15-853: Algorithms in the Real World
Fall 2019

Lecture 3

September 10 2019

Lecturer: Rashmi Vinayak *Scribe: Ines Potier, Kartik Chitturi*

3.1 Recap

3.1.1 Block Codes

The first part of the lecture was a recap of **Block Codes**, which work as follows:

Both message and codeword have a fixed size. We define **k** to be the size of the message, **n** the size of the codeword, **q** the size of the alphabet ($q = |\Sigma|$).

We associate the symbol **C** ($C \subseteq \Sigma^n$) to the set of codewords, Δ**(x,y)** to number of positions such that $x_i \neq y_i$. Finally, we define the minimum distance **d**:

$$d = \min(\Delta(x,y) : x,y \in C, x \neq y)$$

A code is described as: **(n,k,d)**

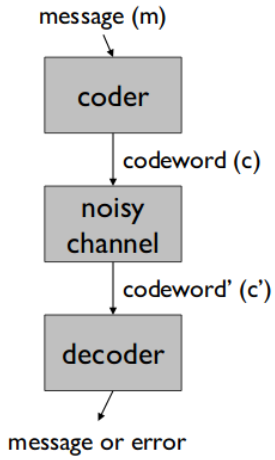

Figure 3.1. Block Code

Theorem 3.1. *A code C with minimum distance "d" can:*

- *detect any (d-1) errors*
- *recover any (d-1) erasures*
- *correct any* $\frac{d-1}{2}$ *errors*

Corollary 3.2. *For s-bit error detection or erasure recovery, d must satisfy* $d \geq s+1$*, for s-bit error correction:* $d \geq 2s+1$*, and to correct a erasures and b errors* $d \geq a+2b+1$

3.1.2 Clarifications

The second part of the recap treated some general clarifications. The **Error model** described in the past and current lectures has two fundamental properties:

- Arbitrary/ adversarial errors

- When considering s errors (for example in the phrase *correct any s-errors*), we consider that there can be **up to** s **errors** in any codeword considered
- Symmetry across alphabet value. That is to say, it is equally probable to go from any alphabet value to any other possible value. There is no notion of preference.

Also, some clarifications were made about the **Role of minimum distance encoding**. First, it is important to understand that *minimum distance decoding* is not a decoding algorithm used in practice. It is more of a global description of what we need to do to decode. Minimum distance decoding assumes that the closest codeword is the correct codeword. A naïve way of achieving min-dist decoding is brute force search across all codewords. There may be more efficient ways of getting to the closest codeword when if the code has structure.

A *sphere of Hamming radius s centered at c* is formed by all the possible values a codeword c with at most s errors can take. If there exist two codewords of c, c' such that the radius s Hamming spheres centered at c and c' overlap or intersect, then no decoding algorithm can **guarantee** a correct decoding in the presence of s errors for all possible codeword values.

3.1.3 Linear Codes

The third part of the Recap concerned **Linear Codes**.

Theorem 3.3. *Let Σ be a field. Then Σ^n is a vector space*

Definition 2 *Linear Code*: C is a linear code iff it is a linear subspace of Σ^n of dimension k.

If C is a linear subspace then there exist at least one set of k independent vectors $v_i \in \Sigma^n$ $(1 \leq i \leq k)$ that span the subspace. This a called a **basis**. Every codeword can then be written as follows:

$$c = a_1 v_1 + a_2 v_2 + ... + a_k v_k \text{ where } a_i \in \Sigma$$

Corollary 3.4. *For a linear code, the linear combination of two codewords is a codeword*

Corollary 3.5. *For a linear code, the minimum distance is equal to the weight of the least-weight non-zero codeword.*

3.1.4 Generator and Parity Check Matrices

The last part of the recap was about **Generator and Parity Check Matrices**

Definition 1 A **generator matrix** is a $k \times n$ matrix **G** such that $\mathrm{C} = \{\mathrm{xG} \mid x \in \Sigma^k\}$

Definition 2 A **parity check matrix** is an $(n-k)\times n$ matrix **H** such that $\mathrm{C} = \{y \in \Sigma^n \mid \mathrm{H}y^T = 0\}$ (codewords are the null space of H).

These **always exist for linear codes**

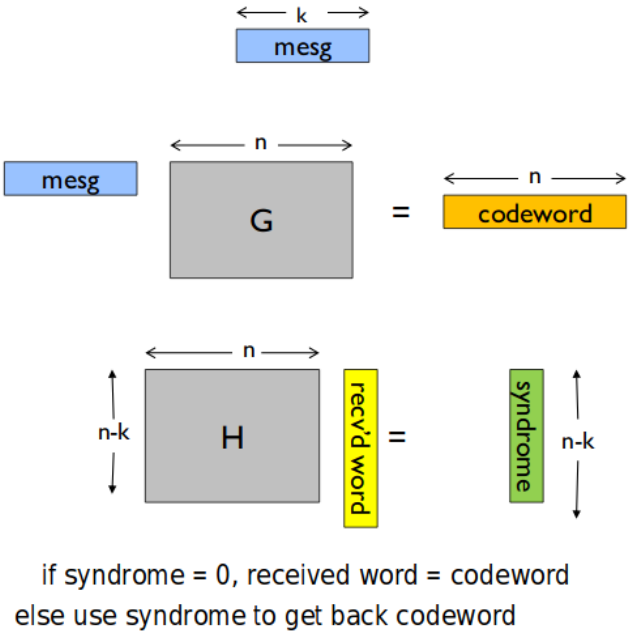

Figure 3.2. Use of generator and parity matrices

3.2 Standard Forms

The generator matrix **G** is said to be in **Standard Form** when it can be written as follows:

$$G = \begin{bmatrix} I_k & A \end{bmatrix}$$

G is of size $k \times n$, I_k is the identity matrix of size $k \times k$ and A is of size $k \times (n-k)$

Question: *Is the generator matrix unique for a code?*
Answer: There exists more than one basis for a linear space. As a result, there also exists

several form of the generator matrix for a code.

The standard form of a generator matrix G can be very useful to find the associated parity check matrix. Indeed,

Theorem 3.6. *For binary codes, if G is in standard form* $[I_k A]$ *then:*

$$H = \begin{bmatrix} A^T & I_{n-k} \end{bmatrix}$$

Proof: This proof contains two parts.

First, let us prove $Hy^T = 0 \; \forall y \in C$. Let us consider $y \in C$. We know that

$$Hy^T = 0 \Leftrightarrow H(xG)^T = 0$$

for an x which satisfies $xG = y$. This must exist because $y \in C$. We can then write:

$$\begin{bmatrix} A^T & I \end{bmatrix} \begin{bmatrix} I \\ A^T \end{bmatrix} x^T = (A^T + A^T)x^T = 0$$

Since we are considering a binary code, and therefore we have $A^T + A^T = 0$

Second, let us prove $Hy^T = 0 \Rightarrow y \in C$. Let us consider y such that $Hy^T = 0$. We have:

$$\begin{bmatrix} A^T & I \end{bmatrix} \begin{bmatrix} y_1^T \\ y_2^T \end{bmatrix} = 0 \Leftrightarrow A^T y_1^T + y_2^T = 0$$

Since we are considering a binary code:

$$A^T y_1^T + y_2^T = 0 \Leftrightarrow A^T y_1^T = y_2^T \Leftrightarrow y_2 = y_1 A$$

That finished the proof. Indeed,

$$y_1 \begin{bmatrix} I & A \end{bmatrix} = \begin{bmatrix} y_1 & y_1 A \end{bmatrix} = \begin{bmatrix} y_1 & y_2 \end{bmatrix}$$

□

N.B.: The proof only held for $\mathbb{F}_2$. In a general field, a minus needs to be added in H:

$$y_1 \begin{bmatrix} -A & I_{n-k} \end{bmatrix}$$

In the binary case, $-A = A$, but that is not true for other alphabets.

3.3 Dual Codes

Recall that a linear code is equivalent to a linear subspace of Σ^n of dimension k. For any subspace W of a vector space V, we may define the set of vectors $W^\perp$ consisting of all vectors that are orthogonal to the vectors in W. This set, $W^\perp$ is called the **orthogonal complement** of W. Formally,

$$W^\perp = \{v \in V | \langle w, v\rangle = 0 \quad \forall w \in W\}$$

Lemma 3.7. *If V is a space of finite dimension n over the field Σ and W is a subspace of dimension k, then $W^\perp$ is a space of dimension $n-k$ and $(W^\perp)^\perp = W$.*

Proof: It is fairly trivial to see that $W^\perp$ is a subspace of V. Suppose $w_1, w_2 \in W^\perp$ and $\alpha, \beta \in \Sigma$. We know that $\langle w_1, v\rangle = 0 \quad \forall v \in W$ and $\langle w_2, v\rangle = 0 \quad \forall v \in W$ so $\alpha w_1 + \beta w_2, v\rangle = \langle \alpha w_1, v\rangle + \langle \beta w_2, v\rangle = \alpha\langle w_1, v\rangle + \beta\langle w_2, b\rangle = 0 + 0 = 0$.

Now we demonstrate the dimension of $W^\perp$. Let β be an orthonormal basis of W and γ an orthonormal basis of $W^\perp$. Then $\beta \cup \gamma \subseteq V$. Now, let $\pi : V \to W$ be defined as $\pi(x) = \sum_{\beta_i}\langle x, \beta_i\rangle\beta_i$. Then for any vector $v \in V$, $v = \pi(v) + (v - \pi(v))$. However, for every basis vector β_j of W,

$$\begin{aligned}
\langle v - \pi(v), \beta_j\rangle &= \langle v - \sum_{\beta_i}\langle v, \beta_i\rangle\beta_i, \beta_j\rangle \\
&= \langle v, \beta_j\rangle - \sum_{\beta_i}(\langle v, \beta_i\rangle\langle\beta_i, \beta_j\rangle) \\
&= \langle v, \beta_j\rangle - \sum_{\beta_i}(\langle v, \beta_i\rangle\delta_{i,j}) \\
&= \langle v, \beta_j\rangle - \langle v, \beta_j\rangle = 0
\end{aligned}$$

Thus $(v - \pi(v)) \in W^\perp$. We find that for any vector $v \in$, $v = w_1 + w_2$ where $w_1 \in W$ and $w_2 \in W^\perp$. We know that for any spaces X, Y that $dim(X+Y) = dim(X) + dim(Y) - dim(X \cap Y)$. It is fairly easy to see that $W \cap W^\perp$ consists solely of the zero vector (no other vector is orthogonal to itself). Thus $dim(V) = dim(W + W^\perp) = dim(W) + dim(W^\perp)$. Thus $dim(W^\perp) = n - k$.

Now note that if $w_1 \in W$, then for all $w_2 \in W^\perp$, $\langle w_1, w_2\rangle$. Thus $(W^\perp)^\perp = W$. □

Now lets apply this to codes. If $C \subseteq \Sigma^n$ is a code, so is $C^\perp$. We call $C^\perp$ the dual code. Now by the lemma above, if C is a (n, k) code then $C^\perp$ is a $(n, n-k)$ code. Moreover, since H is defined by $C = \{y \in \Sigma^n | Hy^T = 0\}$, we see that H is in fact a generator for $C^\perp$. Additionally, since $(C^\perp)^\perp$, the parity check matrix of $C^\perp$, is the generator for C. Since we can swap columns of a generator matrix produces an equivalent code, we have the following lemma.

Lemma 3.8. *Suppose a binary code C has a generator matrix $G = [I_k \ A]$ and a parity check matrix $H = [A^T \ I_{n-k}]$. Then its dual has a generator matrix $G = [I_{n-k} \ A^T]$ and a parity check matrix $H = [A \ I_k]$.*

Now, a motivating example.

Lemma 3.9. *The dual of $(2^r - 1, 2^r - 1 - r, 3)$ Hamming codes are $(2^r - 1, r, 2^{r-1})$ codes known as **Hadamard Codes**.*

Proof: Applying Lemma 1.1, we see that the dimension of Hadamard codes is $2^r - 1 - (2^r - 1 - r) = r$. Now note the generator G of a Hadamard code has $2^r - 1$ columns and each column is of length $n - k = r$. Each column must be distinct and non-zero. There are a total of $2^r - 1$ distinct non-zero binary vectors of length r. Thus every non-zero r bit-vector appears as a column of G. We see that a codeword c of message m is simply the array $[\langle m, v_1\rangle, \ldots, \langle m, v_{2^r-1}\rangle]$ for every non-zero vector v_i.

Call the set of r bit-vectors $\{w\}_i$ (including the zero vector). We show that if $x \neq 0$, then for exactly half of $\{w\}_i$, we have $\langle x, w_i\rangle = 0$ and for the other half $\langle x, w_j\rangle = 1$. Thus, when we exclude the zero vector, we see the Hamming weight of *every* vector is $2^r/2 = 2^{r-1}$. Let $x \neq 0$ and denote the first bit of x as x_1. Without loss of generality, suppose $x_1 = 1$. Then for all w_i, $\langle x, w_i\rangle + \langle x, w_i + e_1\rangle = 1$ where e_1 is the length r vector with the first bit as 1 and all other bits 0. This is since

$$\begin{aligned}\langle x, w_i\rangle + \langle x, w_i + e_1\rangle &= \langle x, w_i + w_i + e_1\rangle \\ &= \langle x, e_1\rangle = x_1 = 1\end{aligned}$$

We know that w_i and $w_i + e_1$ must be distinct and that exactly one of $\langle x, w_i\rangle$ and $\langle x, w_i + e_1\rangle$ equals 1. □

Hadamard codes have a large distance but also have high redundancy (i.e. n/k is high). They are mainly used when the communication channel is very noisy and has high loss. For example, NASA's Mariner 9, which was an unmanned space probe that explored Mars, used Hadamard codes to communicate pictures and terrestial data back to Earth. It also allowed NASA to reprogram Mariner 9's code after it was already orbiting Mars.

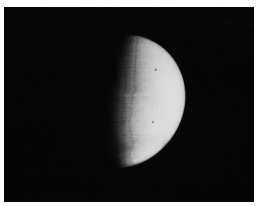

Figure 3.3. Picture of Mars taken by Mariner 9 and transmitted to Earth using a Hadamard Code

3.4 Finding Error Locations

For any $y \in \Sigma^n$, we call Hy^T a **syndrome**. Recall that if $Hy^T = 0$, no error occurred. In general, it is possible to find the location of errors in a codeword by creating a table that maps each syndrome to a set of error locations.

Theorem 3.10. *Assuming there can be no more than $(d-1)/2$ errors, every syndrome value corresponds to a unique set of error locations.*

The table mentioned above would have q^{n-k} entries, each of which has size up to n. Clearly, storing this table or looking up an entry in this table is not efficient for large values of $(n-k)$.

We can use a **standard array table** to help decode a codeword. To create the standard table, we create a row for each syndrome with the first row corresponding to the zero syndrome (no error). Then in each row, we list the codewords that map to that syndrome under H. Typically, a codeword of minimal Hamming weight of each row is placed first within the row.

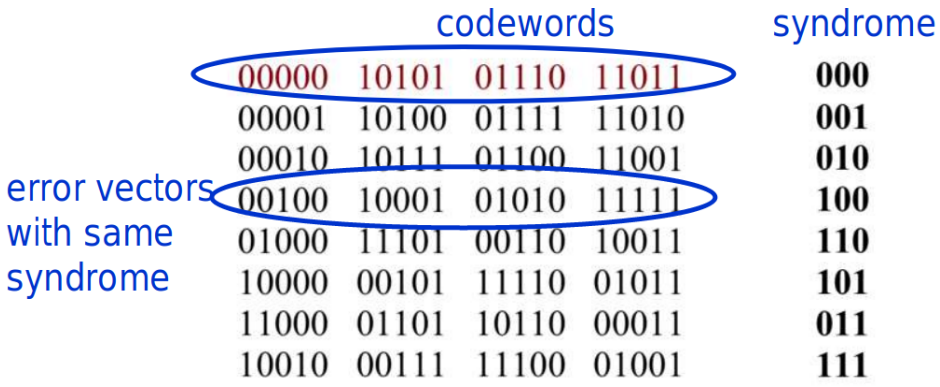

Figure 3.4. Example of a Standard Array Table

To decode a codeword, we find which row the codeword resides in. We then subtract of the leftmost element of that row from the codeword and this will give us a codeword in the first row (the syndromes with no error). We can then easily decode this (error-less) codeword.[1]

For example, suppose we received the codeword 10001. We see that it is in the 4^{th} row which has 00100 as the leftmost element. Subtracting this from 10001 gives 10101 which we see is an error-free codeword.

[1]We can use a bit of group theory to show this works. The error-less syndrome define the kernel of the homomorphism defined by H. Then each row of the standard table is a coset of this kernel. The left-most elements of each row are simply the representatives of these cosets. By subtracting the representative of a coset from an element of that coset, we land within the kernel. Since H is a subgroup of C, Lagrange's Theorem tells us that each syndrome corresponds to exactly $|C|/|H|$ codewords

Lemma 3.11. *Singleton Bound For every $(n, k, d)_q$ code, $n \geq k + d - 1$.*

Proof: Note that there are q^k codewords. Suppose we project each codeword onto the first $(k-1)$ coordinates. There are q^{k-1} points in this space. By the pigeonhole principle, there exists two distinct codewords $x \neq y$ such that both are projected to the same point. Thus the first $k-1$ bits of x and y must match. Then the distance between x and y is at most $n - (k-1) = n - k + 1$. Thus $d \leq n - k + 1$. □

Definition: Codes that meet the Singleton bound are called **Maximum Distance Separable (MDS)**. There are only two binary MDS codes: repetition codes and single-parity check codes.

Thus in order to get efficient codes, we need to move beyond the binary alphabet.

3.5 Group Theory

Definition: A **group** $(G, \star, e)$ is a set G with operator $\star$ such that

- **Closure:** For all a, b, $a \star b \in G$
- **Associativity:** For all $a, b, c \in G$, $a \star (b \star c) = (a \star b) \star c$
- **Identity**: There exists a unique e such that $a \star e = e \star a = a$ for all $a \in G$
- **Inverse:** For all $a \in G$, there exists a unique $b \in G$ such that $a \star b = b \star a = e$. We denote the inverse of a as a^{-1}. Note that in some groups there are elements such that $a = a^{-1}$

Additionally, if for all $a, b \in G$ $a \star b = b \star a$, we call the group **abelian**. Note that we require each pair of elements in the group to commute. A group where only some of the elements commute is not abelian (in fact note that in every group the identity element commutes with all elements).

Some examples of groups include

- The integers, rationals, and reals under addition where 0 is the identity element and $-x$ is the inverse of x
- Nonzero reals and nonzero rationals under multiplication where 1 is the identity element and x^{-1} is the inverse of x. Note that we must exclude zero since 0^{-1} does not exist.
- Permutations over n elements with composition. This is called the **symmetric group on n elements**, S_n. Here the identity element is the trivial permutation (doing nothing). To find the inverse of s, we must decompose s into what are known as cycles and then reverse each cycle.

Groups which have a finite number of elements are called **finite groups**.