Error Correcting Codes: Combinatorics, Algorithms and Applications **(Fall 2007)**

Lecture 3: Error Correction and Distance

August 31, 2007

Lecturer: Atri Rudra *Scribe: Michael Pfetsch & Atri Rudra*

The following topics were discussed in the last lecture:

- Shannon and Hamming noise models.
- The $C_{3,rep}$ repetition code can correct ≤ 1 errors and has a rate of $R = \frac{1}{3}$.
- The $C_{\oplus}$ parity code cannot correct even 1 error and has a rate of $R = \frac{4}{5}$.

The last two points confirmed our intuition that one can correct more errors if the underlying code has more redundancy. In today's lecture we will look a bit more closely at the parity code $C_{\oplus}$, which among other things will motivate another important parameter of codes called the *distance*.

1 A closer look at $C_{\oplus}$

Last lecture we saw that $C_{\oplus}$ cannot correct even 1 error. However, we will now see that $C_{\oplus}$ can *detect* one error. Consider the following algorithm. Let $\mathbf{y} = (y_1, y_2, y_3, y_4, y_5)$ be the received word– compute $b = y_1 \oplus y_2 \oplus y_3 \oplus y_4 \oplus y_5$ and declare an error if $b = 1$. Note that when no error has occurred during transmission, $y_i = x_i$ for $1 \leq i \leq 4$ and $y_5 = x_1 \oplus x_2 \oplus x_3 \oplus x_4$, in which case $b = 0$ as required. If there is a single error then either $y_i = x_i \oplus 1$ (for exactly one $1 \leq i \leq 4$) or $y_5 = x_1 \oplus x_2 \oplus x_3 \oplus x_4 \oplus 1$. It is easy to check that in this case $b = 1$. In fact, one can extend this argument to to obtain the following result.

Proposition 1.1. *The parity code $C_{\oplus}$ can* detect *an odd number of errors.*

Let us now revisit the example that showed that one cannot correct 1 error using $C_{\oplus}$. Consider the two codewords in $C_{\oplus}$, $\mathbf{u} = 00000$ and $\mathbf{v} = 10001$ (which are codewords corresponding to messages 0000 and 1000 respectively). Now consider the two scenarios when $\mathbf{u}$ and $\mathbf{v}$ are transmitted and a single error occurs resulting in the received word $\mathbf{r} = 10000$ in both cases. Now given the received word $\mathbf{r}$ and the fact that at most one error can occur, the decoder has no way of knowing whether the original transmitted codeword was $\mathbf{u}$ or $\mathbf{v}$. Looking back at the example, it is clear that the decoder gets "confused" because the two codewords $\mathbf{u}$ and $\mathbf{v}$ do not differ in a lot of positions. This notion is formalized in the next section.

2 Distance of a code

We now define a notion of distance (due to Hamming) that captures the concept that the two vectors $\mathbf{u}$ and $\mathbf{v}$ are "close-by."

Definition 2.1 (Hamming distance). *Given* $\mathbf{u}, \mathbf{v} \in \Sigma^n$ *(i.e. two vectors of length* n*) the* Hamming distance *between* $\mathbf{u}$ *and* $\mathbf{v}$*, denoted by* $\Delta(\mathbf{u}, \mathbf{v})$*, is defined to be the number of positions in which* $\mathbf{u}$ *and* $\mathbf{v}$ *differ.*

Note that the definition of Hamming distance just depends on the *number* of differences and not the nature of the difference. For example, for the vectors $\mathbf{u}$ and $\mathbf{v}$ from the previous section, $\Delta(\mathbf{u}, \mathbf{v}) = 2$, which is equal to the Hamming distance $\Delta(\mathbf{u}, \mathbf{w})$, where $\mathbf{w} = 01010$, even though the vectors $\mathbf{v}$ and $\mathbf{w}$ are not equal.

Armed with the notion of Hamming distance, we now define another important parameter of a code.

Definition 2.2 (Minimum distance). *Let* $C \subseteq \Sigma^n$*. The* minimum distance *(or just* distance*) of* C *is defined to be*

$$d = \min_{c_1 \neq c_2 \in C} \Delta(c_1, c_2)$$

It is easy to check that the repetition code $C_{3,rep}$ has distance 3. We now claim that the distance of $C_{\oplus}$ is 2. This is a consequence of the following observations. If two messages m_1 and m_2 differ in at least two places then $\Delta(C_{\oplus}(m_1), C_{\oplus}(m_2)) \geq 2$ (by only looking at the first four bits of the codewords). If two messages differ in exactly one place then the parity bits in the corresponding codewords are different which implies a Hamming distance of 2 between the codewords. Thus, $C_{\oplus}$ has smaller distance than $C_{3,rep}$ and can correct less number of errors than $C_{3,rep}$, which seems to suggest that a larger distance implies greater error correcting capabilities. The next result formalizes this intuition.

Proposition 2.3. *The following statements are equivalent for a code* C*:*

1. *C has minimum distance d,*
2. *If d is odd, C can correct $(d-1)/2$ errors.*
3. *C can detect $d-1$ errors.*
4. *C can correct $d-1$ erasures.*[1]

Remark 2.4. *Property (2) above for even d is slightly different. In this case, one can correct up to $\frac{d}{2} - 1$ errors but cannot correct $\frac{d}{2}$ errors. Proof of this claim is similar to that of Proposition 2.3. Note that this implies that if a code C is t-error correctable then it either has a distance of $2t+1$ or $2t+2$.*

[1] In the erasure noise model, the receiver *knows* where the errors have occurred. In this model, an erroneous symbol is denoted by "?", with the convention that any non-? symbol is a correct symbol.

Before we prove Proposition 2.3, let us apply it to the codes $C_{\oplus}$ and $C_{3,rep}$ which have distances of 2 and 3 respectively. Proposition 2.3 implies the following facts that we have already proved:

- $C_{3,rep}$ can correct 1 errors.
- $C_{\oplus}$ can detect 1 error (but cannot correct 1 error).

The proof of Proposition 2.3 will need the following decoding function. Maximum likelihood decoding (MLD) is a well studied decoding method for error correcting codes, which outputs the codeword closest to the received word in Hamming distance (with ties broken arbitrarily). In particular, the MLD function denoted by $D_{MLD} : \Sigma^n \to C$ is defined as follows. For every $\mathbf{y} \in \Sigma^n$,

$$D_{MLD}(\mathbf{y}) = \arg\min_{c \in C} \Delta(c, \mathbf{y}).$$

Proof of Proposition 2.3 We will complete the proof in two steps. First, we will show that if property 1 is satisfied then so are properties 2,3 and 4. Then we show that if property 1 is not satisfied then none of properties 2,3 or 4 hold.

Assume C has distance d. We first prove 2 (for this case assume that $d = 2t+1$). We now need to show that there exists a decoding function such that for all error patterns with at most t errors it always outputs the transmitted message. We claim that the MLD function has this property. Assume this is not so and let c_1 be the transmitted codeword and let $\mathbf{y}$ be the received word. Note that

$$\Delta(\mathbf{y}, c_1) \leq t. \tag{1}$$

As we have assumed that MLD does not work, $D_{MLD}(\mathbf{y}) = c_2 \neq c_1$. Note that by the definition of MLD,

$$\Delta(\mathbf{y}, c_2) \leq \Delta(\mathbf{y}, c_1). \tag{2}$$

Consider the following set of inequalities:

$$\begin{align} \Delta(c_1, c_2) &\leq \Delta(c_2, \mathbf{y}) + \Delta(c_1, \mathbf{y}) \tag{3} \\ &\leq 2\Delta(c_1, \mathbf{y}) \tag{4} \\ &\leq 2t \tag{5} \\ &= d - 1, \tag{6} \end{align}$$

where (3) follows from the triangle inequality, (4) follows from (2) and (5) follows from (1). (6) implies that the distance of C is at most $d-1$, which is a contradiction.

We now show that property 3 holds, that is, we need to describe an algorithm that can successfully detect whether errors have occurred during transmission (as long as the total number of errors is bounded by $d-1$). Consider the following error detection algorithm: check if the received word $\mathbf{y} = c$ for some $c \in C$ (this can be done via an exhaustive check). If no errors occurred during transmission, $\mathbf{y} = c_1$, where c_1 was the transmitted codeword and the algorithm above will accept (as it should). On the other hand if $1 \leq \Delta(\mathbf{y}, c_1) \leq d-1$, then by the the fact that the distance of C is d, $\mathbf{y} \notin C$ and hence the algorithm rejects, as required.

Finally, we prove that property 4 holds. Let $\mathbf{y} \in (\Sigma \cup \{?\})^n$ be the received word. First we claim that there is a unique $c = (c_1, \ldots, c_n) \in C$ that agrees with $\mathbf{y}$ (i.e. $y_i = c_i$ for every i such that $y_i \neq ?$). (For the sake of contradiction, assume that this is not true, i.e. there exists two distinct codewords $c^1, c^2 \in C$ such that both c^1 and c^2 agree with $\mathbf{y}$ in the unerased positions. Note that this implies that c^1 and c^2 agree in the positions i such that $y_i \neq ?$. Thus, $\Delta(c^1, c^2) \leq |\{i | y_i = ?\}| \leq d-1$, which contradicts the assumption that C has distance d.) Given the uniqueness of the codeword $c \in C$ that agrees with $\mathbf{y}$ in the unerased position, here is an algorithm to find it: go through all the codewords in C and output the desired codeword.

For the other direction of the proof, assume that property 1 does not hold, that is, C has distance $d-1$. We now show that property 2 cannot hold, that is, for every decoding function there exists a transmitted codeword c_1 and a received word $\mathbf{y}$ (where $\Delta(\mathbf{y}, c_1)$) such that the decoding function cannot output c_1. Let $c_1 \neq c_2 \in C$ be codewords such that $\Delta(c_1, c_2) = d-1$ (such a pair exists as C has distance $d-1$). Now consider the vector $\mathbf{y}$ such that $\Delta(\mathbf{y}, c_1) = \Delta(\mathbf{y}, c_2) = (d-1)/2$. (Such a $\mathbf{y}$ exists as d is odd and by the choice of c_1 and c_2.) Now, since $\mathbf{y}$ could have been generated if *either* of c_1 or c_2 were the transmitted codeword, no decoding function can work in this case.[2]

For the remainder of the proof, assume that the transmitted word is c_1 and there exists another codeword c_2 such that $\Delta(c_2, c_1) = d-1$. To see why property 3 is not true, let $\mathbf{y} = c_2$. In this case, either the error detecting algorithm detects no error or it declares an error when c_2 is the transmitted codeword and no error takes place during transmission.

We finally argue that property 4 does not hold. Let $\mathbf{y}$ be the received word in which the positions that are erased are exactly those where c_1 and c_2 differ. Thus, given $\mathbf{y}$ both c_1 and c_2 could have been the transmitted codeword and no algorithm for correcting (at most $d-1$) erasures can work in this case. ■

[2] Note that this argument is just a generalization of the argument that $C_\oplus$ cannot correct 1 error.