CS 70 SPRING 2007 — DISCUSSION #6

ALEX FABRIKANT

1. Administrivia

(1) Course Information
- Next homework is **due February 27th** at 2:30pm in 283 Soda Hall; please put your section time on your homework.
- Next Tuesday, Feb 27th, instead of posting a homework, we will post Midterm practice problems. You do not have to turn in your answers, but we highly recommend working through them
- Next Friday, Mar 2nd, 5pm-7pm, there'll be a midterm review session in 306 Soda. We'll go over any midterm practice problems per request, and whatever other questions you guys bring in.
- Midterm 1 will be on Tuesday, March 6th, in class. You'll be allowed **one single-sided** $8.5'' \times 11''$ **page of notes**.

2. Error Correction: Erasure channels (Known Error Positions)

In lecture, we learned an error-correcting scheme that allows us to correct k errors by adding k more characters to the transmitted message. This ability is quite useful in communications where we know we lost characters (such as noise or random disconnections on a line where it is very clear what noise is, etc...).

Let us review how this process works. Use the following encoding:

0	1	2	3	4	5	6	7	8	9	10
E	I	T	L	F	O	K	S	B	P	A

We'll work modulo 11, with messages of size 3 with at most 1 erasure (thus, this requires that the encoding be of length 4).

Exercise 1. Reconstruct the following two size-3 messages encoded to protect against at most 1 erasure. Remember that you are solving for a quadratic polynomial modulo 11.

(1) The first encoding is: A_OP.

Solution: This encoding is represented as $(10, \_, 5, 9)$. Since this is an encoding of a message of size 3 with at most one erasure, we will need to reconstruct a polynomial in $\mathbb{Z}_{11}$ that goes through $(1, 10)$, $(3, 5)$, and $(4, 9)$. Lagrange interpolation gives:

$$\begin{aligned}
x_1 \equiv 1\,; \quad y_1 \equiv 10\,; \quad \Delta_1(x) &\equiv \frac{(x-3)(x-4)}{(1-3)(1-4)} \equiv \frac{(x-3)(x-4)}{6} \equiv 2(x-3)(x-4) \\
x_2 \equiv 3\,; \quad y_2 \equiv 5\,; \quad \Delta_2(x) &\equiv \frac{(x-1)(x-4)}{(3-1)(3-4)} \equiv \frac{(x-1)(x-4)}{9} \equiv 5(x-1)(x-4) \qquad \pmod{11} \\
x_3 \equiv 4\,; \quad y_3 \equiv 9\,; \quad \Delta_3(x) &\equiv \frac{(x-1)(x-3)}{(4-1)(4-3)} \equiv \frac{(x-1)(x-3)}{3} \equiv 4(x-1)(x-3)
\end{aligned}$$

Date: February 23, 2007.

The author gratefully acknowledges the TA's of CS70 Past for the use of their previous notes: Amir Kamil, Chris Crutchfield, David Garmire, Lorenzo Orecchia, and Ben Rubinstein. Their notes form the basis for this handout.

Thus, the whole polynomial is:

$$P(x) \equiv \sum_{i=1}^{3} y_i \Delta_i(x) \equiv 10 \cdot 2(x-3)(x-4) + 5 \cdot 5(x-1)(x-4) + 9 \cdot 4(x-1)(x-3)$$
$$\equiv 9(x-3)(x-4) + 3(x-1)(x-4) + 3(x-1)(x-3)$$

To compute the missing character, we just need to compute $P(2)$:

$$P(2) \equiv 9(2-3)(2-4) + 3(2-1)(2-4) + 3(2-1)(2-3)$$
$$\equiv 9 \cdot (-1) \cdot (-2) + 3 \cdot 1 \cdot (-2) + 3 \cdot 1 \cdot (-1)$$
$$\equiv 18 - 6 - 3 \equiv 9$$

Thus, the erased letter is represented by 9; that is, it's P. Since our protocol sets up P so that the original message is $(P(1), P(2), P(3))$, we get that the original message here was "APO".

(2) The second encoding is: _LOP. Reconstruct the message.

Concatenate the three-character chunks together. Do you see the message?

Notice how we recycled the "message characters" from last week's secret-sharing exercise. This is to remind you that the secret-sharing protocol from last week is based on the same technology as error-correction from this week.

3. Error Correction: Berlekamp-Welsch Algorithm (Unknown Error Positions)

In modern day communications, most data is transmitted in the digital domain. This change makes it hard to determine what is noise or an omission in the data, so the error correcting scheme must be improved. Specifically, the error correcting scheme must tell you where the errors occured and allow you to decode the original message. In fact, CDs and other storage devices contain a large amount of redundant data.

The main idea is to add k more characters to the message such that message may be decoded. To follow this procedure, after we receive *some* function $R(x)$, represented by values $R(1), \ldots, R(n+2k)$:

(1) We define another polynomial $E(x) = (x - e_1)(x - e_2)...(x - e_k)$ whose zeros are at the positions of the errors. Note that we don't know anything about the actual values of $e_1, \ldots, e_k$, and thus know basically nothing about E other than its degree and that its leading coefficient is 1. That is, $E(x) = x^k + b_{k-1}x^{k-1} + \ldots + b_1 x + b_0$, for some unknown $(b_0, \ldots, b_{k-1})$.

(2) Observe that $R(x)E(x) = Q(x) = P(x)E(x)$. Let Q, a $n+k-1$-degree polynomial, be represented by $\sum_{i=0}^{n+k-1} a_i x^i$, where the a_i's are another $n+k$ unknowns.

(3) Since we know the values of R at each x between 1 and $n+2k$, we can set up a linear system of $n+2k$ equations over the $n+2k$ unknowns (the a_i's and the b_i's) as follows:

$$\begin{cases} R(1)E(1) = Q(1) \\ \vdots \quad \vdots \\ R(n+2k)E(n+2k) = Q(n+2k) \end{cases}$$

$$\begin{cases} R(1)b_0 + R(1)b_1 \cdot 1^1 + \ldots + R(1)b_{k-1} \cdot 1^{k-1} + R(1) \cdot 1^k = a_0 + a_1 \cdot 1^1 + \ldots + a_{n+k-1} \cdot 1^{n+k-1} \\ R(2)b_0 + R(2)b_1 \cdot 2^1 + \ldots + R(2)b_{k-1} \cdot 2^{k-1} + R(2) \cdot 2^k = a_0 + a_1 \cdot 2^1 + \ldots + a_{n+k-1} \cdot 2^{n+k-1} \\ \vdots \quad \vdots \\ R(n+2k)b_0 + \ldots + R(n+2k)(n+2k)^k = a_0 + \ldots + a_{n+k-1}(n+2k)^{n+k-1} \end{cases}$$

Exercise 2. Suppose you receive a 3-character message, encoded to protect against 1 error, thus containing $3 + 2 \cdot 1 = 5$ characters: "PETKA" (that is, $(9, 0, 2, 6, 10)$. Use the Berlekamp-Welsch algorithm to find out where the error was located and what the original message contained.

4. Challenge problem

Exercise 3. Suppose you have a channel over which you can send m bytes, but up to k of them — and you don't know which ones — may *disappear* when the receiver gets the transmission (which will thus consist of anywhere between $m - k$ and m bytes). Use the tools from class to construct a reasonable error-correcting code that transmits a message of length n over this channel so that the receiver can definitely reconstruct it completely.