UW Math 135 **Error-correction codes** **July 29, 2005**

Ashwin Nayak

1. Error-correction

Digital data stored in computers or transmitted over computer networks are constantly subject to error due to the physical medium in which they are stored (such as magnetic disks) or the process by which they are transmitted (such as electrical signals). Error-correction codes are a means of introducing redundancy in the data so that even if part of it is corrupted or completely lost, the original data can be recovered.

We will study a particular kind of error-correction code, the Reed-Solomon code, which is constructed by using polynomials over finite fields. We will begin by studying a simple code, the repetition code, to illustrate some of the concepts involved in error-correction.

Data on computers are stored in the form of bit-sequences. Such data can be thought of as a sequence of numbers in the 2-element field $\mathbb{Z}_2$. More generally, we may represent data (or messages) as sequences of numbers from the field $\mathbb{Z}_q$, where q is a prime. In the following, we will assume that our message sequence has length n.

We will restrict our attention to a particular kind of error, in which any number may be mapped arbitrarily to another number in Z_q. For the bit case $q = 2$, this kind of error corresponds to a bit-flip, where a 0 is mapped to a 1 and vice-versa.

If a message (a bit-sequence) is transmitted as-is, even a single error (i.e., a bit flip in a single coordinate) would result in transforming the message into a different message. In the simplest case, when a single bit is transmitted, and a 0 is received, we cannot be sure if a zero was sent and no error occurred, or if a 1 was sent and it got flipped.

The reason that raw messages are so vulnerable to error is that different messages can be very close to each other: that the numbers in a few of the coordinates can be changed to change one message to another. If however, we could ensure, by introducing redundancy in our messages, that the transmitted (encoded) messages were different in many coordinates, they would be immune to a small number of errors.

Definition 1 (Hamming distance)**.** *The* Hamming distance *between two sequences $x, y \in \mathbb{Z}_q^n$ is defined as the number of coordinates i such that $x_i \neq y_i$. This distance is denoted by $d(x, y)$.*

For example, the (Hamming) distance between the bit strings 01101 and 10111 in $\mathbb{Z}_2^5$ is 3, and the distance between the sequences $(2, 5, 3, 3, 1, 0)$ and $(3, 6, 3, 4, 2, 1)$ in $\mathbb{Z}_7^6$ is 5.

It is straightforward to verify that this measure has all the properties required of a distance metric:

(1) $d(x, y) \geq 0$, and $d(x, y) = 0$ iff $x = y$,
(2) It is symmetric $d(x, y) = d(y, x)$, and

(3) It satisfies the triangle inequality $d(x, y) \leq d(x, z) + d(z, y)$.

Definition 2. *An (m,n,d)*-error-correction code *is a subset $C \subseteq \mathbb{Z}_q^m$ of size q^n such that $d(x, y) \geq d$ for every pair of distinct elements $x, y \in C$. The parameter d is called the* minimum distance *of the code, and elements of C are called* codewords.

Proposition 1. *A code $C \subseteq \mathbb{Z}_q^m$ with minimum distance d can correct $\lfloor (d-1)/2 \rfloor$ errors.*

Proof. Suppose that a codeword from C is transmitted, and $\lfloor (d-1)/2 \rfloor$ of the coordinates are corrupted. The original message can be recovered uniquely by choosing the codeword in C that is closest to the received sequence: There is only one codeword x at distance at most $\lfloor (d-1)/2 \rfloor$ from received sequence z. If not, suppose there is another such codeword y. Then $d(x, y) \leq d(x, z) + d(z, y) \leq (d-1)/2 + (d-1)/2 < d$, which contradicts the fact that the minimum distance is d. □

2. Hamming code

We now give a simple example of an error-correction code: a *Hamming* or *repetition* code. In this example, redundancy is introduced directly into a message by repeating each bit (or number in $\mathbb{Z}_q$) three times.

For example, consider messages which are 3-bit strings, so $n = 3$. Each bit in the string is repeated three times, so the resulting message length is $m = 9$.

Message	Codeword
000	000000000
001	000000111
010	000111000
011	000111111
100	111000000
101	111000111
110	111111000
111	111111111

Note that the minimum distance between the messages may be 1, but the minimum distance of the repetition code is 3. This means that any 1 bit error in the codewords may be corrected. Indeed, if we look at the three blocks of three bits each in a received message, we can recover the original bit by taking the most common bit among the three. If no error has occurred, the three bits would be 000 or 111, and if a single bit flip has occurred, the bits would be 100, 010, or 001 in case a zero was encoded, and 011, 101, or 110 if a one was encoded. In either case, the most common bit gives us the correct answer.

The repetition code is not very efficient in that if we expect a large number t of errors to occur, then we would have to repeat each bit (or number) in the message $2t+1$ times. In particular, if we expected a constant fraction of the coordinates to get corrupted, then $t = \delta m$, where $\delta \in (0,1)$ is a constant, and we would have $m = (2t+1)n = (2\delta m + 1)n$. Since $m \geq n$, this would only be possible for $n \leq 1/(2\delta)$. In other words, we'd only be able to send constant size messages. The Reed-Solomon codes we will look at next overcomes this inefficiency.

3. Reed-Solomon codes

In the construction of the Reed-Solomon code, we assume that messages are erpresented as sequences of length n of numbers in $\mathbb{Z}_q$, where q is a prime. We will encode each of these n-sequences into a code of length m as follows.

With each sequence $a = (a_{n-1}, a_{n-2}, \ldots, a_1, a_0) \in \mathbb{Z}_q^n$, we associate a degree $n-1$ polynomial f_a defined as follows:

$$f_a(x) \;=\; a_{n-1}x^{n-1} + a_{n-2}x^{n-2} + \ldots + a_1 x + a_0.$$

The message sequence of values of f_a at m distinct points in $\mathbb{Z}_q$, say the points $m-1, m-2, \ldots, 1, 0$:

$$\hat{a} \;=\; (f_a(m-1), f_a(m-2), \ldots, f_a(1), f_a(0)).$$

Thus in our construction, we need that $q \geq m$. The choice and order of the m points is completely arbitrary, but should be the same for every message encoded in this manner.

For example, consider messages of length 2 over $\mathbb{Z}_7$, i.e., $n = 2$, and $q = 7$. There are 49 such possible messages. Suppose that we encode these into sequences of length $m = 6$. The polynomials and codewords corresponding to the three messages $(1,6), (4,3), (4,6)$ would be (for the points $6, 5, \ldots, 1$):

Message a	Polynomial f_a	Codeword $\hat{a}$
$(1,6)$	$x+6$	$(5,4,3,2,1,0)$
$(4,3)$	$4x+3$	$(6,2,5,1,4,0)$
$(4,6)$	$4x+6$	$(2,5,1,4,0,3)$

Note that the distance between the codewords is at least 5. This is not a coincidence, because each of the codewords corresponds to a line, and two distinct lines are either parallel (as in the last two cases) or they intersect in at most one point (as in the first two cases).

More generally, we can prove the following.

Proposition 2. *The minimum distance of the Reed-Solomon code over $\mathbb{Z}_q$ which maps length n messages to length m codewords is at least $d = m - n + 1$.*

Proof. Consider two messages $a \neq b \in \mathbb{Z}_q^n$. They correspond to two distinct polynomials $f_a(x)$ and $f_b(x)$. Now, each coordinate where the codewords $\hat{a}, \hat{b}$ are the same corresponds to a point x_0 where $f_a(x_0) = f_b(x_0)$. In other words, the point x_0 is a root of the polynomial $f(x) = f_a(x) - f_b(x)$. Since the two polynomials are distinct, $f(x)$ is not identically 0. Therefore, since it has degree at most $n-1$, $f(x)$ has at most $n-1$ roots in $\mathbb{Z}_q$. Thus, the number of coordinates where $\hat{a}, \hat{b}$ differ is at least $m-(n-1)$, i.e., the minimum distance is at least $m-n+1$. □

Reed-Solomon codes are very efficient, in that they allow a constant fraction of errors to occur in the codewords while still being only a cosntant factor longer than the messages: suppose that we need to be able to recover from $t = \delta m$ errors, we need

$$\begin{aligned} d &= 2\delta m + 1 \\ \Leftrightarrow\ m - n + 1 &= 2\delta m + 1 \\ \Leftrightarrow\ m &= \frac{n}{1-2\delta}. \end{aligned}$$