15-853:Algorithms in the Real World

Error Correcting Codes II

- Cyclic Codes
- Reed-Solomon Codes

Some Number Theory

Groups

- Definitions, Examples, Properties

Fields

- Definition, Examples
- Polynomials
- Galois Fields

Why does number theory play such an important role?

It is **the** mathematics of finite sets of values.

Groups

A **Group** $(G,*,I)$ is a set G with operator * such that:

1. **Closure**. For all $a,b \in G$, $a * b \in G$
2. **Associativity**. For all $a,b,c \in G$, $a*(b*c) = (a*b)*c$
3. **Identity**. There exists $I \in G$, such that for all $a \in G$, $a*I=I*a=a$
4. **Inverse**. For every $a \in G$, there exist a unique element $b \in G$, such that $a*b=b*a=I$

An **Abelian or Commutative Group** is a Group with the additional condition

5. **Commutativity**. For all $a,b \in G$, $a*b=b*a$

Examples of groups

- Integers, Reals or Rationals with Addition
- The nonzero Reals or Rationals with Multiplication
- Non-singular n x n real matrices with Matrix Multiplication
- Permutations over n elements with composition
  $[0\to1, 1\to2, 2\to0] \circ [0\to1, 1\to0, 2\to2] = [0\to0, 1\to2, 2\to1]$

We will only be concerned with **finite groups**, I.e., ones with a finite number of elements.

Key properties of finite groups

Notation: $a^j \equiv a * a * a * \ldots$ j times

Theorem (Fermat's little): for any finite group $(G,*,I)$ and $g \in G$, $g^{|G|} = I$

Definition: the **order** of $g \in G$ is the smallest positive integer m such that $g^m = I$

Definition: a group G is **cyclic** if there is a $g \in G$ such that order(g) = $|G|$

Definition: an element $g \in G$ of order $|G|$ is called a **generator** or **primitive element** of G.

Groups based on modular arithmetic

The group of positive integers modulo a prime *p*

$Z_p^* \equiv \{1, 2, 3, \ldots, p\text{-}1\}$

$*_p \equiv$ multiplication modulo p

Denoted as: $(Z_p^*, *_p)$

Required properties

1. Closure. Yes.
2. Associativity. Yes.
3. Identity. 1.
4. Inverse. Yes.

Example: $Z_7^* = \{1,2,3,4,5,6\}$

$1^{-1} = 1$, $2^{-1} = 4$, $3^{-1} = 5$, $6^{-1} = 6$

Other properties

$|Z_p^*| = (p-1)$

By Fermat's little theorem: $a^{(p-1)} = 1 \pmod p$

Example of Z_7^*

x	x^2	x^3	x^4	x^5	x^6
1	1	1	1	1	1
2	4	1	2	4	1
3	**2**	**6**	**4**	**5**	**1**
4	2	1	4	2	1
5	**4**	**6**	**2**	**3**	**1**
6	1	6	1	6	1

Generators

For all p the group is cyclic.

Fields

A **Field** is a set of elements F with binary operators * and + such that

1. (F, +) is an **abelian group**
2. (F \ I_+, *) is an **abelian group**
   the "multiplicative group"
3. **Distribution**: a*(b+c) = a*b + a*c
4. **Cancellation**: a*I_+ = I_+

The **order** of a field is the number of elements.

A field of finite order is a **finite field**.

The reals and rationals with + and * are fields.

Finite Fields

Z_p (p prime) with + and * mod p, is a **finite** field.

1. $(Z_p, +)$ is an **abelian group** (0 is identity)
2. $(Z_p \setminus 0, *)$ is an **abelian group** (1 is identity)
3. **Distribution**: $a*(b+c) = a*b + a*c$
4. **Cancellation**: $a*0 = 0$

Are there other finite fields?

What about ones that fit nicely into bits, bytes and words (i.e with 2^k elements)?

Polynomials over Z_p

$Z_p[x]$ = polynomials on x with coefficients in Z_p.

- Example of $Z_5[x]$: $f(x) = 3x^4 + 1x^3 + 4x^2 + 3$
- $\deg(f(x)) = 4$ (the **degree** of the polynomial)

Operations: (examples over $Z_5[x]$)

- Addition: $(x^3 + 4x^2 + 3) + (3x^2 + 1) = (x^3 + 2x^2 + 4)$
- Multiplication: $(x^3 + 3) * (3x^2 + 1) = 3x^5 + x^3 + 4x^2 + 3$
- $I_+ = 0$, $I_* = 1$
- + and * are associative and commutative
- Multiplication distributes and 0 cancels

Do these polynomials form a field?

Division and Modulus

Long division on polynomials ($Z_5[x]$):

$$
\begin{array}{r}
\boxed{1x+4} \\
x^2+1 \overline{\smash{\big)}\, x^3+4x^2+0x+3} \\
\underline{x^3+0x^2+1x+0} \\
4x^2+4x+3 \\
\underline{4x^2+0x+4} \\
\boxed{4x+4}
\end{array}
$$

$$(x^3+4x^2+3)/(x^2+1)=(x+4)$$

$$(x^3+4x^2+3)\bmod(x^2+1)=(4x+4)$$

$$(x^2+1)(x+4)+(4x+4)=(x^3+4x^2+3)$$

Polynomials modulo Polynomials

How about making a field of polynomials modulo another polynomial? This is analogous to Z_p (i.e., integers modulo another integer).

e.g. $Z_5[x] \bmod (x^2+2x+1)$

Does this work?

Does $(x + 1)$ have an inverse?

Definition: An **irreducible polynomial** is one that is not a product of two other polynomials both of degree greater than 0.

e.g. $(x^2 + 2)$ for $Z_5[x]$

Analogous to a prime number.

Galois Fields

The polynomials

$Z_p[x] \bmod p(x)$

where

$p(x) \in Z_p[x]$,
p(x) is irreducible,
and $\deg(p(x)) = n$ (i.e. n+1 coefficients)

form a finite field. Such a field has p^n elements.

These fields are called **Galois Fields** or **$GF(p^n)$.**

The special case n = 1 reduces to the fields Z_p

The multiplicative group of $GF(p^n)\setminus\{0\}$ is cyclic (this will be important later).

GF(2^n)

Hugely practical!

The coefficients are **bits** {0,1}.

For example, the elements of GF(2^8) can be represented as **a byte**, one bit for each term, and GF(2^{64}) as **a 64-bit word**.

- *e.g.,* $x^6 + x^4 + x + 1 = 01010011$

How do we do addition?

Addition over Z_2 corresponds to xor.

- Just take the xor of the bit-strings (bytes or words in practice). This is dirt cheap

Multiplication over GF(2^n)

If n is small enough can use a table of all combinations.

The size will be $2^n \times 2^n$ (e.g. 64K for GF(2^8)).

Otherwise, use standard shift and add (xor)

Note: dividing through by the irreducible polynomial on an overflow by 1 term is simply a test and an xor.

e.g. 0111 / 1001 = 0111

1011 / 1001 = 1011 xor 1001 = 0010

^ just look at this bit for GF(2^3)

Multiplication over GF(2^n)

```
typedef unsigned char uc;

uc mult(uc a, uc b) {
  int p = a;
  uc r = 0;
  while(b) {
    if (b & 1) r = r ^ p;
    b = b >> 1;
    p = p << 1;
    if (p & 0x100) p = p ^ 0x11B;
  }
  return r;
}
```

Viewing Messages as Polynomials

A (n, k, n-k+1) code:

Consider the polynomial of degree k-1

$$p(x) = a_{k-1} x^{k-1} + \cdots + a_1 x + a_0$$

Message: $(a_{k-1}, \ldots, a_1, a_0)$

Codeword: $(p(1), p(2), \ldots, p(n))$

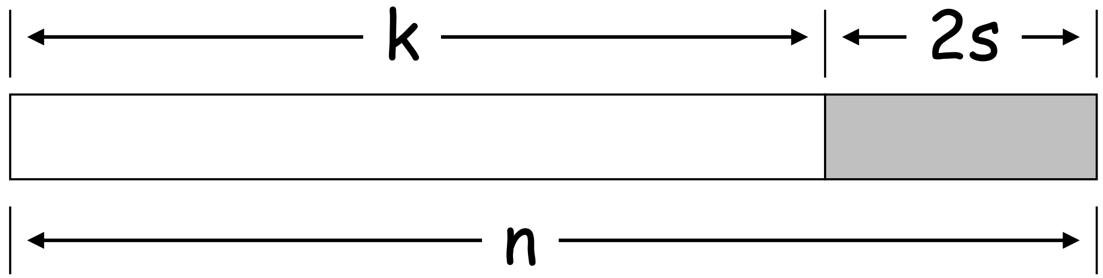
To keep the p(i) fixed size, we use $a_i \in GF(p^r)$

To make the i distinct, $n < p^r$

Unisolvence Theorem: Any subset of size k of (p(1), p(2), …, p(n)) is enough to (uniquely) reconstruct p(x) using polynomial interpolation, e.g., LaGrange's Formula.

Polynomial-Based Code

A (n, k, 2s +1) code:



Can **detect** 2s errors

Can **correct** s errors

Generally can correct α erasures and β errors if
$\alpha + 2\beta \leq 2s$

Correcting Errors

Correcting s errors:

1. Find k + s symbols that agree on a polynomial p(x). These must exist since originally k + 2s symbols agreed and only s are in error
2. There are no k + s symbols that agree on the wrong polynomial p'(x)
   - Any subset of k symbols will define p'(x)
   - Since at most s out of the k+s symbols are in error, p'(x) = p(x)

A Systematic Code

Systematic polynomial-based code

$$p(x) = a_{k-1} x^{k-1} + \cdots + a_1 x + a_0$$

Message: $(a_{k-1}, \ldots, a_1, a_0)$

Codeword: $(a_{k-1}, \ldots, a_1, a_0, p(1), p(2), \ldots, p(2s))$

This has the advantage that if we know there are no errors, it is trivial to decode.

The version of RS used in practice uses something slightly different than $p(1), p(2), \ldots$

This will allow us to use the "**Parity Check**" ideas from linear codes (i.e., $Hc^T = 0$?) to quickly test for errors.

Reed-Solomon Codes in the Real World

$(204,188,17)_{256}$: ITU J.83(A)[2]
$(128,122,7)_{256}$: ITU J.83(B)
$(255,223,33)_{256}$: Common in Practice
- Note that they are all byte based (i.e., symbols are from $GF(2^8)$).

Decoding rate on 1.8GHz Pentium 4:
- (255,251) = 89Mbps
- (255,223) = 18Mbps

Dozens of companies sell hardware cores that operate 10x faster (or more)
- (204,188) = 320Mbps (Altera decoder)

Applications of Reed-Solomon Codes

- **Storage**: CDs, DVDs, "hard drives",
- **Wireless**: Cell phones, wireless links
- **Sateline and Space**: TV, Mars rover, …
- **Digital Television**: DVD, MPEG2 layover
- **High Speed Modems**: ADSL, DSL, ..

Good at handling burst errors.

Other codes are better for random errors.

- e.g., Gallager codes, Turbo codes

RS and "burst" errors

Let's compare to Hamming Codes (which are "optimal").

	code bits	check bits
RS $(255, 253, 3)_{256}$	2040	16
Hamming $(2^{11}-1, 2^{11}-11-1, 3)_2$	2047	11

They can both correct 1 error, but not 2 random errors.

- The Hamming code does this with fewer check bits

However, RS can fix 8 contiguous bit errors in one byte

- Much better than lower bound for 8 arbitrary errors

$$\log\left(1+\binom{n}{1}+\cdots+\binom{n}{8}\right) > 8\log(n-7) \approx 88 \text{ check bits}$$

Galois Field

GF(2^3) with irreducible polynomial: $x^3 + x + 1$

$\alpha = x$ is a generator

α	x	010	**2**
α^2	x^2	100	**3**
α^3	$x + 1$	011	**4**
α^4	$x^2 + x$	110	**5**
α^5	$x^2 + x + 1$	111	**6**
α^6	$x^2 + 1$	101	**7**
α^7	1	001	**1**

Will use this as an example.

Discrete Fourier Transform (DFT)

Another View of polynomial-based codes

α is a primitive n^{th} root of unity ($\alpha^n = 1$) – a generator

$$T = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \\ 1 & \alpha^2 & \alpha^4 & \cdots & \alpha^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{n-1} & \alpha^{2(n-1)} & \cdots & \alpha^{(n-1)(n-1)} \end{pmatrix} \qquad \begin{pmatrix} c_0 \\ \vdots \\ c_{k-1} \\ c_k \\ \vdots \\ c_{n-1} \end{pmatrix} = T \cdot \begin{pmatrix} m_0 \\ \vdots \\ m_{k-1} \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Evaluate polynomial $m_{k-1}x^{k-1} + \cdots + m_1 x + m_0$
at n distinct roots of unity, $1, \alpha, \alpha^2, \alpha^3, \cdots, \alpha^{n-1}$

Inverse DFT: $m = T^{-1}c$

DFT Example

α = x is 7th root of unity in $GF(2^3)/x^3 + x + 1$

(i.e., multiplicative group, which excludes additive inverse)

Recall α = "2", α^2 = "3", … , α^7 = 1 = "1"

$$T = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \alpha & \alpha^2 & \alpha^3 & \alpha^4 & \alpha^5 & \alpha^6 \\ 1 & \alpha^2 & \alpha^4 & \alpha^6 & & & \\ 1 & \alpha^3 & \alpha^6 & & & & \\ 1 & \alpha^4 & & \ddots & & & \\ 1 & \alpha^5 & & & & & \\ 1 & \alpha^6 & & & & & \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 2^2 & 2^3 & 2^4 & 2^5 & 2^6 \\ 1 & 3 & 3^2 & 3^3 & & & \\ 1 & 4 & 4^2 & & & & \\ 1 & 5 & & \ddots & & & \\ 1 & 6 & & & & & \\ 1 & 7 & & & & & 7^6 \end{pmatrix}$$

Should be clear that $c = T \bullet (m_0, m_1, \ldots, m_{k-1}, 0, \ldots)^T$ is the same as evaluating $p(x) = m_0 + m_1 x + \ldots + m_{k-1} x^{k-1}$ at n points.

function fft(a,w,+,*) =

if #a == 1 then return a

Else

$w' = [w_0, w_2, \ldots, w_{n-1}]$

$e = fft([a_0, a_2, \ldots, a_{n-2}], w')$

$o = fft([a_1, a_3, \ldots, a_{n-1}], w')$

return $[e_0 + o_0 * w_0, e_1 + o_1 * w_1, \ldots, e_{n/2-1} + o_{n/2-1} * w_{n/2-1},$
$e_0 + o_0 * w_{n/2}, e_1 + o_1 * w_{n/2+1}, \ldots, e_{n/2-1} + o_{n/2-1} * w_{n-1}]$

Decoding

Why is it hard?

Brute Force: try $k+2s$ choose $k + s$ possibilities and solve for each.

Cyclic Codes

A linear code is cyclic if:

$$(c_0, c_1, \ldots, c_{n-1}) \in C \Rightarrow (c_{n-1}, c_0, \ldots, c_{n-2}) \in C$$

Both **Hamming** and **Reed-Solomon** codes are cyclic.

Note: we might have to reorder the columns to make the code "cyclic".

Motivation: They are more efficient to decode than general codes.

Generator and Parity Check Matrices

Generator Matrix:

A k x n matrix **G** such that:

$$C = \{m \bullet \mathbf{G} \mid m \in \Sigma^k\}$$

Made from stacking the basis vectors

Parity Check Matrix:

A (n – k) x n matrix **H** such that:

$$C = \{v \in \Sigma^n \mid H \bullet v^T = 0\}$$

Codewords are the nullspace of H

These **always exist for linear codes**

$H \bullet G^T = 0$

Generator and Parity Check Polynomials

Generator Polynomial:

A degree (n-k) polynomial **g** such that:

$$C = \{m \bullet \mathbf{g} \mid m \in m_0 + m_1 x + \ldots + m_{k-1} x^{k-1}\}$$

such that $\mathbf{g} \mid x^n - 1$

Parity Check Polynomial:

A degree k polynomial **h** such that:

$$C = \{v \in \Sigma^n [x] \mid h \bullet v = 0 \pmod{x^n - 1}\}$$

such that $\mathbf{h} \mid x^n - 1$

These **always exist for linear cyclic codes**

$h \bullet g = x^n - 1$

Viewing g as a matrix

If $g(x) = g_0 + g_1x + \ldots + g_{n-k-1}x^{n-k-1}$

We can put this generator in matrix form:

$$G = \begin{pmatrix} g_0 & g_1 & \cdots & \cdots & g_{n-k-1} & 0 & \cdots & 0 \\ 0 & g_0 & \cdots & \cdots & g_{n-k-2} & g_{n-k-1} & \cdots & 0 \\ \vdots & & \ddots & & & & \ddots & \vdots \\ 0 & 0 & \cdots & g_0 & g_1 & \cdots & \cdots & g_{n-k-1} \end{pmatrix}$$

Write $m = m_0 + m_1x + \ldots + m_{k-1}x^{k-1}$ as $(m_0, m_1, \ldots, m_{k-1})$

Then c = mG

g generates cyclic codes

$$G = \begin{pmatrix} g_0 & g_1 & \cdots & g_{n-k-1} & 0 & \cdots & 0 \\ 0 & g_0 & \cdots & g_{n-k-2} & g_{n-k-1} & \cdots & 0 \\ \vdots & & \ddots & & & \ddots & \vdots \\ 0 & 0 & \cdots & g_0 & g_1 & \cdots & g_{n-k-1} \end{pmatrix} = \begin{pmatrix} g \\ xg \\ \vdots \\ x^{k-1}g \end{pmatrix}$$

Codes are linear combinations of the rows.

All but last row is clearly cyclic (based on next row)

Shift of last row is $x^k g \bmod (x^n - 1) = g_{n-k-1}, 0, \ldots, g_0, g_1, \ldots, g_{n-k-2}$

Consider $h = h_0 + h_1 x + \ldots + h_{k-1} x^{k-1}$ $(gh = x^n - 1)$

$$h_0 g + (h_1 x)g + \ldots + (h_{k-2} x^{k-2})g + (h_{k-1} x^{k-1})g = x^n - 1$$

$$x^k g = -h_{k-1}^{-1}(h_0 g + h_1(xg) + \ldots + h_{k-1}(x^{k-1}g)) \bmod (x^n - 1)$$

This is a linear combination of the rows.

Viewing h as a matrix

If $h = h_0 + h_1 x + \ldots + h_{k-1} x^{k-1}$

we can put this parity check poly. in matrix form:

$$H = \begin{pmatrix} 0 & \cdots & 0 & h_{k-1} & \cdots & h_1 & h_0 \\ 0 & \cdots & h_{k-1} & h_{k-2} & \cdots & h_0 & 0 \\ \vdots & \iddots & & & \iddots & & \vdots \\ h_{k-1} & \cdots & h_1 & h_0 & 0 & \cdots & 0 \end{pmatrix}$$

$Hc^T = 0$

Hamming Codes Revisited

The Hamming $(7,4,3)_2$ code.

$$g = 1 + x + x^3 \qquad h = x^4 + x^2 + x + 1$$

$$G = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \qquad H = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 \end{pmatrix}$$

$$gh = x^7 - 1, \quad GH^T = 0$$

The columns are not identical to the previous example Hamming code.

Factors of $x^n - 1$

Intentionally left blank

Another way to write g

Let α be a **generator** of $GF(p^r)$.

Let $n = p^r - 1$ (the size of the multiplicative group)

Then we can write a generator polynomial as

$g(x) = (x-\alpha)(x-\alpha^2) \ldots (x - \alpha^{n-k})$, $h = (x- \alpha^{n-k+1})\ldots(x-\alpha^n)$

Lemma: $g \mid x^n - 1$, $h \mid x^n - 1$, $gh \mid x^n - 1$

($a \mid b$ means a divides b)

Proof:

- $\alpha^n = 1$ (because of the size of the group)
  $\Rightarrow \alpha^n - 1 = 0$
  $\Rightarrow \alpha$ root of $x^n - 1$
  $\Rightarrow (x - \alpha) \mid x^n - 1$
- similarly for $\alpha^2, \alpha^3, \ldots, \alpha^n$
- therefore $x^n - 1$ is divisible by $(x - \alpha)(x - \alpha^2) \ldots$

Back to Reed-Solomon

Consider a generator polynomial $g \in GF(p^r)[x]$, s.t. $g \mid (x^n - 1)$
Recall that $n - k = 2s$ (the degree of g is $n-k-1$, $n-k$ coefficients)

Encode:

- $m' = m\, x^{2s}$ (basically shift by 2s)
- $b = m' \pmod{g}$
- $c = m' - b = (m_{k-1}, \ldots, m_0, -b_{2s-1}, \ldots, -b_0)$
- Note that c is a **cyclic code** based on g
  - $m' = qg + b$
  - $c = m' - b = qg$

Parity check:

- $h\, c = 0$?

Example

Lets consider the $(7,3,5)_8$ Reed-Solomon code.
We use $GF(2^3)/x^3 + x + 1$

α	x	010	**2**
α^2	x^2	100	**3**
α^3	$x + 1$	011	**4**
α^4	$x^2 + x$	110	**5**
α^5	$x^2 + x + 1$	111	**6**
α^6	$x^2 + 1$	101	**7**
α^7	1	001	**1**

Example RS $(7,3,5)_8$

$n = 7, k = 3, n\text{-}k = 2s = 4, d = 2s{+}1 = 5$

$g = (x - \alpha)(x - \alpha^2)(x - \alpha^3)(x - \alpha^4)$
$= x^4 + \alpha^3 x^3 + x^2 + \alpha x + \alpha^3$

$h = (x - \alpha^5)(x - \alpha^6)(x - \alpha^7)$
$= x^3 + a^3 x^3 + a^2 x + a^4$

$gh = x^7 - 1$

Consider the message: 110 000 110

$m = (\boxed{\alpha^4, 0, \alpha^4}) = \alpha^4 x^2 + \alpha^4$

$m' = x^4 m = \alpha^4 x^6 + \alpha^4 x^4$
$= (\alpha^4 x^2 + x + \alpha^3)g + (\boxed{\alpha^3 x^3 + \alpha^6 x + \alpha^6})$

$c = (\boxed{\alpha^4, 0, \alpha^4}, \boxed{\alpha^3, 0, \alpha^6, \alpha^6})$
$= 110\ 000\ 110\ 011\ 000\ 101\ 101$

α	010
α^2	100
α^3	011
α^4	110
α^5	111
α^6	101
α^7	001

$ch = 0 \pmod{x^7 - 1}$

A useful theorem

Theorem: For any β, if $g(\beta) = 0$ then $\beta^{2s}m(\beta) = b(\beta)$

Proof:

$x^{2s}m(x) = m'(x) = g(x)q(x) + b(x)$

$\beta^{2s}m(\beta) = g(\beta)q(\beta) + b(\beta) = b(\beta)$

Corollary: $\beta^{2s}m(\beta) = b(\beta)$ for $\beta \in \{\alpha, \alpha^2, \alpha^3, \ldots, \alpha^{2s=n-k}\}$

Proof:

$\{\alpha, \alpha^2, \ldots, \alpha^{2s}\}$ are the roots of g by definition.

Fixing errors

Theorem: Any k symbols from c can reconstruct c and hence m

Proof:

We can write 2s equations involving m ($c_{n-1}, \ldots, c_{2s}$) and b ($c_{2s-1}, \ldots, c_0$). These are

$$\alpha^{2s}\, m(\alpha) = b(\alpha)$$

$$\alpha^{4s}\, m(\alpha^2) = b(\alpha^2)$$

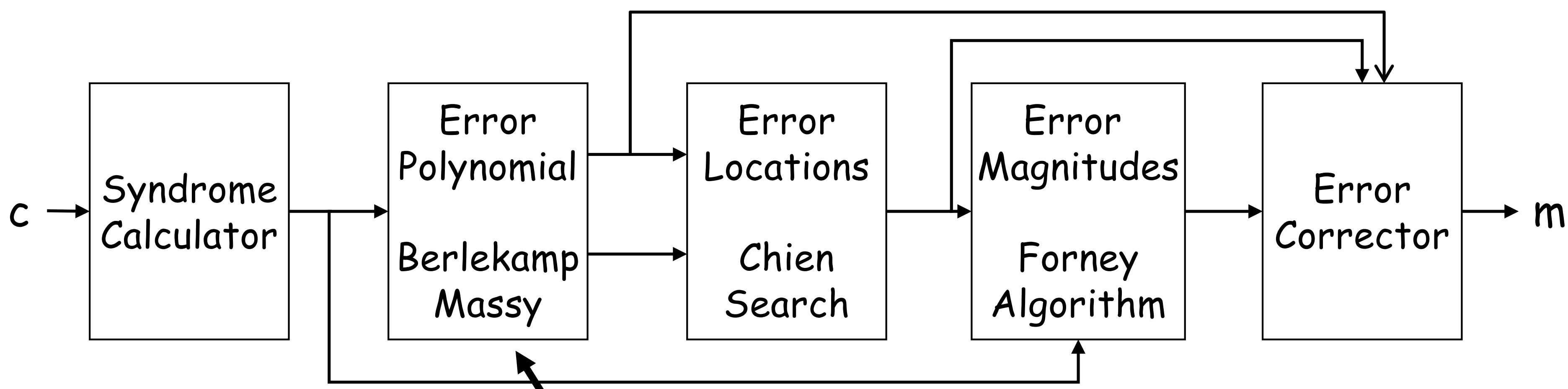
$$\ldots$$

$$\alpha^{2s(2s)}\, m(\alpha^{2s}) = b(\alpha^{2s})$$

We have at most 2s unknowns, so we can solve for them. (I'm skipping showing that the equations are linearly independent).

Efficient Decoding

I don't plan to go into the Reed-Solomon decoding algorithm, other than to mention the steps.



This is the hard part. CD players use this algorithm.
(Can also use Euclid's algorithm.)