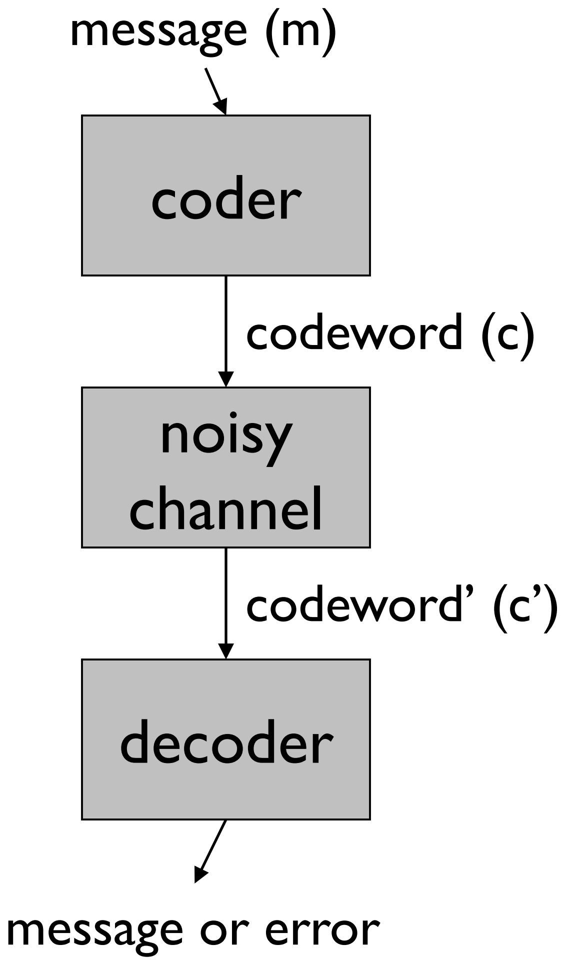
15-853:Algorithms in the Real World

- Reed Solomon Codes (Cont.)
- Concatenation of codes
- Start with LDPC codes

Announcements:

1. No class this Thursday, Sept. 19. Rescheduled to Friday, Sept 27.
2. Homework1 on ECC will be released on Tuesday Sept 24. Submission deadline Oct 4th **noon**.

Recap: Block Codes



Each message and codeword is of fixed size

Σ = codeword alphabet

k =|m| **n** = |c| **q** = $|\Sigma|$

C = "code" = set of codewords

C $\subseteq \Sigma^n$ (codewords)

Δ(x,y) = number of positions s.t. $x_i \neq y_i$

d = $\min\{\Delta(x,y) : x,y \in C, x \neq y\}$

Code described as: $\mathbf{(n,k,d)_q}$

Recap: Linear Codes

If Σ is a field, then Σ^n is a vector space

Definition: C is a linear code if it is a linear subspace of Σ^n of dimension k.

This means that there is a set of k independent vectors $v_i \in \Sigma^n$ $(1 \le i \le k)$ that span the subspace.

i.e. every codeword can be written as:

$$c = a_1 v_1 + a_2 v_2 + \ldots + a_k v_k \quad \text{where } a_i \in \Sigma$$

"Linear": linear combination of two codewords is a codeword.

Minimum distance = weight of least-weight codeword

Recap: Generator and Parity Check Matrices

Generator Matrix:

A k x n matrix **G** such that: $C = \{ xG \mid x \in \Sigma^k \}$

Made from stacking the spanning vectors

Parity Check Matrix:

An (n – k) x n matrix **H** such that: $C = \{y \in \Sigma^n \mid Hy^T = 0\}$

(Codewords are the null space of H.)

These **always exist for linear codes**

Recap: Singleton bound and MDS codes

Theorem: For every $(n , k, d)_q$ code, $n \geq k + d - 1$

Codes that meet Singleton bound with equality are called

Maximum Distance Separable (MDS)

Only two binary MDS codes!

1. Repetition codes
2. Single-parity check codes

Need to go beyond the binary alphabet!

(We will need some number theory for this)

Recap: Finite fields

- Size (or order): Prime or power of prime
- Power-of-prime finite fields:
  - Constructed using polynomials
  - Mod by irreducible polynomial
- Correspondence between polynomials and vector representation

Recap: GF(2^n)

$\mathbb{F}_{2^n}$ = set of polynomials in $\mathbb{F}_2[x]$ modulo irreducible polynomial $\mathrm{p}(x) \in \mathbb{F}_2[x]$ of degree n.

Elements are all polynomials in $\mathbb{F}_2[x]$ of degree $\leq n - 1$.

Has 2^n elements.

Natural correspondence with bits in $\{\mathbf{0}, \mathbf{1}\}^{\boldsymbol{n}}$.

Elements of $\mathbb{F}_{2^8}$ can be represented as a byte, one bit for each term.

E.g., $x^6 + x^4 + x + 1 = 01010011$

RS code: Polynomials viewpoint

Message: $[a_{k-1}, \ldots, a_1, a_0]$ where $a_i \in GF(q^r)$

Consider the polynomial of degree k-1

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

RS code:

Codeword: $[P(1), P(2), \ldots, P(n)]$

To make the i in p(i) distinct, need field size $q^r \geq n$

That is, need sufficiently large field size for desired codeword length.

Recap: Minimum distance of RS code

Theorem: RS codes have minimum distance d = n-k+1

Proof:

1. *RS is a linear code:* if we add two codewords corresponding to P(x) and Q(x), we get a codeword corresponding to the polynomial P(x) + Q(x). Similarly any linear combination..
2. *So look at the least weight codeword*. It is the evaluation of a polynomial of degree k-1 at some n points. So it can be zero on only k-1 points. Hence non-zero on at most (n-(k-1)) points. This means distance at least n-k+1
3. Apply Singleton bound

Meets Singleton bound: RS codes are MDS

Recap: Generator matrix of RS code

Q: What is the generator matrix?

<board>

“Vandermonde matrix”

Special property of Vandermonde matrices:

Full rank (columns linearly independent)

Vandermonde matrix: Very useful in constructing codes.

Next we move on to RS decoding

Polynomials and their degrees

Fundamental theorem of Algebra:

Any non-zero polynomial of degree k has at most k roots (over any field).

Corollary 1:

If two degree-k polynomials P, Q agree on k+1 locations (i.e., if $P(x_i) = Q(x_i)$ for $x_0, x_1, \ldots, x_k$), then P = Q.

Corollary 2:

Given any k+1 points (x_i, y_i), there is at most one degree-k polynomial that has $P(x_i) = y_i$ for all these i.

Polynomials and their degrees

Corollary 2:

Given any k+1 points (x_i, y_i), there is at most one degree-k polynomial that has $P(x_i) = y_i$ for all these i.

Theorem:

Given any k+1 points (x_i, y_i), there is exactly one degree-k polynomial that has $P(x_i) = y_i$ for all these i.

Proof: e.g., use Lagrange interpolation.

Decoding: Recovering Erasures

Recovering from at most (d-1) erasures:

Received codeword:

[P(1), *, …, *, P(n)]: at most (d-1) symbols erased

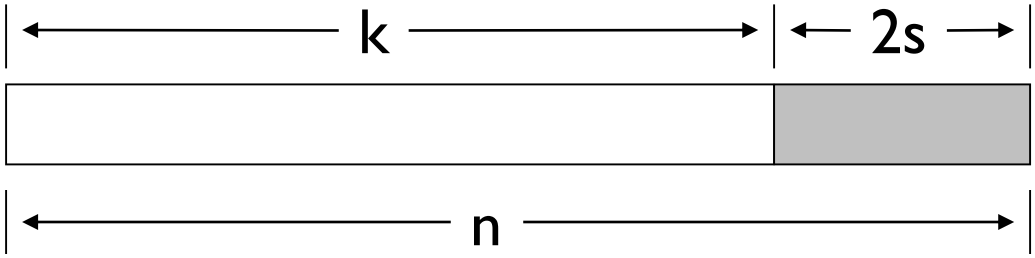
Ideas?

1. At most n-k symbols erased
2. So have p(i) for at least k evaluations
3. Interpolation to recover the polynomial

Matrix viewpoint: ideas?

RS Code

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



Can **detect** 2s errors

Can **correct** s errors

Generally can correct a erasures and b errors if
$a + 2b \leq 2s$

Decoding: Correcting Errors

Correcting s errors: (d = 2s+1)

Naïve algo:

- Find k+s symbols that agree on a degree (k-1) poly P(x).
  - There must exist one: since originally k + 2s symbols agreed and at most s are in error

    (i.e., “guess” the n-s uncorrupted locations)
- Can we go wrong?

  Are there k+s symbols that agree on the wrong degree (k-1) polynomial P’(x)?

  No.
  - 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)

Decoding: Correcting Errors

Correcting s errors: (d = 2s+1)

Naïve algo:

- Find k+s symbols that agree on a degree (k-1) poly P(x).
  - There must exist one: since originally k + 2s symbols agreed and at most s are in error

    (i.e., "guess" the n-s uncorrupted locations)

But this suggests a brute-force approach, very inefficient.

"guess" = "enumerate", so time is (n choose s) ~ n^s.

More efficient algorithms exist.

The Berlekamp Welch Algorithm

Say we sent $c_i = P(i)$ for $i = 1..n$

Received c_i' where $c_i = c_i'$ for all but s locations.

Let S be the set of these s error locations.

Suppose we magically know "error-locator" polynomial $E(x)$

such that $E(x) = 0$ for all x in S.

And $E(x)$ has degree s.

Does such a thing exist?

Sure. $E(x) = \prod_{a\ in\ S}(x - a)$

The Berlekamp Welch Algorithm

Say we sent $c_i = P(i)$ for $i = 1..n$

Received c_i' where $c_i = c_i'$ for all but s locations.

Let S be the set of these s error locations.

Suppose we magically know "error-locator" polynomial $E(x)$ such that $E(x) = 0$ for all x in S.

And $E(x)$ has degree s.

Then we know that

$$P(i) \cdot E(i) = c_i' \cdot E(i) \qquad \text{for all } i \text{ in } 1..n$$

The Berlekamp Welch Algorithm

Know that

$P(i) \cdot E(i) = c_i' \cdot E(i)$ for all i *in* $1..n$

Want to solve for polys $P(x)$ (of deg $k-1$), $E(x)$ of deg s.

How? First, rewrite as:

$\mathrm{R}(i) = c_i' \cdot E(i)$ for all i *in* $1..n$

for polynomials R of degree (k+s-1), E of degree s.

R has k+s "degrees of freedom". E has s+1.

Have n equalities.

So perhaps can get solution if $(k+s)+(s+1) \geq n$.

Return $\frac{R(x)}{E(x)}$.

The current situation

We know that

$$R(i) = c_i' \cdot E(i) \qquad \text{for all } i \; in \; 1..n$$

Suppose $\mathrm{R(x)} = \sum_{j=1..k+s-1} r_j \, x^j$

$k + s$ unknowns (the r_i values)

And $E(x) = \sum_{j=0..s} e_j \, x^j$

$s + 1$ unknowns (the e_i values)

How to solve for $R(x), E(x)$?

The linear system

Linear equalities

$$r_0 + r_1 \cdot 1 + r_2 \cdot 1^2 + \ \dots + r_{k+s-1} 1^{k+s-1} = c'_1 \cdot (e_0 + e_1 \cdot 1 + \cdots + e_s 1^s)$$

$$r_0 + r_1 \cdot 2 + r_2 \cdot 2^2 + \ \dots + r_{k+s-1} 2^{k+s-1} = c'_2 \cdot (e_0 + e_1 \cdot 2 + \cdots + e_s 2^s)$$

$$\dots$$

$$r_0 + r_1 \cdot i + r_2 \cdot i^2 + \ \dots + r_{k+s-1} i^{k+s-1} = c'_i \cdot (e_0 + e_1 \cdot i + \cdots + e_s i^s)$$

$$\dots$$

$$r_0 + r_1 \cdot n + r_2 \cdot n^2 + \ \dots + r_{k+s-1} n^{k+s-1} = c'_n \cdot (e_0 + e_1 \cdot n + \cdots + e_s n^s)$$

- Linearly independent equalities. Why?
  (Vandermonde structure.)
- Under-constrained. Why?
  n equations, (k+s)+(s+1) = n+1 variables.
- Can have multiple solutions. Problem?
  - <board>

RS and "burst" bit errors

Let's compare to Hamming Codes.

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

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

CONCATENATION OF CODES

Concatenation of Codes

Take a RS code $(n,k,n-k+1)_q$ code.

Can encode each alphabet symbol using another code.

David Forney

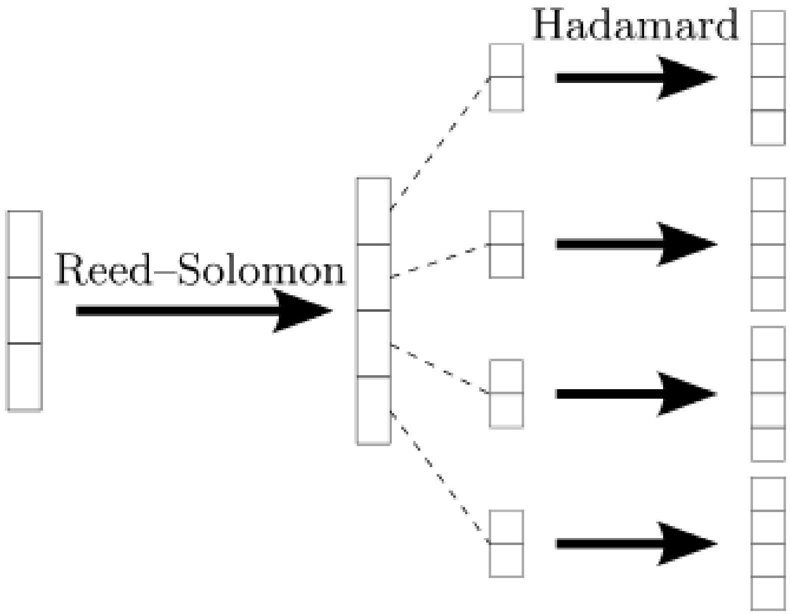


Image credit: Wikipedia

Concatenation of Codes

Take any $(N, K, D)_{q^k}$ code.

Can encode each alphabet symbol of k bits using another $(n, k, d)_q$ code.

Theorem:

The concatenated code is a $(Nn, Kk, Dd)_q$ code

Proof:

<Discuss>

Concatenation of Codes

Take a RS code $(n,k,n-k+1)_{Q=q^{k'}}$ code with $n = q^{k'}$.

Can encode each alphabet symbol of k' = log Q = log n bits using another code.

E.g., use $((k' + \log k'), k', 3)_2$-Hamming code. Now we can correct one error per alphabet symbol with little rate loss. (Good for sparse periodic errors.)

Or $(2^{k'}, k', 2^{k'-1})_2$ Hadamard code. (Say k = n/2.) Then get $(n^2, (n/2) \log n, n^2/4)_2$ code. Much better than plain Hadamard code in rate, distance worse only by factor of 2.