Error Correcting Codes

CS70 Summer 2016 - Lecture 8A

David Dinh

08 August 2016

UC Berkeley

Today

Final logistics

Erasure codes

Berlekamp-Walsh codes

Final logistics

Final will be held on **Friday, 12 August** from **11:30-2:30** in 120 Latimer (last names A-H) and 1 Pimentel (last names I-Z).

Students with conflicts and DSP students: if you haven't heard from us by now, contact us ASAP. Students clearing old incompletes: just show up as normal.

170 minutes. 11 questions. 3 pages (1 double sided + 1 single sided sheet, or 3 single sided sheets) of notes allowed.

Final Composition

Mix of T/F, short answer, free-form questions. Same style as the midterms: not too much calculation, tests intuitive understanding of material. Difficulty range should be around the same.

Coverage: everything we've learned in this class. Emphasis on material from last week and this week. Around half of the questions require this material (but many of these also involve material from before MT2).

Best way to study: practice questions that really *test your understanding of the material*. We're testing for how well you can apply concepts to things you haven't exactly seen before, not how well you can perform some procedure you memorized.

Erasure Codes (1/2)

Polynomial interpolation can also be used to recover data.

Erasure Codes (1/2)

Polynomial interpolation can also be used to recover data.

Same principle as secret sharing!

Packets dropped $\rightarrow$ dead officials.

Packets you receive $\rightarrow$ live officials.

Erasure Codes (1/2)

Polynomial interpolation can also be used to recover data.

Same principle as secret sharing!

Packets dropped $\rightarrow$ dead officials.

Packets you receive $\rightarrow$ live officials.

Don't need all the bits to recover the message $\rightarrow$ not the officials need to be present to recover the codes.

You want to recover the original message if you receive enough information!

Erasure Codes (2/2)

Alex wants to send David n packets over a lossy channel (each one some number over $GF(q)$, q prime); call the packets $m_1, m_2, \ldots, m_n$. Say the channel drops k packets (although we don't know which ones).

Erasure Codes (2/2)

Alex wants to send David n packets over a lossy channel (each one some number over $GF(q)$, q prime); call the packets $m_1, m_2, \ldots, m_n$. Say the channel drops k packets (although we don't know which ones).

Has to be a unique degree-$n-1$ polynomial passing through n points in $GF(q)$.

Erasure Codes (2/2)

Alex wants to send David n packets over a lossy channel (each one some number over $GF(q)$, q prime); call the packets $m_1, m_2, \ldots, m_n$. Say the channel drops k packets (although we don't know which ones).

Has to be a unique degree-$n-1$ polynomial passing through n points in $GF(q)$.

Alex defines a degree-$n-1$ polynomial $P(x)$ passing through $(1, m_1), (2, m_2), \ldots, (n, m_n)$ in $GF(q)$. Want to send enough information to reconstruct this polynomial on the other side of the channel.

Erasure Codes (2/2)

Alex wants to send David n packets over a lossy channel (each one some number over $GF(q)$, q prime); call the packets $m_1, m_2, ..., m_n$. Say the channel drops k packets (although we don't know which ones).

Has to be a unique degree-$n-1$ polynomial passing through n points in $GF(q)$.

Alex defines a degree-$n-1$ polynomial $P(x)$ passing through $(1, m_1), (2, m_2), ..., (n, m_n)$ in $GF(q)$. Want to send enough information to reconstruct this polynomial on the other side of the channel.

Trick: send k extra points too! $(n+1, P(n+1)), ..., (n+d, P(n+k))$.

Erasure Codes (2/2)

Alex wants to send David n packets over a lossy channel (each one some number over $GF(q)$, q prime); call the packets $m_1, m_2, \ldots, m_n$. Say the channel drops k packets (although we don't know which ones).

Has to be a unique degree-$n-1$ polynomial passing through n points in $GF(q)$.

Alex defines a degree-$n-1$ polynomial $P(x)$ passing through $(1, m_1), (2, m_2), \ldots, (n, m_n)$ in $GF(q)$. Want to send enough information to reconstruct this polynomial on the other side of the channel.

Trick: send k extra points too! $(n+1, P(n+1)), \ldots, (n+d, P(n+k))$.

No matter which packets are dropped, David can recover P and find the original packets!

Erasure Codes (2/2)

Alex wants to send David n packets over a lossy channel (each one some number over $GF(q)$, q prime); call the packets $m_1, m_2, \dots, m_n$. Say the channel drops k packets (although we don't know which ones).

Has to be a unique degree-$n-1$ polynomial passing through n points in $GF(q)$.

Alex defines a degree-$n-1$ polynomial $P(x)$ passing through $(1, m_1), (2, m_2), \dots, (n, m_n)$ in $GF(q)$. Want to send enough information to reconstruct this polynomial on the other side of the channel.

Trick: send k extra points too! $(n+1, P(n+1)), \dots, (n+d, P(n+k))$.

No matter which packets are dropped, David can recover P and find the original packets!

Note: does require that $q \geq n+k, \max_i m_i$, but finding big primes is easy so it's not normally a problem.

Live Demo

Corruption Errors

What if things aren't just erased, but also corrupted?

Problem: now you don't know which packets are correct and which ones are incorrect anymore.

Corruption Errors

What if things aren't just erased, but also corrupted?

Problem: now you don't know which packets are correct and which ones are incorrect anymore.

Model: channel corrupts k packets (i.e. changes them to an arbitrary number). No information on which packets it corrupts.

Corruption Errors

What if things aren't just erased, but also corrupted?

Problem: now you don't know which packets are correct and which ones are incorrect anymore.

Model: channel corrupts k packets (i.e. changes them to an arbitrary number). No information on which packets it corrupts.

Now we need to figure out which packets were corrupted in addition to the original message.

Corruption Errors

What if things aren't just erased, but also corrupted?

Problem: now you don't know which packets are correct and which ones are incorrect anymore.

Model: channel corrupts k packets (i.e. changes them to an arbitrary number). No information on which packets it corrupts.

Now we need to figure out which packets were corrupted in addition to the original message.

Need to send more packets!

Berlekamp-Walsh

Suppose again that Alex is trying to send some degree-$n-1$ polynomial $P(x)$ to David over a corrupting channel by sending points $P(1), P(2), \ldots$. David receives points $r_1, r_2, \ldots$.

Berlekamp-Walsh

Suppose again that Alex is trying to send some degree-$n-1$ polynomial $P(x)$ to David over a corrupting channel by sending points $P(1), P(2), \ldots$. David receives points $r_1, r_2, \ldots$.

Suppose the channel corrupts packets $e_1, e_2, ..., e_k$. David doesn't know e_k, wants to recover $P(i)$.

Berlekamp-Walsh

Suppose again that Alex is trying to send some degree-$n-1$ polynomial $P(x)$ to David over a corrupting channel by sending points $P(1), P(2), \ldots$. David receives points $r_1, r_2, \ldots$.

Suppose the channel corrupts packets $e_1, e_2, ..., e_k$. David doesn't know e_k, wants to recover $P(i)$.

Main trick: let $E(x) = (x-e_1)(x-e_2)\ldots(x-e_k)$ (again, we don't know what E is yet). Notice that $P(i)E(i) = r_iE(i)$ at all points i: if i is corrupted, E is 0; otherwise, $P(i) = r_i$.

Berlekamp-Walsh

Suppose again that Alex is trying to send some degree-$n-1$ polynomial $P(x)$ to David over a corrupting channel by sending points $P(1), P(2), \dots$. David receives points $r_1, r_2, \dots$.

Suppose the channel corrupts packets $e_1, e_2, ..., e_k$. David doesn't know e_k, wants to recover $P(i)$.

Main trick: let $E(x) = (x-e_1)(x-e_2)\dots(x-e_k)$ (again, we don't know what E is yet). Notice that $P(i)E(i) = r_iE(i)$ at all points i: if i is corrupted, E is 0; otherwise, $P(i) = r_i$.

Idea: solve for P and E. Let $Q = PE$.

Degree of Q?

Berlekamp-Walsh

Suppose again that Alex is trying to send some degree-$n-1$ polynomial $P(x)$ to David over a corrupting channel by sending points $P(1), P(2), \ldots$. David receives points $r_1, r_2, \ldots$.

Suppose the channel corrupts packets $e_1, e_2, ..., e_k$. David doesn't know e_k, wants to recover $P(i)$.

Main trick: let $E(x) = (x-e_1)(x-e_2)\ldots(x-e_k)$ (again, we don't know what E is yet). Notice that $P(i)E(i) = r_iE(i)$ at all points i: if i is corrupted, E is 0; otherwise, $P(i) = r_i$.

Idea: solve for P and E. Let $Q = PE$.

Degree of Q? $\deg(P) = n-1$. $\deg(E) = k$.

Berlekamp-Walsh

Suppose again that Alex is trying to send some degree-$n-1$ polynomial $P(x)$ to David over a corrupting channel by sending points $P(1), P(2), \ldots$. David receives points $r_1, r_2, \ldots$.

Suppose the channel corrupts packets $e_1, e_2, ..., e_k$. David doesn't know e_k, wants to recover $P(i)$.

Main trick: let $E(x) = (x-e_1)(x-e_2)\ldots(x-e_k)$ (again, we don't know what E is yet). Notice that $P(i)E(i) = r_i E(i)$ at all points i: if i is corrupted, E is 0; otherwise, $P(i) = r_i$.

Idea: solve for P and E. Let $Q = PE$.

Degree of Q? $\deg(P) = n-1$. $\deg(E) = k$. $\deg(Q) = n-1+k$.

Berlekamp-Walsh

Suppose again that Alex is trying to send some degree-$n-1$ polynomial $P(x)$ to David over a corrupting channel by sending points $P(1), P(2), \ldots$. David receives points $r_1, r_2, \ldots$.

Suppose the channel corrupts packets $e_1, e_2, ..., e_k$. David doesn't know e_k, wants to recover $P(i)$.

Main trick: let $E(x) = (x-e_1)(x-e_2)\ldots(x-e_k)$ (again, we don't know what E is yet). Notice that $P(i)E(i) = r_iE(i)$ at all points i: if i is corrupted, E is 0; otherwise, $P(i) = r_i$.

Idea: solve for P and E. Let $Q = PE$.

Degree of Q? $\deg(P) = n-1$. $\deg(E) = k$. $\deg(Q) = n-1+k$. $n+k$ unknown coefficients for this polynomial.

Berlekamp-Walsh

Suppose again that Alex is trying to send some degree-$n-1$ polynomial $P(x)$ to David over a corrupting channel by sending points $P(1), P(2), \ldots$. David receives points $r_1, r_2, \ldots$.

Suppose the channel corrupts packets $e_1, e_2, ..., e_k$. David doesn't know e_k, wants to recover $P(i)$.

Main trick: let $E(x) = (x-e_1)(x-e_2)\ldots(x-e_k)$ (again, we don't know what E is yet). Notice that $P(i)E(i) = r_iE(i)$ at all points i: if i is corrupted, E is 0; otherwise, $P(i) = r_i$.

Idea: solve for P and E. Let $Q = PE$.

Degree of Q? $\deg(P) = n-1$. $\deg(E) = k$. $\deg(Q) = n-1+k$. $n+k$ unknown coefficients for this polynomial.

$n+k$ unknown coeffs for Q on the left, k for E on the right (since the coefficient of E for x^k is 1). How many unknowns total?

Berlekamp-Walsh

Suppose again that Alex is trying to send some degree-$n-1$ polynomial $P(x)$ to David over a corrupting channel by sending points $P(1), P(2), \ldots$. David receives points $r_1, r_2, \ldots$.

Suppose the channel corrupts packets $e_1, e_2, ..., e_k$. David doesn't know e_k, wants to recover $P(i)$.

Main trick: let $E(x) = (x - e_1)(x - e_2) \ldots (x - e_k)$ (again, we don't know what E is yet). Notice that $P(i)E(i) = r_i E(i)$ at all points i: if i is corrupted, E is 0; otherwise, $P(i) = r_i$.

Idea: solve for P and E. Let $Q = PE$.

Degree of Q? $\deg(P) = n - 1$. $\deg(E) = k$. $\deg(Q) = n - 1 + k$. $n + k$ unknown coefficients for this polynomial.

$n + k$ unknown coeffs for Q on the left, k for E on the right (since the coefficient of E for x^k is 1). How many unknowns total? $n + 2k$.

Berlekamp-Walsh

Suppose again that Alex is trying to send some degree-$n-1$ polynomial $P(x)$ to David over a corrupting channel by sending points $P(1), P(2), \dots$. David receives points $r_1, r_2, \dots$.

Suppose the channel corrupts packets $e_1, e_2, ..., e_k$. David doesn't know e_k, wants to recover $P(i)$.

Main trick: let $E(x) = (x-e_1)(x-e_2)\dots(x-e_k)$ (again, we don't know what E is yet). Notice that $P(i)E(i) = r_iE(i)$ at all points i: if i is corrupted, E is 0; otherwise, $P(i) = r_i$.

Idea: solve for P and E. Let $Q = PE$.

Degree of Q? $\deg(P) = n-1$. $\deg(E) = k$. $\deg(Q) = n-1+k$. $n+k$ unknown coefficients for this polynomial.

$n+k$ unknown coeffs for Q on the left, k for E on the right (since the coefficient of E for x^k is 1). How many unknowns total? $n+2k$.

Send $n+2k$ points to solve this equation and recover the polynomial!

Berlekamp-Walsh, step-by-step

Alex wants to send a message of n numbers to David over a channel that corrupts k packets. Operate in $GF(q)$ for some $p > n+2k$ and p bigger than the max packet size.

Berlekamp-Walsh, step-by-step

Alex wants to send a message of n numbers to David over a channel that corrupts k packets. Operate in $GF(q)$ for some $p > n+2k$ and p bigger than the max packet size.

1. Alex interpolates a degree $n-1$ polynomial $P(x)$ over the messages, like for erasure codes.

Berlekamp-Walsh, step-by-step

Alex wants to send a message of n numbers to David over a channel that corrupts k packets. Operate in $GF(q)$ for some $p > n+2k$ and p bigger than the max packet size.

1. Alex interpolates a degree $n-1$ polynomial $P(x)$ over the messages, like for erasure codes.
2. Alex sends $n+2k$ points to David:
   $(1, P(1)), (2, P(2)), \ldots, (n+2k, P(n+2k))$.

Berlekamp-Walsh, step-by-step

Alex wants to send a message of n numbers to David over a channel that corrupts k packets. Operate in $GF(q)$ for some $p > n+2k$ and p bigger than the max packet size.

1. Alex interpolates a degree $n-1$ polynomial $P(x)$ over the messages, like for erasure codes.
2. Alex sends $n+2k$ points to David: $(1, P(1)), (2, P(2)), \ldots, (n+2k, P(n+2k))$.
3. David receieves $n+2k$ points $(1, r_1), (2, r_2), \ldots, (n+2k, r_{n+2k})$.

Berlekamp-Walsh, step-by-step

Alex wants to send a message of n numbers to David over a channel that corrupts k packets. Operate in $GF(q)$ for some $p > n+2k$ and p bigger than the max packet size.

1. Alex interpolates a degree $n-1$ polynomial $P(x)$ over the messages, like for erasure codes.
2. Alex sends $n+2k$ points to David: $(1, P(1)), (2, P(2)), \ldots, (n+2k, P(n+2k))$.
3. David receieves $n+2k$ points $(1, r_1), (2, r_2), \ldots, (n+2k, r_{n+2k})$.
4. David writes down a system of equations:

   $$q_{n+k-1}x_i^{n+k-1} + \cdots + q_2 x_i^2 + q_1 x_i + q_0 = r_i(x_i^k + b_{k-1}x_i^{k-1} + \cdots + b_1 x_i + b_0)$$

   for each x_i.

Berlekamp-Walsh, step-by-step

Alex wants to send a message of n numbers to David over a channel that corrupts k packets. Operate in $GF(q)$ for some $p > n+2k$ and p bigger than the max packet size.

1. Alex interpolates a degree $n-1$ polynomial $P(x)$ over the messages, like for erasure codes.
2. Alex sends $n+2k$ points to David: $(1,P(1)),(2,P(2)),\dots,(n+2k,P(n+2k))$.
3. David receieves $n+2k$ points $(1,r_1),(2,r_2),\dots,(n+2k,r_{n+2k})$.
4. David writes down a system of equations:

   $$q_{n+k-1}x_i^{n+k-1}+\cdots+q_2x_i^2+q_1x_i+q_0 = r_i(x_i^k+b_{k-1}x_i^{k-1}+\cdots+b_1x_i+b_0)$$

   for each x_i.
5. David solves the equations for the coefficients for Q and E.

Berlekamp-Walsh, step-by-step

Alex wants to send a message of n numbers to David over a channel that corrupts k packets. Operate in $GF(q)$ for some $p > n+2k$ and p bigger than the max packet size.

1. Alex interpolates a degree $n-1$ polynomial $P(x)$ over the messages, like for erasure codes.
2. Alex sends $n+2k$ points to David: $(1, P(1)), (2, P(2)), \ldots, (n+2k, P(n+2k))$.
3. David receieves $n+2k$ points $(1, r_1), (2, r_2), \ldots, (n+2k, r_{n+2k})$.
4. David writes down a system of equations:
$$q_{n+k-1}x_i^{n+k-1} + \cdots + q_2 x_i^2 + q_1 x_i + q_0 = r_i(x_i^k + b_{k-1}x_i^{k-1} + \cdots + b_1 x_i + b_0)$$
for each x_i.
5. David solves the equations for the coefficients for Q and E.
6. David recovers $P(x) = Q(x)/E(x)$ by polynomial division.

Berlekamp-Walsh, step-by-step

Alex wants to send a message of n numbers to David over a channel that corrupts k packets. Operate in $GF(q)$ for some $p > n+2k$ and p bigger than the max packet size.

1. Alex interpolates a degree $n-1$ polynomial $P(x)$ over the messages, like for erasure codes.
2. Alex sends $n+2k$ points to David: $(1, P(1)), (2, P(2)), \ldots, (n+2k, P(n+2k))$.
3. David receieves $n+2k$ points $(1, r_1), (2, r_2), \ldots, (n+2k, r_{n+2k})$.
4. David writes down a system of equations:

   $$q_{n+k-1}x_i^{n+k-1} + \cdots + q_2 x_i^2 + q_1 x_i + q_0 = r_i(x_i^k + b_{k-1}x_i^{k-1} + \cdots + b_1 x_i + b_0)$$

   for each x_i.
5. David solves the equations for the coefficients for Q and E.
6. David recovers $P(x) = Q(x)/E(x)$ by polynomial division.

Live Demo

Correctness of BW

Does BW necessarily give a solution?

Correctness of BW

Does BW necessarily give a solution? Yes, since we know that $P(x_i)E(x_i) = r_iE(x_i)$ at all the points we sent.

Correctness of BW

Does BW necessarily give a solution? Yes, since we know that $P(x_i)E(x_i) = r_iE(x_i)$ at all the points we sent.

Is this solution **unique**? i.e. do we know that there aren't solutions floating around somewhere that don't correspond to the answer?

Correctness of BW

Does BW necessarily give a solution? Yes, since we know that $P(x_i)E(x_i) = r_i E(x_i)$ at all the points we sent.

Is this solution **unique**? i.e. do we know that there aren't solutions floating around somewhere that don't correspond to the answer?

Formally: suppose that we have some solution coefficients that specify some polyomial $Q'(x)$ and $E'(x)$. How do we know thaht $Q'(x)/E'(x) = P(x)$?

Uniqueness of BW

Claim: Let $Q(x) = P(x)E(x)$. Then for any Q', E' as defined above, $Q(x)E'(x) = Q'(x)E(x)$ for $1 \leq x \leq n+2k$ (i.e. they are the same polynomial, since their degree is $n+2k-1$). Therefore we would have $Q'(x)/E'(x) = Q(x)/E(x) = P(x)$.

Uniqueness of BW

Claim: Let $Q(x) = P(x)E(x)$. Then for any Q', E' as defined above, $Q(x)E'(x) = Q'(x)E(x)$ for $1 \leq x \leq n+2k$ (i.e. they are the same polynomial, since their degree is $n+2k-1$). Therefore we would have $Q'(x)/E'(x) = Q(x)/E(x) = P(x)$.

Proof of claim: We know $Q'(i) = r_i E'(i)$ by construction.

Uniqueness of BW

Claim: Let $Q(x) = P(x)E(x)$. Then for any Q', E' as defined above, $Q(x)E'(x) = Q'(x)E(x)$ for $1 \leq x \leq n+2k$ (i.e. they are the same polynomial, since their degree is $n+2k-1$). Therefore we would have $Q'(x)/E'(x) = Q(x)/E(x) = P(x)$.

Proof of claim: We know $Q'(i) = r_i E'(i)$ by construction.

Case 1: Suppose $E(i) = 0$.

Uniqueness of BW

Claim: Let $Q(x) = P(x)E(x)$. Then for any Q', E' as defined above, $Q(x)E'(x) = Q'(x)E(x)$ for $1 \leq x \leq n+2k$ (i.e. they are the same polynomial, since their degree is $n+2k-1$). Therefore we would have $Q'(x)/E'(x) = Q(x)/E(x) = P(x)$.

Proof of claim: We know $Q'(i) = r_i E'(i)$ by construction.

Case 1: Suppose $E(i) = 0$. Then $Q(i)$ is 0. So $Q(x)E'(x) = Q'(x)E(x)$, as desired.

Uniqueness of BW

Claim: Let $Q(x) = P(x)E(x)$. Then for any Q', E' as defined above, $Q(x)E'(x) = Q'(x)E(x)$ for $1 \leq x \leq n+2k$ (i.e. they are the same polynomial, since their degree is $n+2k-1$). Therefore we would have $Q'(x)/E'(x) = Q(x)/E(x) = P(x)$.

Proof of claim: We know $Q'(i) = r_i E'(i)$ by construction.

Case 1: Suppose $E(i) = 0$. Then $Q(i)$ is 0. So $Q(x)E'(x) = Q'(x)E(x)$, as desired.

Case 2: Suppose $E'(i) = 0$. Then $Q'(i) = 0$. So $Q(x)E'(x) = Q'(x)E(x)$, as desired.

Uniqueness of BW

Claim: Let $Q(x) = P(x)E(x)$. Then for any Q', E' as defined above, $Q(x)E'(x) = Q'(x)E(x)$ for $1 \leq x \leq n+2k$ (i.e. they are the same polynomial, since their degree is $n+2k-1$). Therefore we would have $Q'(x)/E'(x) = Q(x)/E(x) = P(x)$.

Proof of claim: We know $Q'(i) = r_i E'(i)$ by construction.

Case 1: Suppose $E(i) = 0$. Then $Q(i)$ is 0. So $Q(x)E'(x) = Q'(x)E(x)$, as desired.

Case 2: Suppose $E'(i) = 0$. Then $Q'(i) = 0$. So $Q(x)E'(x) = Q'(x)E(x)$, as desired.

Case 3: Suppose both $E(i)$ and $E'(i)$ are nonzero.

Uniqueness of BW

Claim: Let $Q(x) = P(x)E(x)$. Then for any Q', E' as defined above, $Q(x)E'(x) = Q'(x)E(x)$ for $1 \leq x \leq n+2k$ (i.e. they are the same polynomial, since their degree is $n+2k-1$). Therefore we would have $Q'(x)/E'(x) = Q(x)/E(x) = P(x)$.

Proof of claim: We know $Q'(i) = r_i E'(i)$ by construction.

Case 1: Suppose $E(i) = 0$. Then $Q(i)$ is 0. So $Q(x)E'(x) = Q'(x)E(x)$, as desired.

Case 2: Suppose $E'(i) = 0$. Then $Q'(i) = 0$. So $Q(x)E'(x) = Q'(x)E(x)$, as desired.

Case 3: Suppose both $E(i)$ and $E'(i)$ are nonzero. Since $Q'(i) = r_i E'(i)$, $r_i = Q'(i)/E'(i)$. Similarly, $r_i = Q(i)/E(i)$. Therefore, $Q'(i)/E'(i) = Q(i)/E(i)$, as desired. $\square$