CS 70 Discrete Mathematics and Probability Theory
Spring 2019 Course Notes

DIS 5B

1 Berlekamp-Welch Warm Up

(a) When does $r_i = P(i)$? When does r_i not equal $P(i)$?

(b) If you want to send a length-n message, what should the degree of $P(x)$ be? Why?

(c) If there are at most k erasure errors, how many packets should you send? If there are at most k general errors, how many packets should you send? (We will see the reason for this later.) Now we will only consider general errors.

(d) What do the roots of the error polynomial $E(x)$ tell you? Does the receiver know the roots of $E(x)$? If there are at most k errors, what is the maximum degree of $E(x)$? Using the information about the degree of $P(x)$ and $E(x)$, what is the degree of $Q(x) = P(x)E(x)$?

(e) Why is the equation $Q(i) = P(i)E(i) = r_iE(i)$ always true? (Consider what happens when $P(i) = r_i$, and what happens when $P(i)$ does not equal r_i.)

(f) In the polynomials $Q(x)$ and $E(x)$, how many total unknown coefficients are there? (These are the variables you must solve for. Think about the degree of the polynomials.) When you receive packets, how many equations do you have? Do you have enough equations to solve for all of the unknowns? (Think about the answer to the earlier question - does it make sense now why we send as many packets as we do?)

(g) If you have $Q(x)$ and $E(x)$, how does one recover $P(x)$? If you know $P(x)$, how can you recover the original message?

Solution:

(a) The received packet is correct; the received packet is corrupted.

(b) P has degree at most $n-1$ since n points determine a degree $\leq n-1$ polynomial.

(c) $n+k$; $n+2k$.

(d) The locations of corrupted packets. No ($E(x)$ is a polynomial that the receiver needs to compute in order to obtain $P(x)$). k. The degree of Q is $(n-1)+(k) = n+k-1$.

(e) If $P(i) = r_i$, then $P(i)E(i) = r_iE(i)$. If $P(i) \neq r_i$, then $E(i) = 0$.

(f) $(n+k-1+1)+(k) = n+2k$ unknowns. There are $n+2k$ equations. Yes (if the actual number of errors is less than k, then there will be multiple solutions).

(g) $P(x) = Q(x)/E(x)$. Compute $P(i)$ for $1 \le i \le n$. Alternatively, since we know the error-locator polynomial $E(x)$, we can find its roots to figure out which packets were corrupted and then we only need to evaluate $P(x)$ at the locations of the errors.

2 Berlekamp-Welch Algorithm

In this question we will send the message $(m_0, m_1, m_2) = (4, 3, 2)$ of length $n = 3$. We will use an error-correcting code for $k = 1$ general error, doing arithmetic over GF(5).

(a) Construct a polynomial $P(x) \pmod 5$ of degree at most 2, so that

$$P(0) = 4, \qquad P(1) = 3, \qquad P(2) = 2.$$

What is the message $(c_0, c_1, c_2, c_3, c_4)$ that is sent?

(b) Suppose the message is corrupted by changing c_0 to 0. Set up the system of linear equations in the Berlekamp-Welch algorithm to find $Q(x)$ and $E(x)$.

(c) Assume that after solving the equations in part (b) we get $Q(x) = -x^2 + 4x$ and $E(x) = x$. Show how to recover the original message from Q and E.

Solution:

(a) We use Lagrange interpolation to construct the unique quadratic polynomial $P(x)$ such that $P(0) = m_0 = 4, P(1) = m_1 = 3, P(2) = m_2 = 2$.

$$\begin{aligned}
\Delta_0(x) &= \frac{(x-1)(x-2)}{(0-1)(0-2)} = \frac{x^2-3x+2}{2} \\
\Delta_1(x) &= \frac{(x-0)(x-2)}{(1-0)(1-2)} = \frac{x^2-2x}{-1} \\
\Delta_2(x) &= \frac{(x-0)(x-1)}{(2-0)(2-1)} = \frac{x^2-x}{2} \\
P(x) &= m_0\Delta_0(x) + m_1\Delta_1(x) + m_2\Delta_2(x) \\
&= 4\Delta_0(x) + 3\Delta_1(x) + 2\Delta_2(x) \\
&= -x + 4
\end{aligned}$$

[Note that all arithmetic is over GF(5), so for example $2^{-1} \equiv 3 \pmod 5$.] For the final message we need to add 2 redundant points of P. Since 3 and 4 are the only points in GF(5) that we have not used yet, we compute $P(3) = 1, P(4) = 0$, and so our message is $(4, 3, 2, 1, 0)$.

(b) The message received is $(c'_0, c'_1, c'_2, c'_3, c'_4) = (0, 3, 2, 1, 0)$. Let $R(x)$ be the function such $R(i) = c'_i$ for $0 \le i < 5$. Let $E(x) = x + b_0$ be the error-locator polynomial, and $Q(x) = P(x)E(x) =$

$a_3x^3 + a_2x^2 + a_1x + a_0$. Since $Q(i) = P(i)E(i) = R(i)E(i)$ for $1 \le i < 5$, we have the following equalities $\pmod 5$:

$$\begin{aligned} Q(0) &= 0E(0) \\ Q(1) &= 3E(1) \\ Q(2) &= 2E(2) \\ Q(3) &= 1E(3) \\ Q(4) &= 0E(4) \end{aligned}$$

They lead to the following system of linear equations:

$$\begin{array}{rcrcrcrcrcl} & & & & & & a_0 & & & = & 0 \\ a_3 & + & a_2 & + & a_1 & + & a_0 & - & 3b_0 & = & 3 \\ 8a_3 & + & 4a_2 & + & 2a_1 & + & a_0 & - & 2b_0 & = & 4 \\ 27a_3 & + & 9a_2 & + & 3a_1 & + & a_0 & - & b_0 & = & 3 \\ 64a_3 & + & 16a_2 & + & 4a_1 & + & a_0 & & & = & 0 \end{array}$$

(c) From the solution, we know

$$\begin{aligned} Q(x) &= a_3x^3 + a_2x^2 + a_1x + a_0 = -x^2 + 4x, \\ E(x) &= x + b_0 = x. \end{aligned}$$

Since $Q(x) = P(x)E(x)$, the recipient can compute $P(x) = Q(x)/E(x) = -x+4$ [note that this is the same polynomial $P(x)$ from part (a) used by the sender]. The recipient may deduce the location of the error from $E(x)$ as follows. There is only one error at location e_1, we have $E(x) = (x - e_1) = x$, so $e_1 = 0$ and the error is at position 0. To correct the error we evaluate $P(0) = 4$. Since the other two positions m_1, m_2 of the message are uncorrupted, we recover the original message $(m_0, m_1, m_2) = (4, 3, 2)$.

3 Error-Correcting Codes

(a) Recall from class the error-correcting code for erasure errors, which protects against up to k lost packets by sending a total of $n+k$ packets (where n is the number of packets in the original message). Often the number of packets lost is not some fixed number k, but rather a *fraction* of the number of packets sent. Suppose we wish to protect against a fraction α of lost packets (where $0 < \alpha < 1$). At least how many packets do we need to send (as a function of n and α)?

(b) Repeat part (a) for the case of general errors.

Solution:

(a) Suppose we send a total of m packets (where m is to be determined). Since at most a fraction α of these are lost, the number of packets received is at least $(1-\alpha)m$. But in order to reconstruct the polynomial used in transmission, we need at least n packets. Hence it is sufficient to have $(1-\alpha)m \geq n$, which can be rearranged to give $m \geq n/(1-\alpha)$.

(b) Suppose we send a total of $m = n+2k$ packets, where k is the number of errors we can guard against. The number of corrupted packets is at most αm, so we need $k \geq \alpha m$. Hence $m \geq n+2\alpha m$. Rearranging gives $m \geq n/(1-2\alpha)$.

Note: Recovery in this case is impossible if $\alpha \geq 1/2$.