} \alpha^j B_j$. The circuits are described in [2]. For the decoding, we have two decoding modes. Let us treat them separately.

Mode I: Correction of one track-error

Assume row k is in error, that is, $\hat{Z}_k = Z_k + e_k$ and $\hat{Z}_j = Z_j$ for $j \neq k$. Assume first that $k \neq 8$. According to (11) and (13), the syndrome is

$$\left.\begin{aligned} S_p &= \sum_{i=0}^{8} \hat{Z}_i \\ S_0 &= \sum_{j=0}^{7} \alpha^j \hat{B}_j \end{aligned}\right\} \tag{14}$$

S_p and S_0 are calculated immediately from the received bits. According to (12), if $0 \leq k \leq 7$

$$\left.\begin{aligned} S_p &= e_k \\ S_0 &= \alpha^k e_k \end{aligned}\right\} \tag{15}$$

So, S_p gives us the error e_k, while $\alpha^k = S_0/S_p$ tells us which track is in error. Adding e_k to $\hat{Z}_k$, we obtain Z_k, recovering the information. If the error occurs in track 8, then $S_0 = 0$ and we do not need to bother to recover the information.

Mode II: Correction of two erasures

Assume that the information in tracks i and j is erased. We have to find Z_i and Z_j. So, assume $\hat{Z}_i = \hat{Z}_j = 0$ in order to compute the syndrome S_p and S_0. Hence $e_i = Z_i$ and $e_j = Z_j$. Let $0 \leq i < j \leq 8$. If $j = 8$, then

$$\left.\begin{aligned} S_p &= e_i + e_8 \\ S_0 &= \alpha^i e_i \end{aligned}\right\} \tag{16}$$

So, $e_i = \alpha^{-i} S_0$ and we are not interested in e_8.

If $j < 8$, we have

$$\left.\begin{aligned} S_p &= e_i + e_j \\ S_0 &= \alpha^i e_i + \alpha^j e_j \end{aligned}\right\} \tag{17}$$

Solving this system,

$$e_i = \frac{\alpha^j S_p + S_0}{\alpha^i + \alpha^j} \quad , \quad e_j = \frac{\alpha^i S_p + S_0}{\alpha^i + \alpha^j}$$

The encoding and decoding circuits are discussed in detail in [2].

Example (iii): $\hat{B}(8,2)$

This code has rate 2/3 and was first reported in [1]. According to (2) and lemma 3.1., the code is defined by

$$\left.\begin{aligned} \sum_{i=0}^{8} Z_i &= 0 \\ \sum_{i=0}^{7} \alpha^i Z_i &= \sum_{i=0}^{7} \alpha^i B_i = 0 \\ \sum_{i=0}^{7} \alpha^i (Z_i)^2 &= \sum_{i=0}^{7} \alpha^{2i} B_i = 0 \end{aligned}\right\} \tag{18}$$

Now the parity check bits are contained in B_0, B_1 and Z_8. Let us describe in detail the encoding and the decoding.

Encoding

Z_8 is obtained as in example (ii). B_2, B_3, B_4, B_5, B_6 and B_7 are given, since they contain the information symbols. From (18),

$$\left.\begin{aligned} B_0 + \alpha B_1 &= \sum_{i=2}^{7} \alpha^i B_i \\ B_0 + \alpha^2 B_1 &= \sum_{i=2}^{7} \alpha^{2i} B_i \end{aligned}\right\} \tag{19}$$

Solving system (19), we obtain

$$\left.\begin{aligned} B_0 &= \sum_{i=2}^{7} \alpha^{i+1} \left(\alpha^{i-2} + \alpha^{i-3} + \cdots + 1\right) B_i \\ B_1 &= \sum_{i=2}^{7} \alpha^{i-1} \left(\alpha^{i-1} + \alpha^{i-2} + \cdots + 1\right) B_i \end{aligned}\right\} \tag{20}$$

Circuits performing (20) are easily constructed.

Decoding

Assume rows $\hat{Z}_0, \hat{Z}_1, \cdots, \hat{Z}_8$ are received (resp., columns $\hat{B}_0, \hat{B}_1, \cdots, \hat{B}_7$). The decoder's first step is to calculate the syndrome

$$\left.\begin{aligned} S_p &= \sum_{i=0}^{8} \hat{Z}_i \\ S_0 &= \sum_{i=0}^{7} \alpha^i \hat{B}_i \\ S_1 &= \sum_{i=0}^{7} \alpha^{2i} \hat{B}_i \end{aligned}\right\} \tag{21}$$

If no errors occur, we have $S_p = S_0 = S_1 = 0$. Since $\hat{B}(8,2)$ has minimum distance 4, it can correct either a track-error together with a track-erasure, or three track-erasures. Hence, we need two decoding modes.

Mode I: Correction of a track-error and a track-erasure

Assume that an error pattern e_i occurs in track i and e_j occurs in track j, but j is known, and all the other tracks are correctly transmitted. Assume first that $j \leq 7$. The decoder has to determine first if $i = 8$. Notice that, if $i = 8$, then $S_0 = \alpha^j e_j$ and $S_1 = \alpha^j e_j^2$. Thus, $\alpha^{-j} S_0 = S_1/S_0 = e_j$, or $(S_0)^2 = \alpha^j S_1$. We see that if $i \leq 7$, then $(S_0)^2 \neq \alpha^j S_1$. So, when $i = 8$, this fact is easily determined and we correct track j after finding e_j.

So, assume $\alpha^j S_1 \neq (S_0)^2$, then $i \leq 7$ and the syndrome is given by

$$\left.\begin{aligned} S_p &= e_i + e_j \\ S_0 &= \alpha^i e_i + \alpha^j e_j \\ S_1 &= \alpha^i (e_i)^2 + \alpha^j (e_j)^2 \end{aligned}\right\} \tag{22}$$

Solving system (22), we obtain

$$\alpha^i (S_1 + \alpha^j (S_p)^2) = (S_0)^2 + \alpha^j S_1 \tag{23}$$

Hence, we have to construct circuits that find $S_1 + \alpha^j (S_p)^2$ and $(S_0)^2 + \alpha^j S_1$, then we multiply $S_1 + \alpha^j (S_p)^2$ by α until we obtain $(S_0)^2 + \alpha^j S_1$.

Counting how many times we had to multiply by α, we obtain i. Once we know i, we are in the case of $\hat{B}(8,1)$ with two erasures, i.e., we have to solve system (17).

If $j = 8$, since we are not interested in e_8, we have to solve

$$\left.\begin{aligned} S_0 &= \alpha^i e_i \\ S_1 &= \alpha^i (e_i)^2 \end{aligned}\right\} \tag{24}$$

We obtain $\alpha^i S_1 = (S_0)^2$, so i is easily calculated, and then $e_i = \alpha^{-i} S_1$ gives us e_i.

The decoder's final step is to add e_i and e_j to the corresponding tracks.

Mode II: Correction of a triple track-erasure

Assume erasure patterns e_i, e_j and e_k occur in tracks i, j and k where $0 \leq i < j < k \leq 8$. If $k < 8$, we have

$$\left.\begin{aligned} S_p &= e_i + e_j + e_k \\ S_0 &= \alpha^i e_i + \alpha^j e_j + \alpha^k e_k \\ S_1 &= \alpha^i (e_i)^2 + \alpha^j (e_j)^2 + \alpha^k (e_k)^2 \end{aligned}\right\} \tag{25}$$

The solution of this system is given by the following:

$$\left.\begin{aligned} (e_i)^2 &= \frac{\alpha^{j+k}(S_p)^2 + (S_0)^2 + (\alpha^j + \alpha^k) S_1}{(\alpha^i + \alpha^j)(\alpha^i + \alpha^k)} \\ (e_j)^2 &= \frac{\alpha^{i+k}(S_p)^2 + (S_0)^2 + (\alpha^i + \alpha^k) S_1}{(\alpha^i + \alpha^j)(\alpha^j + \alpha^k)} \\ (e_k)^2 &= \frac{\alpha^{i+j}(S_p)^2 + (S_0)^2 + (\alpha^i + \alpha^j) S_1}{(\alpha^i + \alpha^k)(\alpha^j + \alpha^k)} \end{aligned}\right\} \tag{26}$$

Circuits solving (26) are more complicated than in the case of two erasures, but still perfectly feasible. In order to find e_i, e_j and e_k, we need

to take the square root, but this is easily done, since square root is a 1-1 operation.

Finally, if $k = 8$, we have to solve the system

$$\left.\begin{aligned} S_0 &= \alpha^i e_i + \alpha^j e_j \\ S_1 &= \alpha^i (e_i)^2 + \alpha^j (e_j)^2 \end{aligned}\right\} \tag{27}$$

and the solution is given by

$$\left.\begin{aligned} (e_i)^2 &= \frac{(S_0)^2 + \alpha^j S_1}{\alpha^i(\alpha^i + \alpha^j)} \\ (e_j)^2 &= \frac{(S_0)^2 + \alpha^i S_1}{\alpha^j(\alpha^i + \alpha^j)} \end{aligned}\right\} \tag{28}$$

References

[1] M. Blaum and R. J. McEliece, "Coding Protection for Magnetic Tapes: a Generalization of the Patel-Hong Code," to appear.

[2] S. Lin and D. J. Costello, "Error Control Coding," Prentice-Hall, 1983, 16.2.

[3] F. J. MacWilliams and N. J. A. Sloane, "The Theory of Error-Correcting Codes," North Holland, Amsterdam, 1978.

[4] A. M. Patel and S. J. Hong, "Optimal Rectangular Code for High Density Magnetic Tapes," IBM J. Res. Dev., 18, pp 579-588, November 1974.

APPENDIX I

ASYMPTOTIC ESTIMATES OF INTEGRALS

We want to prove theorem 3.1. of chapter 1. We need some lemmas first.

1. Lemma

Let $I_t = \int_0^\infty e^{Ma_k x^k} x^t \, dx$ *where* $a_k < 0$, $M > 0$ *and* t *and* k *are positive integers. Then,*

$$I_t = \frac{(M(-a_k))^{-\frac{t+1}{k}}}{k} \Gamma\left(\frac{t+1}{k}\right) \tag{1}$$

In particular,

$$I_t = O\left(M^{-\frac{t+1}{k}}\right) \tag{2}$$

Proof: Make the substitution $M(-a_k)x^k = u$. Then,

$$x = \left(\frac{u}{-Ma_k}\right)^{1/k} \quad , \quad dx = \frac{u^{\frac{1}{k}-1}}{k(-Ma_k)^{1/k}} \, du, \quad \text{so}$$

$$I_t = \frac{1}{k(-Ma_k)^{\frac{t+1}{k}}} \int_0^\infty e^{-u} u^{\frac{t+1}{k}-1} \, du$$

$$= \frac{(-Ma_k)^{\frac{-t+1}{k}}}{k} \Gamma\left(\frac{t+1}{k}\right) = O\left(M^{-\frac{t+1}{k}}\right)$$

Proved. ∎

2. Lemma

Let $\sum_{i\geq 0, j\geq 0} c_{ij} z^i w^j$ *be a double series convergent for* $0 \leq z < 2R$, $0 \leq w < 2S$. *Then, if* $0 \leq z < R/3$, $0 \leq w < S/3$, A *is a positive integer,*

$$\sum_{\substack{i\geq 0, j\geq 0\\ i+j>A}} c_{ij} z^i w^j = O\left(z^{A+1}\right) + O\left(w^{A+1}\right) \tag{3}$$

Proof: $c_{ij} = O\left(R^{-i}S^{-j}\right)$ since the terms of a convergent series are bounded. Now, if $0 \leq z < R/3$, $0 \leq w < R/3$, we estimate

$$\begin{aligned}\sum_{\substack{i\geq 0, j\geq 0\\ i+j>A}} c_{ij} z^i w^j &= O\left(\sum_{i+j>A} \left(\frac{z}{R}\right)^i \left(\frac{w}{S}\right)^j\right)\\ &= O\left(\sum_{l=A+1}^{\infty} \left(\frac{z}{R}+\frac{w}{S}\right)^l\right)\\ &= O\left[\left(\frac{z}{R}+\frac{w}{S}\right)^{A+1}\right] = O\left[(z+w)^{A+1}\right]\end{aligned}$$

But

$$\begin{aligned}(z+w)^{A+1} &\leq (2\max\{z,w\})^{A+1} = 2^{A+1}\max\{z^{A+1}, w^{A+1}\}\\ &\leq 2^{A+1}(z^{A+1}+w^{A+1})\end{aligned}$$

Hence (3) holds. ∎

Notice that estimate (3) is not uniform with respect to A.

We can now prove theorem 3.1. of chapter I. Let us state it again.

3. Theorem

Let $F(M) = \int_0^\infty g(x)e^{Mh(x)}\,dx$, where g is continuous and positive when $x \geq 0$, h is infinitely differentiable for $x \geq 0$, $h(x) < h(0)$ for all $x > 0$,

$$h'(0) = h''(0) = \cdots = h^{(k-1)}(0) = 0 \quad , \quad h^{(k)}(0) < 0$$

for some $k \geq 1$, $\lim_{x\to\infty} h(x) = -\infty$, $\int_0^\infty g(x)e^{h(x)}\,dx$ converges, and let

$$h(x) = a_0 + \sum_{j \geq k} a_j x^j \quad , \quad g(x) = \sum_{j=0}^{\infty} b_j x^j$$

for $0 \leq x \leq \delta$ *for some* $\delta > 0$. *Then*

$$F(M) \sim \left(\sum_{\nu=0}^{\infty} d_\nu M^{-\frac{\nu+1}{k}} \right) e^{Mh(0)} \tag{4}$$

where

$$d_\nu = \frac{(-a_k)^{-\frac{\nu+1}{k}}}{k} \sum_{j=0}^{\nu} c_{j,\nu-j} (-a_k)^{-j} \Gamma\left(j + \frac{\nu+1}{k}\right) \tag{5}$$

and

$$g(v) \exp\left(u \left(\sum_{s=0}^{\infty} a_{k+1+s} v^s \right)\right) = \sum_{i,j} c_{ij} u^i v^j \quad in \quad [0, \delta) \tag{6}$$

Proof: Assume $h(0) = a_0 = 0$. Claim: for any $\tau > 0$ and l a positive integer,

$$\int_\tau^\infty g(x) e^{Mh(x)}\, dx = O\left(M^{-l}\right) \tag{7}$$

as $M \to \infty$. In effect, since $\lim_{x \to \infty} = -\infty$ and h is continuous, there exists a constant $c > 0$ such that $h(x) < -c$ for all $x \geq \tau$. Thus,

$$\begin{aligned} \int_\tau^\infty g(x) e^{Mh(x)}\, dx &= \int_\tau^\infty g(x) e^{(M-1)h(x)} e^{h(x)}\, dx \\ &\leq e^{-c(M-1)} \int_0^\infty g(x) e^{h(x)}\, dx = O\left(e^{-cM}\right) = O\left(M^{-l}\right) \end{aligned}$$

as claimed. Now, consider $e^{Ma_k x^k}$ as the main factor in the integrand. The remaining factor $g(x) \exp\left(Mx^{k+1}(a_{k+1} + a_{k+2}x + a_{k+3}x^2 + \cdots)\right)$ can be expanded as a double power series in the two arguments Mx^{k+1} and x, conver-

gent for $0 \leq x < \delta$ and for all values of Mx^{k+1}. We denote this double series by

$$P(Mx^{k+1}, x) = \sum_{i,j} c_{ij} \left(Mx^{k+1}\right)^i x^j$$

The coefficients c_{ij} are independent of M and x. We want to approximate P uniformly by its partial sums. Therefore, we restrict Mx^{k+1} to some finite interval. Take for instance $0 \leq Mx^{k+1} < 1$. Then, we use the power series only if $0 \leq x \leq M^{-1/k+1}$. Call $\tau = M^{-1/k+1}$. We may assume that $M > \delta^{-(k+1)}$, whence $\tau < \delta$.

Choose a positive integer A and write

$$P_A\left(Mx^{k+1}, x\right) = \sum_{\substack{i \geq 0, j \geq 0 \\ i+j \leq A}} c_{ij} \left(Mx^{k+1}\right)^i x^j$$

Then, we have

$$\int_0^\infty g(x) e^{Mh(x)}\, dx - \int_0^\infty P_A e^{Ma_k x^k}\, dx =$$

$$= \int_0^\tau (P - P_A) e^{Ma_k x^k}\, dx + \int_\tau^\infty g(x) e^{Mh(x)}\, dx - \int_\tau^\infty P_A e^{Ma_k x^k}\, dx =$$

$$= \int_0^\tau (P - P_A) e^{Ma_k x^k}\, dx + O\left(M^{-l}\right) + O\left(\int_\tau^\infty e^{Ma_k x^k} x^A\, dx\right) \tag{8}$$

this last step by (7). Notice that $\int_\tau^\infty e^{Ma_k x^k} x^A\, dx = O\left(M^{-l}\right)$ as $M \to \infty$ by (7), with x^A in place of $g(x)$ and $a_k x^k$ in place of $h(x)$. From (3),

$$P - P_A = O\left(\left(Mx^{k+1}\right)^{A+1}\right) + O\left(x^{A+1}\right),$$

for x small enough, thus,

$$\int_0^r (P - P_A)e^{Ma_kx^k} = O\left(\int_0^\infty M^{A+1}e^{Ma_kx^k}x^{(k+1)(A+1)}\,dx\right) + O\left(\int_0^\infty e^{Ma_kx^k}x^{A+1}\,dx\right)$$

so, replacing in (8) and using the definition of I_t in lemma 1, by (2), we obtain

$$\int_0^\infty g(x)e^{Mh(x)}\,dx - \int_0^\infty P_A e^{Ma_kx^k}\,dx =$$
$$= M^{A+1}O\left(I_{(k+1)(A+1)}\right) + O\left(I_{A+1}\right) + O\left(M^{-l}\right) =$$
$$= O\left(M^{-\frac{A+2}{k}}\right) + O\left(M^{-\frac{A+2}{k}}\right) + O\left(M^{-l}\right) = O\left(M^{-\frac{A+2}{k}}\right)$$

since l is arbitrary. But

$$\int_0^\infty P_A e^{Ma_kx^k}\,dx = \sum_{\substack{i\geq 0, j\geq 0\\ i+j\leq A}} c_{ij}\int_0^\infty (Mx^{k+1})^i x^j e^{Ma_kx^k}\,dx$$
$$= \sum_{i+j\leq A} c_{ij}M^i I_{i(k+1)+j}$$
$$= \frac{1}{k}\sum_{i+j\leq A} c_{ij}M^{-\frac{i+j+1}{k}}(-a_k)^{-\frac{i(k+1)+j+1}{k}}\Gamma\left(\frac{i(k+1)+j+1}{k}\right)$$

So,

$$\int_0^\infty g(x)e^{Mh(x)}\,dx = \frac{1}{k}\sum_{i+j\leq A} c_{ij}M^{-\frac{i+j+1}{k}}(-a_k)^{-\frac{i(k+1)+j+1}{k}}\Gamma\left(\frac{i(k+1)+j+1}{k}\right) + O\left(M^{-\frac{A+2}{k}}\right)$$

Hence, we have

$$\int_0^\infty g(x)e^{Mh(x)}\,dx = \sum_{\nu=0}^{A} d_\nu M^{-\frac{\nu+1}{k}} + O\left(M^{-\frac{A+2}{k}}\right)$$

where

$$d_\nu = \frac{1}{k}\sum_{i=0}^{\nu} c_{i,\nu-i}(-a_k)^{-i-\frac{\nu+1}{k}}\,\Gamma\left(i+\frac{\nu+1}{k}\right)$$
$$= \frac{(-a_k)^{-\frac{\nu+1}{k}}}{k}\sum_{i=0}^{\nu} c_{i,\nu-i}(-a_k)^{-i}\,\Gamma\left(i+\frac{\nu+1}{k}\right)$$

If $h(0) = a_0 \neq 0$, we simply have an extra factor $e^{Mh(0)}$. ∎

APPENDIX II

HOW GOOD IS THE POISSON APPROXIMATION?

In chapter II, we found the $MTBF$ and CG of singly and doubly error-protected computer memories. In particular, the singly error-protected case allowed all kinds of errors to occur. An important assumption when we found the reliability $R(t)$ of a row of chips was, failures in the row are distributed according to a Poisson process. Without this assumption, the formulae would become hopelessly complicated. The question is, how good is this Poisson approximation?

In order to answer this, we are going to find $MTBF$ and CG for both singly and doubly error-protected computer memories in a simplified situation: We assume that all the chip failures are catastrophic. We shall consider the asymptotic case of M rows of chips, where M is a large number. The notation will be the same as the one used in chapter II.

When k failures occur in a row, the Poisson approximation we used in chapter II was

$$\binom{n}{k}\left(1-e^{-\lambda t}\right)^k e^{-\lambda t(n-k)} \sim \frac{(\lambda n t)^k}{k!} e^{-\lambda n t} \tag{1}$$

We consider the two cases separately.

Case (i): Single-error protection

The reliability of a row when 1-ECC is implemented is

$$R(t) = e^{-\lambda nt} + n(1 - e^{-\lambda t})e^{-\lambda(n-1)t}$$
$$= e^{-\lambda nt}\left(1 + n(e^{\lambda t} - 1)\right) \tag{2}$$

Thus,

$$MTBF = \int_0^\infty (R(t))^M \, dt$$
$$= \int_0^\infty e^{-\lambda nMt}\left(1 + n(e^{\lambda t} - 1)\right)^M \, dt \tag{3}$$

Making the change of variable $\lambda nt = x$, we obtain

$$MTBF = \frac{1}{\lambda n}\int_0^\infty e^{-Mx}\left(1 + n(e^{x/n} - 1)\right)^M \, dx$$
$$= \frac{1}{\lambda n}\int_0^\infty e^{Mh(x)} \, dx \tag{4}$$

where

$$h(x) = \log\left(1 + n(e^{x/n} - 1)\right) - x \tag{5}$$

Hence, dividing by $(\lambda Mk)^{-1}$,

$$CG = \frac{k}{n}M\int_0^\infty e^{Mh(x)} \, dx \tag{6}$$

We need to estimate $I = \int_0^\infty e^{Mh(x)} \, dx$. We easily verify

$$h(0) = h'(0) = 0 \quad \text{and} \quad h''(0) = -\frac{n-1}{n} \tag{7}$$

Using (11), chapter II, we have

$$I \sim \sqrt{\frac{\pi}{2M\,(-h''(0))}} = \sqrt{\frac{n\pi}{2(n-1)M}} \tag{8}$$

Replacing in (4) and (6), we obtain

$$MTBF \sim \frac{1}{\lambda n}\sqrt{\frac{n}{n-1}}\sqrt{\frac{\pi}{2M}} \tag{9}$$

and

$$CG \sim \frac{k}{n}\sqrt{\frac{n}{n-1}}\sqrt{\frac{\pi M}{2}} \tag{10}$$

Using the Poisson approximation, Goodman and McEliece found ([3], chapter II),

$$CG \sim \frac{k}{n}\sqrt{\frac{\pi M}{2}}$$

(This approximation is also obtained taking $a = b = c = 0$, $d = 1$ in (14), chapter II). So, the two values differ by a factor $\sqrt{\frac{n}{n-1}}$.

In our typical examples, $n = 39$. Hence $\sqrt{\frac{n}{n-1}} \cong 1.01$. This means, the Poisson approximation is very good in this case.

Case (ii): Doubly error protection

Now the reliability of each row is given by

$$\begin{aligned} R(t) &= e^{-\lambda nt} + n(1-e^{-\lambda t})e^{-\lambda(n-1)t} + \frac{n(n-1)}{2}\left(1-e^{\lambda t}\right)^2 e^{-\lambda(n-2)t} \\ &= e^{-\lambda nt}\left(1 + n(e^{\lambda t}-1) + \frac{n(n-1)}{2}(e^{\lambda t}-1)^2\right) \end{aligned} \tag{11}$$

thus, making the change of variable $\lambda nt = x$,

$$MTBF = \frac{1}{\lambda n}\int_0^\infty e^{-Mx}\left[1 + n(e^{x/n} - 1) + \frac{n(n-1)}{2}(e^{x/n} - 1)^2\right]^M dx \quad (12)$$

We can also write (12) as

$$MTBF = \frac{1}{\lambda n}\int_0^\infty e^{-Mh(x)}\, dx \quad (13)$$

where

$$h(x) = \log\left(1 + n(e^{x/n} - 1) + \frac{n(n-1)}{2}(e^{x/n} - 1)^2\right) - x \quad (14)$$

Differentiating, we obtain

$$h(0) = h'(0) = h''(0) = 0 \quad , \quad h'''(0) = -\frac{n^2 - 3n + 2}{6n^2}$$

Applying the theorem proved in appendix I to (13), and taking first approximation, we get

$$MTBF \sim \frac{1}{\lambda n}\left(\frac{6}{1 - \frac{3}{n} + \frac{2}{n^2}}\right)^{\frac{1}{3}} \Gamma\left(\frac{4}{3}\right) M^{-\frac{1}{3}} \quad (15)$$

and

$$CG \sim \frac{k}{n}\left(\frac{6}{1 - \frac{3}{n} + \frac{2}{n^2}}\right)^{\frac{1}{3}} \Gamma\left(\frac{4}{3}\right) M^{\frac{2}{3}} \quad (16)$$

as $M \to \infty$. (15) and (16) differ from the corresponding results that use the Poisson approximation by the factor $\left(1 - \frac{3}{n} + \frac{2}{n^2}\right)^{-\frac{1}{3}}$.

In typical cases, when we have doubly error protection, $n = 45$. In this case $\left(1 - \frac{3}{n} + \frac{2}{n^2}\right)^{-\frac{1}{3}} \cong 1.02$. Hence, the Poisson approximation is also very good in this case.