

Lecture 10: Error Correcting Codes

WCaT Can You 2o Cith A Noisy ChahDel?

Sema-Five?

Suppose I am trying to communicate via semaphore:

Sema-Five?

Suppose I am trying to communicate via semaphore:

What if my recipient misses some letters?

Sema-Five?

Suppose I am trying to communicate via semaphore:

What if my recipient misses some letters?

Less silly: deal with dropped internet packets

Problem Statement

Formally: have message in n parts $\{c_1, \dots, c_n\}$
Channel may drop up to W packets sent
How many packets needed to ensure receipt?

Problem Statement

Formally: have message in n parts $\{c_1, \dots, c_n\}$
Channel may drop up to W packets sent
How many packets needed to ensure receipt?

Naïve idea: repetition coding

- I Repeat message “enough times”

Problem Statement

Formally: have message in n parts $\{c_1, \dots, c_n\}$
Channel may drop up to W packets sent
How many packets needed to ensure receipt?

Naïve idea: repetition coding

- 1 Repeat message “enough times”

How many reps required to $L \sim q^n$ receipt?

Problem Statement

Formally: have message in n parts $\{c_1, \dots, c_n\}$
Channel may drop up to W packets sent
How many packets needed to ensure receipt?

Naïve idea: repetition coding

- 1 Repeat message “enough times”

How many reps required to receive n packets?
Could drop first packet every time!
Need $W + 1$ repetitions to be safe

Problem Statement

Formally: have message in n parts $\{c_1, \dots, c_n\}$
Channel may drop up to W packets sent
How many packets needed to ensure receipt?

Naïve idea: repetition coding

- 1 Repeat message “enough times”

How many reps required to receive n packets?
Could drop first packet every time!
Need $W + 1$ repetitions to be safe

For n packet message, send $n(W + 1)$ packets

Problem Statement

Formally: have message in n parts $\{c_1, \dots, c_n\}$
Channel may drop up to W packets sent
How many packets needed to ensure receipt?

Naïve idea: repetition coding

- 1 Repeat message “enough times”

How many reps required to receive n packets?
Could drop first packet every time!
Need $W + 1$ repetitions to be safe

For n packet message, send $n(W + 1)$ packets
Can we do better?

A Better Encoding

; **YS** : Can get away with $\hat{+}$ + **W**ackets

A Better Encoding

; **YS** : Can get away with $\hat{+}$ + **W**ackets

How?

A Better Encoding

; **YS** : Can get away with $\hat{+}$ + **W**ackets

How? Using polynomials!

A Better Encoding

; **YS** : Can get away with $\hat{+} + W$ packets

How? Using polynomials!

Idea: Take prime I st $I > \hat{+} + W >$ largest message
Encode message as polynomial in $KQ(I)$

A Better Encoding

; \mathbf{YS} : Can get away with $\hat{+} + \mathbf{W}$ packets

How? Using polynomials!

Idea: Take prime I st $I > \hat{+} + \mathbf{W} >$ largest message

Encode message as polynomial in $KQ(I)$

Interpolate poly $e(\hat{+})$ st $e(\mathbf{S} = \setminus_s \text{for } 1 \quad \mathbf{S} \quad \hat{+}$

Send $e(1), e(2), \dots, e(\hat{+} + \mathbf{W})$

Recovery

; **YS** : With ~~W~~erasures, recovery always possible

Recovery

; **YS** : With W erasures, recovery always possible

dpbH

- I Suppose receive $\hat{\cdot}$ points
- I Interpolate poly $e^{\hat{\cdot}}(\hat{\cdot})$ through them

Recovery

; **YS** : With W erasures, recovery always possible

dpbH

- I Suppose receive \hat{n} points
- I Interpolate poly $e^\ell(\hat{f})$ through them
- I $\deg(e) = \deg(e^\ell) = \hat{n} - 1$
- I e and e^ℓ agree on \hat{n} points

Recovery

; **YS** : With W erasures, recovery always possible

dpbH

- I Suppose receive \hat{e} points
- I Interpolate poly $e^\ell(\hat{e})$ through them
- I $\deg(e) = \deg(e^\ell) = \hat{n} - 1$
- I e and e^ℓ agree on \hat{n} points
- I So $e = e^\ell$

Recovery

; **YS** : With W erasures, recovery always possible

dpbH

- I Suppose receive \hat{e} points
- I Interpolate poly $e^\ell(\hat{e})$ through them
- I $\deg(e) = \deg(e^\ell) = \hat{n} - 1$
- I e and e^ℓ agree on \hat{n} points
- I So $e = e^\ell$
- I Thus $\forall s = e^\ell(S)$ for $1 \leq S \leq \hat{n}$

Encoding Example

Want to send $\backslash = (4;0;5)$, protect for 2 erasures

Encoding Example

Want to send $\mathbf{v} = (4;0;5)$, protect for 2 erasures

Interpolate polynomial modulo 7:

Encoding Example

Want to send $\mathbf{v} = (4; 0; 5)$, protect for 2 erasures

Interpolate polynomial modulo 7:

$$c(\mathcal{I}) = (\mathcal{I} - 2)(\mathcal{I} - 3)[(1 - 2)(1 - 3)]^{-1} \mathbf{c}$$
$$4(\mathcal{I}^2 - 5\mathcal{I} + 6) \quad 4\mathcal{I}^2 + \mathcal{I} + 3 \pmod{7}$$

Encoding Example

Want to send $\mathbf{v} = (4; 0; 5)$, protect for 2 erasures

Interpolate polynomial modulo 7:

$$c(\mathcal{I}) = (\mathcal{I} - 2)(\mathcal{I} - 3)[(1 - 2)(1 - 3)]^{-1} c$$
$$4(\mathcal{I}^2 - 5\mathcal{I} + 6) = 4\mathcal{I}^2 + \mathcal{I} + 3 \pmod{7}$$

Don't need to calculate $(\mathcal{I})!$

Encoding Example

Want to send $\mathbf{v} = (4; 0; 5)$, protect for 2 erasures

Interpolate polynomial modulo 7:

$$c(\mathcal{I}) = (\mathcal{I} - 2)(\mathcal{I} - 3)[(1 - 2)(1 - 3)]^{-1} c \\ 4(\mathcal{I} - 5\mathcal{I} + 6) = 4\mathcal{I}^2 + \mathcal{I} + 3 \pmod{7}$$

Don't need to calculate $(\mathcal{I})!$

$$\{c(\mathcal{I}) = (\mathcal{I} - 1)(\mathcal{I} - 2)[(3 - 1)(3 - 2)]^{-1} c \\ 4(\mathcal{I} - 3\mathcal{I} + 2) = 4\mathcal{I}^2 + 2\mathcal{I} + 1 \pmod{7}$$

Encoding Example

Want to send $\mathbf{v} = (4; 0; 5)$, protect for 2 erasures

Interpolate polynomial modulo 7:

$$c(\mathcal{I}) = (\mathcal{I} - 2)(\mathcal{I} - 3)[(1 - 2)(1 - 3)]^{-1} c$$

$$4(\mathcal{I}^2 - 5\mathcal{I} + 6) \quad 4\mathcal{I}^2 + \mathcal{I} + 3 \pmod{7}$$

Don't need to calculate $(\mathcal{I})!$

$$\{(\mathcal{I}) = (\mathcal{I} - 1)(\mathcal{I} - 2)[(3 - 1)(3 - 2)]^{-1} c$$

$$4(\mathcal{I}^2 - 3\mathcal{I} + 2) \quad 4\mathcal{I}^2 + 2\mathcal{I} + 1 \pmod{7}$$

$$e(\mathcal{I}) = 4 c(\mathcal{I}) + 5 \{(\mathcal{I}) \quad \mathcal{I}^2 + 3 \pmod{7}$$

Encoding Example

Want to send $\mathbf{v} = (4; 0; 5)$, protect for 2 erasures

Interpolate polynomial modulo 7:

$$c(\mathcal{I}) = (\mathcal{I} - 2)(\mathcal{I} - 3)[(1 - 2)(1 - 3)]^{-1} \\ 4(\mathcal{I}^2 - 5\mathcal{I} + 6) \quad 4\mathcal{I}^2 + \mathcal{I} + 3 \pmod{7}$$

Don't need to calculate $(\mathcal{I})!$

$$\{(\mathcal{I}) = (\mathcal{I} - 1)(\mathcal{I} - 2)[(3 - 1)(3 - 2)]^{-1} \\ 4(\mathcal{I}^2 - 3\mathcal{I} + 2) \quad 4\mathcal{I}^2 + 2\mathcal{I} + 1 \pmod{7}$$

$$e(\mathcal{I}) = 4 c(\mathcal{I}) + 5 \{(\mathcal{I}) \quad \mathcal{I}^2 + 3 \pmod{7}$$

Send $(e(1); e(2); e(3); e(4); e(5)) = (4; 0; 5; 5; 0)$

Recovery Example

Sent: (4;0;5;5;0); Received: (;0;5;5;)

Recovery Example

Sent: (4;0;5;5;0); Received: (;0;5;5;)
Need to interpolate!

Recovery Example

Sent: (4;0;5;5;0); Received: (;0;5;5;)

Need to interpolate!

Don't need | (~~4~~)!

Recovery Example

Sent: (4;0;5;5;0); Received: (;0;5;5;)

Need to interpolate!

Don't need $(\hat{t})!$

$$\{(\hat{t}) = (\hat{t} - 2)(\hat{t} - 4)[(3 - 2)(3 - 4)]^c$$
$$6(\hat{t} - 6\hat{t} + 8) \quad 6\hat{t} + 6\hat{t} + 6$$

Recovery Example

Sent: (4;0;5;5;0); Received: (;0;5;5;)

Need to interpolate!

Don't need $|(\ddagger)|$!

$$\begin{matrix} \{(\ddagger) = (\ddagger - 2)(\ddagger - 4)[(3 - 2)(3 - 4)]^c \\ 6(\ddagger^| - 6\ddagger + 8) & 6\ddagger^| + 6\ddagger + 6 \end{matrix}$$

$$\begin{matrix} \text{J}(\ddagger) = (\ddagger - 2)(\ddagger - 3)[(4 - 2)(4 - 3)]^c \\ 4(\ddagger^| - 5\ddagger + 6) & 4\ddagger^| + \ddagger + 3 \end{matrix}$$

Recovery Example

Sent: (4;0;5;5;0); Received: (;0;5;5;)

Need to interpolate!

Don't need $|(\hat{t})!$

$$\begin{aligned} \{(\hat{t}) &= (\hat{t} \ 2)(\hat{t} \ 4)[(3 \ 2)(3 \ 4)]^c \\ &6(\hat{t}^| \ 6\hat{t} + 8) \quad 6\hat{t}^| + 6\hat{t} + 6 \end{aligned}$$

$$\begin{aligned} J(\hat{t}) &= (\hat{t} \ 2)(\hat{t} \ 3)[(4 \ 2)(4 \ 3)]^c \\ &4(\hat{t}^| \ 5\hat{t} + 6) \quad 4\hat{t}^| + \hat{t} + 3 \end{aligned}$$

Interpolate $e^{\hat{t}}(\hat{t}) = 5 \{(\hat{t}) + 5 J(\hat{t}) \hat{t}^| + 3$

Recovery Example

Sent: (4;0;5;5;0); Received: (;0;5;5;)

Need to interpolate!

Don't need $|(\#)$!

$$\begin{aligned} \{(\#) &= (\# - 2)(\# - 4)[(3 - 2)(3 - 4)]^{-1} \\ &= 6(\#^2 - 6\# + 8) = 6\#^2 + 6\# + 6 \end{aligned}$$

$$\begin{aligned} J(\#) &= (\# - 2)(\# - 3)[(4 - 2)(4 - 3)]^{-1} \\ &= 4(\#^2 - 5\# + 6) = 4\#^2 + \# + 3 \end{aligned}$$

Interpolate $e^{\#}(\#) = 5 \{(\#) + 5 J(\#) - \#^2 + 3$

Evaluate for message: $(e^{\#}(1); e^{\#}(2); e^{\#}(3)) = (4; 0; 5)$

Optimality

; **YS** : Can't guarantee success w/ $< \hat{+}$ W packets

Optimality

; **YS** : Can't guarantee success w/ $< \lambda + W$ packets

dpbH

I May send one of two messages:

I $(\setminus c; \setminus |; \dots; \setminus ^ c; \setminus ^)$ or

I $(\setminus c; \setminus |; \dots; \setminus ^ c; \setminus \emptyset)$

Optimality

; **YS** : Can't guarantee success w/ $< \hat{\alpha} + W$ packets

dpbH

I May send one of two messages:

I $(\setminus_c; \setminus_1; \dots; \setminus^{\wedge}_c; \setminus^{\wedge})$ or

I $(\setminus_c; \setminus_1; \dots; \setminus^{\wedge}_c; \setminus^{\emptyset})$

I Channel drops $\hat{\alpha}$ th packet and all extras

Optimality

; **YS** : Can't guarantee success w/ $< \hat{n} + W$ packets

dpbH

- I May send one of two messages:
 - I $(\setminus_c; \setminus_1; \dots; \setminus^{\wedge}_c; \setminus^{\wedge})$ or
 - I $(\setminus_c; \setminus_1; \dots; \setminus^{\wedge}_c; \setminus^{\emptyset})$
- I Channel drops \hat{n} th packet and all extras
- I Which message was sent?

Optimality

; **YS** : Can't guarantee success w/ $< \hat{n} + W$ packets

dpbH

- I May send one of two messages:
 - I $(\setminus_c; \setminus_1; \dots; \setminus^{\wedge}_c; \setminus^{\wedge})$ or
 - I $(\setminus_c; \setminus_1; \dots; \setminus^{\wedge}_c; \setminus^{\emptyset})$
- I Channel drops \hat{n} th packet and all extras
- I Which message was sent?
- I Impossible to know!

Corruption Errors

More difficult: what if packets are corrupted?

C0rrupt1on Err0rs

More difficult: what if packets are corrupted?
Don't know which packets are wrong!

C0rrupt1on Err0rs

More difficult: what if packets are corrupted?

Don't know which packets are wrong!

; **YS** : Previous encoding not good enough

Corruption Errors

More difficult: what if packets are corrupted?
Don't know which packets are wrong!

; **YS** : Previous encoding not good enough

dpbH

I Again, two possible original messages:

I $(\setminus_c; \dots; \setminus^{\wedge} c; \setminus^{\wedge})$ or

I $(\setminus_c; \dots; \setminus^{\wedge} c; \setminus^{\theta})$

Corruption Errors

More difficult: what if packets are corrupted?
Don't know which packets are wrong!

; **YS** : Previous encoding not good enough

dpbH

- | Again, two possible original messages:
 - | $(\setminus_c; \dots; \setminus^{\wedge}_c; \setminus^{\wedge})$ or
 - | $(\setminus_c; \dots; \setminus^{\wedge}_c; \setminus^{\theta}_{\wedge})$
- | First \wedge rec'd match 1st, but next **W** match 2nd

Corruption Errors

More difficult: what if packets are corrupted?
Don't know which packets are wrong!

; **YS** : Previous encoding not good enough

dpbH

- I Again, two possible original messages:
 - I $(\setminus_c; \dots; \setminus^{\wedge}_c; \setminus^{\wedge})$ or
 - I $(\setminus_c; \dots; \setminus^{\wedge}_c; \setminus^{\theta}_{\wedge})$
- I First \wedge rec'd match 1st, but next **W** match 2nd
- I Which message was sent?

Corruption Errors

More difficult: what if packets are corrupted?
Don't know which packets are wrong!

; **YS** : Previous encoding not good enough

dpbH

- I Again, two possible original messages:
 - I $(\setminus_c; \dots; \setminus^{\wedge}_c; \setminus^{\wedge})$ or
 - I $(\setminus_c; \dots; \setminus^{\wedge}_c; \setminus^{\theta}_{\wedge})$
- I First \wedge rec'd match 1st, but next **W** match 2nd
- I Which message was sent?
- I Impossible to know!

Corruption Errors

More difficult: what if packets are corrupted?
Don't know which packets are wrong!

; **YS** : Previous encoding not good enough

dpbH

- | Again, two possible original messages:
 - | $(\setminus_c; \dots; \setminus^{\wedge}_c; \setminus^{\wedge})$ or
 - | $(\setminus_c; \dots; \setminus^{\wedge}_c; \setminus^{\theta}_{\wedge})$
- | First \wedge rec'd match 1st, but next **W** match 2nd
- | Which message was sent?
- | Impossible to know!

Note: works for - \wedge %padding by **W** packets

NEED MOAR PACKETS

γ_{PCBC}^{Δ} : For W corruptions, need $\gamma + 2W$ packets

NEED MOAR PACKETS

yPCbcA : For W corruptions, need $W + 2W$ packets

Suppose only send $2W + 1$ extra packets

NEED MOAR PACKETS

$\mathbf{y}^{\mathbf{C}} : \text{For } W \text{ corruptions, need } \hat{\mathbf{y}} + 2W \text{ packets}$

Suppose only send $2W + 1$ extra packets

Consider two possible messages:

$$(\mathbf{y}_c; \mathbf{y}_1; \dots; \mathbf{y}_c; \mathbf{y}_1; \mathbf{C}_c; \dots; \mathbf{C}_{Wc}; \mathbf{C}_W; \dots; \mathbf{C}_{Wc})$$

$$(\mathbf{y}_c; \mathbf{y}_1; \dots; \mathbf{y}_c; \mathbf{y}_1; \mathbf{C}_c^0; \dots; \mathbf{C}_{Wc}^0; \mathbf{C}_W^0; \dots; \mathbf{C}_{Wc}^0)$$

NEED MOAR PACKETS

$y_{\text{PCBC}}^{\Lambda}$: For W corruptions, need $W + 2W$ packets

Suppose only send $2W + 1$ extra packets

Consider two possible messages:

$$(\lambda_c; \lambda_1; \dots; \lambda_c; \lambda; C_c; \dots; C_{Wc}; C_W; \dots; C_{Wc})$$

$$(\lambda_c; \lambda_1; \dots; \lambda_c; \lambda^0; C_c^0; \dots; C_{Wc}^0; C_W^0; \dots; C_{Wc}^0)$$



$$(\lambda_c; \lambda_1; \dots; \lambda_c; \lambda; C_c; \dots; C_{Wc}; C_W^0; \dots; C_{Wc}^0)$$

NEED MOAR PACKETS

$y_{\text{PCBC}}^{\Lambda}$: For W corruptions, need $W + 2W$ packets

Suppose only send $2W + 1$ extra packets

Consider two possible messages:

$$(\lambda_c; \lambda_1; \dots; \lambda_c; \lambda; C_c; \dots; C_{Wc}; \boxed{C_W; \dots; C_{|Wc}})$$

$$(\lambda_c; \lambda_1; \dots; \lambda_c; \lambda^0; C_c^0; \dots; C_{Wc}^0; C_W^0; \dots; C_{|Wc}^0)$$



$$(\lambda_c; \lambda_1; \dots; \lambda_c; \lambda; C_c; \dots; C_{Wc}; C_W^0; \dots; C_{|Wc}^0)$$

NEED MOAR PACKETS

$y_{\text{PCBC}}^{\Lambda}$: For W corruptions, need $W + 2W$ packets

Suppose only send $2W + 1$ extra packets

Consider two possible messages:

$$(\lambda_c; \lambda_1; \dots; \lambda_{W+c}; \lambda_{W+c}; C_c; \dots; C_{W+c}; \boxed{C_W; \dots; C_{W+c}})$$

$$(\lambda_c; \lambda_1; \dots; \lambda_{W+c}; \boxed{\lambda_{W+c}^0; C_c^0; \dots; C_{W+c}^0}; C_W^0; \dots; C_{W+c}^0)$$



$$(\lambda_c; \lambda_1; \dots; \lambda_{W+c}; \lambda_{W+c}; C_c; \dots; C_{W+c}; C_W^0; \dots; C_{W+c}^0)$$

NEED MOAR PACKETS

$y_{\text{Ch}}^{\text{C}} : \text{For } W \text{ corruptions, need } \wedge + 2W \text{ packets}$

Suppose only send $2W + 1$ extra packets

Consider two possible messages:

$$(\backslash_c; \backslash_{|}; \dots; \backslash^{\wedge}_c; \backslash^{\wedge}; C_c; \dots; C_{Wc}; \boxed{C_W; \dots; C_{|Wc}})$$

$$(\backslash_c; \backslash_{|}; \dots; \backslash^{\wedge}_c; \boxed{\backslash^{\circ}_c; C_c; \dots; C_{Wc}}; C_W; \dots; C_{|Wc})$$



$$(\backslash_c; \backslash_{|}; \dots; \backslash^{\wedge}_c; \backslash^{\wedge}; C_c; \dots; C_{Wc}; C_W; \dots; C_{|Wc})$$

Don't know which message originally sent!

Relaaaaax

Take a 4 minute break!

Relaaaaax

Take a 4 minute break!

yb@ %s ? S<~ssb^ k~Cszb^:

What's your strangest family tradition?

Corruption Recovery

$y = Cx$: If use previous encoding with $2W$ extra packets, can recover from W corruptions.

Corruption Recovery

$y = Cx$: If use previous encoding with $2W$ extra packets, can recover from W corruptions.

How?

Corruption Recovery

$y = Pc$: If use previous encoding with $2W$ extra packets, can recover from W corruptions.

How? Find $\deg \leq 1$ poly through $2W$ points

Corruption Recovery

PCBC: If use previous encoding with $2W$ extra packets, can recover from W corruptions.

How? Find $\deg \leq 1$ poly through $n + W$ points

; **YS**: Such a poly exists

Corruption Recovery

Prop: If use previous encoding with $2W$ extra packets, can recover from W corruptions.

How? Find $\deg \leq 1$ poly through $n + W$ points

; **YS**: Such a poly exists

- | Original poly through $n + W$ uncorrupted points

Corruption Recovery

Prop: If use previous encoding with $2W$ extra packets, can recover from W corruptions.

How? Find $\deg \leq 1$ poly through $n + W$ points

; **YS**: Such a poly exists

┆ Original poly through $n + W$ uncorrupted points

; **YS**: Only one such poly

Corruption Recovery

Prop: If use previous encoding with $2W$ extra packets, can recover from W corruptions.

How? Find $\deg \leq 1$ poly through $n + W$ points

; **YS**: Such a poly exists

- | Original poly through $n + W$ uncorrupted points

; **YS**: Only one such poly

- | For any $n + W$ points, at least n uncorrupted

- | Those n define the original polynomial

Efficiency?

How long does it take to recover?

Efficiency?

How long does it take to recover?

Naïvely, need to try all possible sets of W corruptions

Efficiency?

How long does it take to recover?

Naïvely, need to try all possible sets of W corruptions
 $\binom{W}{w}$ possibilities — much too slow

Efficiency?

How long does it take to recover?

Naïvely, need to try all possible sets of W corruptions
 $\binom{W}{w}$ possibilities — much too slow

State-of-the-art for over 25 years! (1960 - 1986)

Efficiency?

How long does it take to recover?

Naïvely, need to try all possible sets of W corruptions
 $\binom{W}{w}$ possibilities — much too slow

State-of-the-art for over 25 years! (1960 - 1986)

Elwyn Berlekamp

Lloyd Welch

Efficiency?

How long does it take to recover?

Naïvely, need to try all possible sets of W corruptions
 $\binom{W}{w}$ $\left(\binom{W}{w}\right)^W$ possibilities — much too slow

State-of-the-art for over 25 years! (1960 - 1986)

Elwyn Berlekamp

Lloyd Welch

Berlekamp-Welch Recovery

Main idea: have (unknown) error-location poly

$$Q(x) = (x - C_0)(x - C_1) \cdots (x - C_w)$$

Berlekamp-Welch Recovery

Main idea: have (unknown) error-location poly

$$Q(x) = (x - C_1)(x - C_2)\cdots(x - C_w)$$

If can find this poly, can fix corruptions!

Berlekamp-Welch Recovery

Main idea: have (unknown) error-location poly

$$Q(x) = (x - \alpha_1)(x - \alpha_2) \cdots (x - \alpha_w)$$

If can find this poly, can fix corruptions!

Define (unknown) $L(x) = e(x)Q(x)$ to help solve

Berlekamp-Welch Recovery

Main idea: have (unknown) error-location poly

$$Q(x) = (x - C_1)(x - C_2) \cdots (x - C_w)$$

If can find this poly, can fix corruptions!

Define (unknown) $L(x) = e(x)Q(x)$ to help solve

Claim: $L(S) = q(S)$ for all S

Berlekamp-Welch Recovery

Main idea: have (unknown) error-location poly

$$Q(x) = (x - C_1)(x - C_2)\cdots(x - C_w)$$

If can find this poly, can fix corruptions!

Define (unknown) $L(x) = e(x)Q(x)$ to help solve

Claim: $L(S) = e(S)Q(S)$ for all S

- | If S error, both sides zero
- | Otherwise $e(S) = L(S)/Q(S)$, so true by definition

Berlekamp-Welch Recovery

Main idea: have (unknown) error-location poly

$$Q(x) = (x - C_0)(x - C_1) \cdots (x - C_w)$$

If can find this poly, can fix corruptions!

Define (unknown) $I(x) = e(x)Q(x)$ to help solve

Claim: $I(S) = e(S)Q(S)$ for all S

- | If S error, both sides zero
- | Otherwise $e(S) = I(S)/Q(S)$, so true by definition

Gives $n + 2w$ equations known to be true!

Unknowns are coefficients for $I(x)$ and $Q(x)$

Berlekamp-Welch: A Closer Look

What does $I(\neq)$ look like?

Berlekamp-Welch: A Closer Look

What does $I(\neq)$ look like?

$$\deg(e) = n - 1, \deg(Q) = W, \text{so } \deg(I) = n + W - 1$$

Berlekamp-Welch: A Closer Look

What does $I(\mathbb{F})$ look like?

$$\deg(e) = n - 1, \deg(\mathcal{O} = W) \text{ so } \deg(I) = n - W - 1$$

$$I(\mathbb{F}) = -\sum_{c \in \mathbb{F}} \mathbb{F}^c + \dots + -c \mathbb{F} + -\mathbb{F}$$

Berlekamp-Welch: A Closer Look

What does $I(\mathbb{F})$ look like?

$$\deg(e) = n - 1, \deg(Q) = W \text{ so } \deg(I) = n - W + 1$$

$$I(\mathbb{F}) = \sum_{c \in \mathbb{F}} \alpha_c x^{n-1-Wc} + \dots + \alpha_{-c} x^{-c} + \alpha_{-c} x^{-c}$$

What does $Q(\mathbb{F})$ look like?

Berlekamp-Welch: A Closer Look

What does $I(\mathbb{F})$ look like?

$\deg(e) = n - 1$, $\deg(Q) = W$ so $\deg(I) = n - W - 1$

$$I(\mathbb{F}) = \sum_{c \in \mathbb{F}} \alpha_c x^{n-1-W} + \dots + \alpha_c x + \alpha_{\infty}$$

What does $Q(\mathbb{F})$ look like?

$Q(\mathbb{F}) = (x - c_1)(x - c_2) \dots (x - c_W)$, so degree W

Berlekamp-Welch: A Closer Look

What does $I(\mathbb{F})$ look like?

$\deg(e) = n - 1$, $\deg(\mathcal{O} = W)$ so $\deg(I) = n - W - 1$

$$I(\mathbb{F}) = -\sum_{c \in W} \mathbb{F}^{n-1-Wc} + \dots + -c\mathbb{F} + -\mathbb{E}$$

What does $\mathcal{Q}(\mathbb{F})$ look like?

$\mathcal{Q}(\mathbb{F}) = (\mathbb{F} - c) (\mathbb{F} - c_1) \dots (\mathbb{F} - c_W)$, so degree W

$$\mathcal{Q}(\mathbb{F}) = \mathbb{F}^W + \dots + \mathbb{F} + \mathbb{E}$$

Berlekamp-Welch: A Closer Look

What does $I(\neq)$ look like?

$\deg(e) = \wedge 1$, $\deg(\mathcal{O} = W)$ so $\deg(I) = \wedge + W - 1$

$$I(\neq) = -\wedge + W_c \neq^{\wedge + W_c} + \dots + -_c \neq + -_{\mathcal{E}}$$

What does $\mathcal{Q}(\neq)$ look like?

$\mathcal{Q}(\neq) = (\neq \mathcal{C}_c)(\neq \mathcal{C}_1) \dots (\neq \mathcal{C}_W)$, so degree W

$$\mathcal{Q}(\neq) = 4_W \neq^W + \dots + 4_c \neq + 4_{\mathcal{E}}$$

But wait! $4_W = 1$ for any $\mathcal{C}_c; \dots; \mathcal{C}_W$

$$\text{So } \mathcal{Q}(\neq) = \neq^W + 4_{W_c} \neq^{W_c} + \dots + 4_c \neq + 4_{\mathcal{E}}$$

Berlekamp-Welch: A Closer Look

What does $I(\mathcal{F})$ look like?

$\deg(e) = \sum_{c \in C} 1$, $\deg(\mathcal{G} = W)$ so $\deg(I) = \sum_{c \in C} 1 + W - 1$

$$I(\mathcal{F}) = \sum_{c \in C} \mathcal{F}(c)^{\sum_{c' \in C} 1 + W} + \dots + \sum_{c \in C} \mathcal{F}(c) + \mathcal{E}$$

What does $Q(\mathcal{F})$ look like?

$Q(\mathcal{F}) = (\mathcal{F}(c_1) - \mathcal{F}(c_2)) \dots (\mathcal{F}(c_{W-1}) - \mathcal{F}(c_W))$, so degree W

$$Q(\mathcal{F}) = \sum_{c \in C} \mathcal{F}(c)^W + \dots + \sum_{c \in C} \mathcal{F}(c) + \mathcal{E}$$

But wait! $\sum_{c \in C} \mathcal{F}(c)^W = 1$ for any c_1, \dots, c_W

$$\text{So } Q(\mathcal{F}) = \sum_{c \in C} \mathcal{F}(c)^W + \sum_{c \in C} \mathcal{F}(c)^{W-1} + \dots + \sum_{c \in C} \mathcal{F}(c) + \mathcal{E}$$

Have $\sum_{c \in C} 1 + W$ unknowns from I , W from C

Matches $\sum_{c \in C} 1 + 2W$ linear eqns of the form $I(\mathcal{S}) = \mathcal{Q}(\mathcal{S})$

Berlekamp-Welch: A Closer Look

What does $I(\mathbb{F})$ look like?

$\deg(e) = n - 1$, $\deg(C) = W$ so $\deg(I) = n - W - 1$

$$I(\mathbb{F}) = -\sum_{c \in C} \mathbb{F}^{n-1-Wc} + \dots + -c \mathbb{F}^{n-1-W} + -e$$

What does $Q(\mathbb{F})$ look like?

$Q(\mathbb{F}) = (\mathbb{F} - C_1)(\mathbb{F} - C_2) \dots (\mathbb{F} - C_W)$, so degree W

$$Q(\mathbb{F}) = \mathbb{F}^W + \dots + \mathbb{F} + 1$$

But wait! $\mathbb{F} = 1$ for any C_1, \dots, C_W

$$\text{So } Q(\mathbb{F}) = \mathbb{F}^W + \sum_{c \in C} \mathbb{F}^{Wc} + \dots + \mathbb{F} + 1$$

Have $n - W$ unknowns from I , W from C

Matches $n - 2W$ linear eqns of the form $I(\mathbb{F}) = e Q(\mathbb{F})$

Linear Algebra: can find I, C so have $e(\mathbb{F}) = \frac{I(\mathbb{F})}{Q(\mathbb{F})}$

Berlekamp-Welch: Example

Want to send length 2 message, have 1 corruption

Receive messages $(1;3)$, $(2;1)$, $(3;4)$, $(4;0)$ mod 7

Berlekamp-Welch: Example

Want to send length 2 message, have 1 corruption

Receive messages (1;3), (2;1), (3;4), (4;0) mod 7

$$I(\hat{f}) = -_1 \hat{f}^1 + -_c \hat{f} + -_{\mathbb{E}} \mathcal{Q}(\hat{f}) = \hat{f} + 4_{\mathbb{E}}$$

Berlekamp-Welch: Example

Want to send length 2 message, have 1 corruption

Receive messages (1;3), (2;1), (3;4), (4;0) mod 7

$$I(f) = -_1 f^1 + -_c f + -_{CE} \quad C(f) = f + 4_{CE}$$

$$\text{Eq 1: } I(1) = qC(1), \text{ so } -_1 + -_c + -_{CE} = 3(1 + 4_{CE})$$

Berlekamp-Welch: Example

Want to send length 2 message, have 1 corruption

Receive messages (1;3), (2;1), (3;4), (4;0) mod 7

$$I(f) = -_1 f^1 + -_c f + -_E \mathcal{Q}(f) = f + 4E$$

$$\text{Eq 1: } I(1) = q\mathcal{Q}(1), \text{ so } -_1 + -_c + -_E = 3(1 + 4E)$$

$$\text{Eq 2: } I(2) = q\mathcal{Q}(2), \text{ so } 4-_1 + 2-_c + -_E = 1(2 + 4E)$$

Berlekamp-Welch: Example

Want to send length 2 message, have 1 corruption

Receive messages (1;3), (2;1), (3;4), (4;0) mod 7

$$I(\hat{t}) = -_1 \hat{t}^1 + -_c \hat{t} + -_{CE} \mathcal{C}(\hat{t}) = \hat{t} + 4_{CE}$$

$$\text{Eq 1: } I(1) = q\mathcal{C}(1), \text{ so } -_1 + -_c + -_{CE} = 3(1 + 4_{CE})$$

$$\text{Eq 2: } I(2) = q\mathcal{C}(2), \text{ so } 4-_1 + 2-_c + -_{CE} = 1(2 + 4_{CE})$$

$$\text{Eq 3: } I(3) = q\mathcal{C}(3), \text{ so } 9-_1 + 3-_c + -_{CE} = 4(3 + 4_{CE})$$

Berlekamp-Welch: Example

Want to send length 2 message, have 1 corruption

Receive messages (1;3), (2;1), (3;4), (4;0) mod 7

$$I(\mathcal{I}) = -_1 \mathcal{I}^1 + -_c \mathcal{I}^c + -_{\mathcal{E}} \mathcal{C}(\mathcal{I}) = \mathcal{I} + 4_{\mathcal{E}}$$

$$\text{Eq 1: } I(1) = q\mathcal{C}(1), \text{ so } -_1 + -_c + -_{\mathcal{E}} = 3(1 + 4_{\mathcal{E}})$$

$$\text{Eq 2: } I(2) = q\mathcal{C}(2), \text{ so } 4-_1 + 2-_c + -_{\mathcal{E}} = 1(2 + 4_{\mathcal{E}})$$

$$\text{Eq 3: } I(3) = q\mathcal{C}(3), \text{ so } 9-_1 + 3-_c + -_{\mathcal{E}} = 4(3 + 4_{\mathcal{E}})$$

$$\text{Eq 4: } I(4) = q\mathcal{C}(4), \text{ so } 16-_1 + 4-_c + -_{\mathcal{E}} = 0(4 + 4_{\mathcal{E}})$$

Berlekamp-Welch: Example

Want to send length 2 message, have 1 corruption

Receive messages (1;3), (2;1), (3;4), (4;0) mod 7

$$I(\#) = -_1 \#^1 + -_c \# + -_E \mathcal{C}(\#) = \# + 4E$$

$$\text{Eq 1: } I(1) = q\mathcal{C}(1), \text{ so } -_1 + -_c + -_E = 3(1 + 4E)$$

$$\text{Eq 2: } I(2) = q\mathcal{C}(2), \text{ so } 4-_1 + 2-_c + -_E = 1(2 + 4E)$$

$$\text{Eq 3: } I(3) = q\mathcal{C}(3), \text{ so } 9-_1 + 3-_c + -_E = 4(3 + 4E)$$

$$\text{Eq 4: } I(4) = q\mathcal{C}(4), \text{ so } 16-_1 + 4-_c + -_E = 0(4 + 4E)$$

Note: all eqns modulo 7, so can shrink some nums

(Berlekamp-Welch: Example): Continued

Simplify equations mod 7, move all variables to left:

$$-1 + -c + -cE \quad 34cE = 3$$

$$4-1 + 2-c + -cE \quad 4cE = 2$$

$$2-1 + 3-c + -cE \quad 44cE = 5$$

$$2-1 + 4-c + -cE = 0$$

(Berlekamp-Welch: Example): Continued

Simplify equations mod 7, move all variables to left:

$$-1 + -c + -e \quad 34e = 3$$

$$4-1 + 2-c + -e \quad 4e = 2$$

$$2-1 + 3-c + -e \quad 44e = 5$$

$$2-1 + 4-c + -e = 0$$

Can use Gaussian Elimination (mod 7) to solve

(Berlekamp-Welch: Example): Continued

Simplify equations mod 7, move all variables to left:

$$-| + -_c + -_{\mathbb{E}} \quad 34_{\mathbb{E}} = 3$$

$$4-| + 2-_c + -_{\mathbb{E}} \quad 4_{\mathbb{E}} = 2$$

$$2-| + 3-_c + -_{\mathbb{E}} \quad 44_{\mathbb{E}} = 5$$

$$2-| + 4-_c + -_{\mathbb{E}} = 0$$

Can use Gaussian Elimination (mod 7) to solve

$$\text{Here, } -| = 3, \quad -_c = 6, \quad -_{\mathbb{E}} = 5, \quad 4_{\mathbb{E}} = 6$$

$$\text{So } I(\hat{t}) = 3\hat{t}^3 + 6\hat{t} + 5, \quad Q(\hat{t}) = \hat{t} + 6$$

(Berlekamp-Welch: Example): Continued

Simplify equations mod 7, move all variables to left:

$$-| + -_c + -_{CE} \quad 34_{CE} = 3$$

$$4-| + 2-_c + -_{CE} \quad 4_{CE} = 2$$

$$2-| + 3-_c + -_{CE} \quad 44_{CE} = 5$$

$$2-| + 4-_c + -_{CE} = 0$$

Can use Gaussian Elimination (mod 7) to solve

$$\text{Here, } -| = 3, -_c = 6, -_{CE} = 5, 4_{CE} = 6$$

$$\text{So } I(\hat{t}) = 3\hat{t}^2 + 6\hat{t} + 5, C(\hat{t}) = \hat{t} + 6$$

Do poly long division mod 7 to get $e(\hat{t}) = 3\hat{t} + 2$

Original messages: $e(1) = 5, e(2) = 1$

Fin

Next time: countability!