Burst Codes

Alex Broihier, Porter Shawver

Outline

Section 1

Cyclic Codes

Burst Errors

- Burst Description (P, L) where P is the error, L is the starting index
- $E = [0, 1, 1, 0, 0, 1]$ is the error in some message
- $(11001, 2)$ describes E
- More common in the real world (think scratching a CD, the internet dropping in the middle of a message, etc.)

What are Cyclic Codes

- Invariant under rotation
  - 011001, 101100, 010110, 001011, 100101, 110010 all the same
- When considering cycles with burst errors, the burst description is no longer unique
- $E = [0, 1, 1, 0, 0, 1]$ is described by $(11001, 2), (100101, 3),$ and $(1011, 6)$
- $E \xrightarrow{\text{Rotated To}} [1, 0, 0, 1, 0, 1]$ is described by $(1011, 4)$
- $E \xrightarrow{\text{Rotated To}} [1, 0, 1, 1, 0, 0]$ is described by $(100101, 4)$

Generating Functions for Linear Cyclic Codes

- Coefficient of each term corresponds to a corresponding digit in code
- $g(x) = 1x^0 + 0x^1 + 1x^2 + 1x^3 + 0x^4$ corresponds to 10110
- A multiplication by x corresponds to a rotation:

$$
\begin{aligned}
x \cdot g(x) &= 1x^1 + 0x^2 + 1x^3 + 1x^4 + 0x^5 \\
&= 1x^1 + 0x^2 + 1x^3 + 1x^4 + 0x^0 \\
&= 0x^0 + 1x^1 + 0x^2 + 1x^3 + 1x^4 \\
&\to 01011
\end{aligned}
$$

Cyclic Codespace

Let w be the original, un-encoded message.

$$w \xrightarrow{\text{Encode}} w \cdot g(x) \xrightarrow{\text{Transmission Error}} w \cdot g(x) + e(x) \xrightarrow{\text{Mod } g(x)} e(x).$$

- $e(x)$ obtained as remainder when dividing by $g(x)$

Example

- Say we want to encode $\{00, 10, 01, 11\}$
- Let's pick $g(x) = 1 + x^2$ as our generator

$$
\begin{array}{llll}
\quad 00 & 10 & 01 & 11 \\
\to 0 & 1 & x & 1 + x \\
\to 0(1 + x^2) & 1(1 + x^2) & x(1 + x^2) & (1 + x)(1 + x^2) \\
\to 0 & 1 + x^2 & x + x^3 & 1 + x + x^2 + x^3 \\
\to 0000 & 1010 & 0101 & 1111
\end{array}
$$

- If I receive 0100, I know an error occurred

Example

1. Codewords: $\{0000, 1010, 0101, 1111\}$
2. Generator Polynomial: $g(x) = 1 + x^2$
3. Receive 1110

$$1110 \to 1 + x + x^2 \qquad \to \frac{1+x+x^2}{1+x^2} = 1 + \frac{x}{1+x^2} \qquad \to x \text{ is the remainder}$$

$$\to \text{ error } = 0100 \to \text{original message} = 1010$$

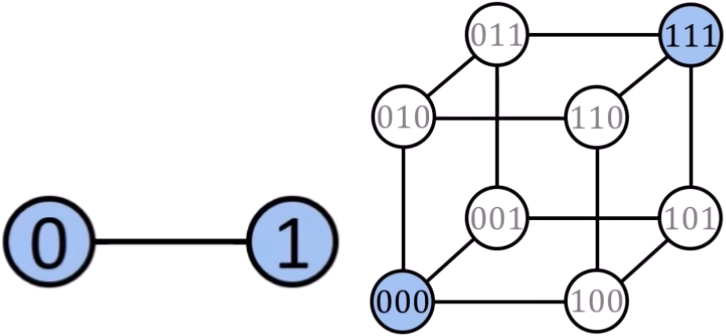
Cosets

- The set of all errors that differ by a code word: $e_1 = e_2 + c$
- *Ex*: for the previous example, the error 0100 is in the same coset as 1110, 0001 and 1011, by adding the codewords 1010, 0101, and 1111 respectively.

Lemma

A linear code C is an ℓ-burst-error-correcting code if distinct burst errors of length $\leq \ell$ are in distinct cosets of C.

Cosets Hand-Wavy Intuition



Section 2

Fire Codes

Fire Codes

- Type of burst error correcting code.
- Appeared originally in Philip Fire's 1959 dissertation, *A class of multiple-error-correcting binary codes for non-independent errors.*

Building a Fire Code

- Let $p(x)$ be a prime/irreducible polynomial of degree m over $\mathbb{F}_2$.
  - An irreducible polynomial cannot be factored into products of non-constant polynomials.
- Let ρ be the smallest integer such that $p(x) \mid (1 + x^\rho)$. ρ is called the *period*.
- Let ℓ be a positive integer not divisible by ρ with $\ell \leq m$.
- $g(x) = (1 + x^{2\ell-1})p(x)$ is the generator polynomial for a Fire code.

Example

- Start with $p(x) = 1 + x + x^3$. (Note: $m = 3$.)
- We can find ρ with $\rho = 2^m - 1$, so $\rho = 2^3 - 1 = 7$.
- Select $\ell = 3$. We have $\ell \leq m$ and $p \not| (2\ell - 1)$, so this choice works.
- Thus,

$$
\begin{aligned}
g(x) &= (1 + x^{2\ell-1})p(x) \\
&= (1 + x^5)(1 + x + x^3) \\
&= 1 + x + x^3 + x^5 + x^6 + x^8
\end{aligned}
$$

Correct codes of length $\leq \ell$

Theorem

Fire codes can correct burst errors of length ℓ.

Proof

General idea: proof by contradiction of the lemma from before that distinct burst errors must be in distinct cosets.

Lemma

$(1 + x^{2\ell-1})$ and $p(x)$ (the factors of $g(x)$) are relatively prime.

Proof

- Take two *distinct* burst errors with lengths $\ell_1, \ell_2 < \ell$ represented by

$$a(x) = 1 + a_1 x + a_2 x^2 + \cdots + a_{\ell_1 - 2} + x^{\ell_1 - 1}$$
$$b(x) = 1 + b_1 x + b_2 x^2 + \cdots + b_{\ell_2 - 2} + x^{\ell_2 - 1}$$

- These errors could be anywhere, so we write $x^i a(x)$ and $x^j b(x)$ for some $i, j < n$ representing start of error ($WLOG$ assume $i < j$).
- Suppose for contradiction $x^i a(x)$ and $x^j b(x)$ are in the same coset. ($x^i a(x) = x^j b(x) + c$ for some code word c)
- Then their sum, $x^i a(x) + x^j b(x)$, is a polynomial $v(x)$ in the code.
- Let q, b such that $j - i = q(2\ell - 1) + b$.

Proof

- Then

$$
\begin{aligned}
v(x) &= x^i a(x) + x^j b(x) \\
&= x^i a(x) + x^j b(x) + 2x^{b+i} b(x) \\
&= x^i \big(a(x) + x^b b(x)\big) + x^{b+i} b(x)(1 + x^{q(2\ell-1)})
\end{aligned}
$$

- Because $v(x)$ represents code word, it is divisible by $g(x)$.
- Because the factors of $g(x)$ are relatively prime, $v(x)$ must be divisible by $1 + x^{2\ell-1}$.
- So $a(x) + x^b b(x)$ is divisible by $1 + x^{2\ell-1}$ (or is 0). Let $d(x)$ be the quotient with degree δ.

Proof

$$\overbrace{d(x)}^{\delta}\overbrace{(1+x^{2\ell-1})}^{2\ell-1} = \overbrace{a(x)}^{\ell_1-1} + \overbrace{x^b b(x)}^{b+\ell_2-1}$$

Proof

$$\underbrace{\overbrace{d(x)}^{\delta}\overbrace{(x^{2\ell-1}+1)}^{2\ell-1}}_{2\ell-1+\delta} = \overbrace{a(x)}^{\ell_1-1} + \overbrace{x^b b(x)}^{b+\ell_2-1}$$

$$\begin{aligned} \ell_1 - 1 &< 2\ell - 1 \\ \implies b + \ell_2 - 1 &= 2\ell - 1 + \delta \\ \implies b &= 2\ell - \ell_2 + \delta \\ \implies b &\geq \ell + \delta \end{aligned}$$

$$\Downarrow$$

$$b > \ell_1 - 1 \text{ and } b > \delta$$

Proof

$$b > \ell_1 - 1 \text{ and } b > \delta$$

- Using $b > \ell_1 - 1$, we know x^b appears in the expansion of $a(x) + x^b b(x)$:

$$\begin{aligned} &1 + a_1 x + a_2 x^2 + \cdots + a_{\ell_1 - 2} + x^{\ell_1 - 1} \\ &\quad + x^b(1 + b_1 x + b_2 x^2 + \cdots + b_{\ell_2 - 2} + x^{\ell_2 - 1}) \end{aligned}$$

- Then, using $b > \delta$, we know $d(x)$ does not have x^b, so $a(x) + x^b b(x)$ is not divisible by $d(x)$.
- Recall this means $a(x) + x^b b(x) = 0$.

Proof

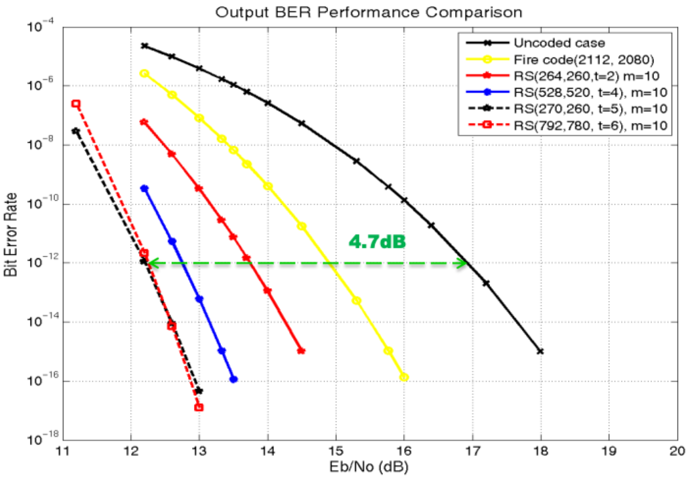
$$
\begin{aligned}
a(x) + x^b b(x) &= 1 + a_1 x + a_2 x^2 + \cdots + a_{\ell_1 - 2} + x^{\ell_1 - 1} \\
&\quad + x^b (1 + b_1 x + b_2 x^2 + \cdots + b_{\ell_2 - 2} + x^{\ell_2 - 1}) \\
&= 0 \\
&\implies b = 0 \qquad\qquad \text{(remember, we're in } \mathbb{F}_2\text{)} \\
&\implies a(x) = b(x) \\
&\implies \text{Contradiction!}
\end{aligned}
$$

So, if two errors are distinct, they are in different cosets.

Example and Analysis

- Recall our example of $g(x) = (1 + x^5)(1 + x + x^3)$ with $m = 3$, $\rho = 7$, and $\ell = 3$.
- Block length $n = \mathit{LeastCommonMulitple}(2\ell - 1, p)$.
  - In this case, $n = \mathit{LCM}(5, 7) = 35$.
- Original message length $k = n - m - 2\ell + 1$.
  - $k = 27$.
  - Would be 29.8 if it were a Hamming code*.
- $(35, 27)$ code. Rate gets better with larger blocks.

Not as Good as Reed-Solomon



Section 3

Interleaved Codes

Interleaved Codes

- We have many codes that work well, if errors are randomly distributed in our message
- But errors are more likely to be spatially correlated
- What if we split up the errors, so errors within a burst error are spread out across different words?

Interleaved Codes

- Built off of codes that are better suited for randomly distributed errors (ex: Hamming Codes)
- After encoding the message, but before sending it, we use some bijective function to scramble up the bits (the interleave step)
- We send the message, and some burst error occurs
- After receiving the message, we descramble the bits (the deinterleave step), sending errors to different code words
- We use the underlying code to detect and / or correct errors

Interleaved Codes With a Block Interleaver

- One way to interleave a message
- Organize message as a $M \times N$ matrix: write bits in row major order, read in column major order
- Alternatively, write matrix in row major order, transpose the matrix, read the matrix in row major order

$$x_0x_1x_2x_3x_4x_5x_6x_7x_8 \rightarrow \begin{array}{|c|c|c|}\hline x_0 & x_1 & x_2 \\ \hline x_3 & x_4 & x_5 \\ \hline x_6 & x_7 & x_8 \\ \hline \end{array} \rightarrow x_0x_3x_6x_1x_4x_7x_2x_5x_8$$

Block Interleaver Example

000 000 111 000 $\rightarrow$

0	0	0
0	0	0
1	1	1
0	0	0

$\rightarrow$ 001 000 100 010

Block Interleaver Example

$$001000100010 + 00111 0000000 = 00011 0100010 \rightarrow$$

0	1	0
0	0	0
0	1	1
1	0	0

$\rightarrow$ 010 000 011 100

Block Interleaver Analysis

- Take a burst error of length ℓ
- After interleaving, the distance between consecutive errors becomes M
- We need a burst error of length $M \cdot \ell + 1$ to get $\ell + 1$ consecutive errors in the output
- Thus, a code that can correct t errors can correct $M \cdot t$ burst errors.

Block Interleaver Analysis

- Block Interleaver takes up $M \cdot N$ space
- We can measure its efficiency by comparing how many errors can occur until it fails and how much space it takes up
- efficiency $= \frac{M \cdot t + 1}{M \cdot N} \approx \frac{M \cdot t}{M \cdot N} = \frac{t}{N}$

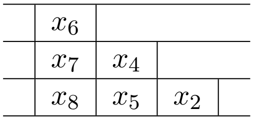
Block Interleaver Analysis

- One major downside: we need to read almost the entire transmitted message before we can start deinterleaving it and running error detection / correction on it
- Not necessarily as good for data streams
- Possible solution: apply interleaving to blocks of data at a time
- Possible issue with this solution: how do we know when blocks start / end in the data stream?

Interleaved Codes With a Convolution Interleaver

- A different interleaving approach
- Sometimes called a cross interleaver
- Interleave by putting consecutive elements into consequtive queues of varying lengths

$\ldots x_3 x_4 x_5 x_6 x_7 x_8 \ldots \rightarrow$



$\rightarrow \ldots x_6 x_4 x_2 x_9 x_7 x_5 \ldots$

Convolution Interleaver: Another Approach

- Write the message as a matrix and shift columns down by varying amounts.

$x_0x_1x_2x_3x_4x_5x_6x_7x_8 \rightarrow$

x_0	x_1	x_2
x_3	x_4	x_5
x_6	x_7	x_8

Convolution Interleaver: Another Approach

- Write the message as a matrix in row major order and shift columns down by varying amounts.

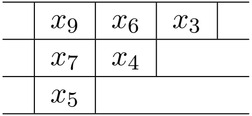
$x_0 x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 \rightarrow$

x_0		
x_3	x_1	
x_6	x_4	x_2
	x_7	x_5
		x_8

$\rightarrow \ldots x_6 x_4 x_2 \ldots$

Convolution Interleaver: Deinterleaving

$\ldots x_6 x_4 x_2 x_9 x_7 x_5 \ldots \rightarrow$



$\rightarrow \ldots x_3 x_4 x_5 x_6 x_7 x_8 \ldots$

Convolution Interleaver: Deinterleaving

$\ldots x_6 x_4 x_2 \ldots \rightarrow$

x_0		
x_3	x_1	
x_6	x_4	x_2
	x_7	x_5
		x_8

x_0	x_1	x_2
x_3	x_4	x_5
x_6	x_7	x_8

$\rightarrow x_0 x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8$

Convolution Interleaver Example

000 000 111 000 →

0	0	0
0	0	0
1	1	1
0	0	0

→

0	-	-
0	0	-
1	0	0
0	1	0
-	0	1
-	-	0

→ 0__00_100010_01__0

Convolution Interleaver Example

$$
\begin{aligned}
0\_\_00\_100010\_01\_\_0 + 0\_\_01\_100100\_00\_\_0 \\
= 0\_\_01\_000110\_01\_\_0
\end{aligned}
$$

$\rightarrow$

0	-	-
0	1	-
0	0	0
1	1	0
-	0	1
-	-	0

Convolution Interleaver Example

$\rightarrow$

0	1	0
0	0	0
0	1	1
1	0	0

$\rightarrow$ 010 000 011 100

Convolution Interleaver Analysis

- Difference between consecutive errors becomes $N + 1$
- We can correct up to $(N + 1)(t - 1)$ errors
- Takes up $0 + 1 + \ldots + (N - 1) = \frac{N(N-1)}{2}$ space
- We no longer have to read nearly the entire message to start decoding

Convolution Interleaver Analysis

- We can measure its efficiency by comparing how many errors can occur until it fails and how much space it takes up
- efficiency $= \frac{(N+1)(t-1)+1}{\frac{N(N-1)}{2}} \approx \frac{N \cdot t}{\frac{N^2}{2}} = \frac{2t}{N}$
- Notice that the efficiency for the convolution interleaver is approximately twice as good as that of the block interleaver (which had efficiency $= \frac{t}{N}$)

Section 4

Unary Codes

Unary Codes

$$000000 \xrightarrow{\text{error of } 0{\color{red}0}0{\color{red}00}0} 0{\color{red}0}0{\color{red}00}0 \rightarrow 000000$$

Information is the resolution of uncertainty.

— Claude Shannon (1948)

Bibliography

Moon, Todd K. *Error Correction Coding: Mathematical Methods and Algorithms.* John Wiley & Sons, Inc., 2005.