Administrivia

- Send email to be added to course mailing list. Critical!
- Sign up for scribing.
- Pset 1 out today. First part due in a week, second in two weeks.
- Course under perpetual development! Limited staffing. Patience and constructive criticism appreciated.

Hamming’s Problem (1940s)

- Magnetic storage devices are prone to making errors.
- How to store information (32 bit words) so that any 1 bit flip (in any word) can be corrected?

Simple solution:
- Repeat every bit three times.
- Works. To correct 1 bit flip error, take majority vote for each bit.
- Can store 10 “real” bits per word this way. Efficiency of storage $\approx 1/3$. Can we do better?

Hamming’s Solution - 1

- Break (32-bit) word into four blocks of size 7 each (discard four remaining bits).
- In each block apply a transform that maps 4 “real” bits into a 7 bit string, so that any 1 bit flip in a block can be corrected.
- How? Will show next.
- Result: Can now store 16 “real” bits per word this way. Efficiency already up to $\frac{1}{2}$.

$[7, 4, 3]$-Hamming code

- Will explain notation later.
- Let

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

- Encode $b = (b_0 b_1 b_2 b_3)$ as $b \cdot G$.
- Claim: If $a \neq b$, then $a \cdot G$ and $b \cdot G$ differ in at least 3 coordinates.
- Will defer proof of claim.
Hamming’s Notions

• Since codewords (i.e., \( \mathbf{b} \cdot G \)) differ in at least 3 coordinates, can correct one error.

• Motivates Hamming distance, Hamming weight, Error-correcting codes etc.

• Alphabet \( \Sigma \) of size \( q \). Ambient space, \( \Sigma^n \): Includes codewords and their corruptions.

• Hamming distance between strings \( x, y \in \Sigma^n \), denoted \( \Delta(x, y) \), is \# of coordinates \( i \) s.t. \( x_i \neq y_i \). (Converts ambient space into metric space.)

• Hamming weight of \( z \), denoted \( \text{wt}(z) \), is \# coordinate where \( z \) is non-zero.

Code: Subset \( C \subseteq \Sigma^n \).

Min. distance: Denoted \( \Delta(C) \), is \( \min_{x \neq y \in C} \{ \Delta(x, y) \} \).

\( e \) error detecting code If up to \( e \) errors happen, then codeword does not mutate into any other code.

\( t \) error-correcting code If up to \( t \) errors happen, then codeword is uniquely determined (as the unique word within distance \( t \) from the received word).

Proposition: \( C \) has min. dist. \( 2t + 1 \) \( \iff \) it is \( 2t \) error-detecting \( \iff \) it is \( t \) error-correcting.

Standard notation/terminology

• \( q \): Alphabet size

• \( n \): Block length

• \( k \): Message length, where \( |C| = q^k \).

• \( d \): Min. distance of code.

• Code with above is an \( (n, k, d)_q \) code. 
  \( [n, k, d]_q \) code if linear. Omit \( q \) if \( q = 2 \).

• \( k/n \): Rate

• \( d/n \): Relative distance.

Back to Hamming code

• So we have an \( [7, 4, 3] \) code (modulo proof of claim).

• Can correct 1 bit error.

• Storage efficiency (rate) approaches \( 4/7 \) (as word size approached \( \infty \)).

• Will do better, by looking at proof of claim.
Proof of Claim

Let \( H = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} \)

Sub-Claim 1: \( \{ x \in \mathcal{G} | x \} = \{ y | y \cdot H = 0 \} \).
Simple linear algebra (mod 2). You’ll prove this as part of Pset 1.

Sub-claim 2: Exist codewords \( z_1 \neq z_1 \) s.t. \( \Delta(z_1, z_2) \leq 2 \) iff exists \( y \) of weight at most 2 s.t. \( y \cdot H = 0 \).

Generalizing Hamming codes

Important feature: Parity check matrix should not have identical rows. But then can do this for every \( \ell \).

\( H_\ell = \begin{bmatrix} 0 & \cdots & 0 & 0 & 1 \\ 0 & \cdots & 0 & 1 & 0 \\ 0 & \cdots & 0 & 1 & 1 \\ \vdots & \ddots & \vdots & \vdots & \vdots \\ 1 & \cdots & 1 & 1 & 1 \end{bmatrix} \)

\( H_\ell \) has \( \ell \) columns, and \( 2^{\ell-1} \) rows.

\( H_\ell \) : Parity check matrix of \( \ell \)th Hamming code.

Message length of code = exercise. Implies rate \( \rightarrow 1 \).

Summary of Hamming’s paper (1950)

- Defined Hamming metric and codes.
- Gave codes with \( d = 1, 2, 3, 4 \! \)!
- \( d = 2 \) : Parity check code.
- \( d = 3 \) : We’ve seen.
- \( d = 4 \)?
- Gave a tightness result: His codes have maximum number of codewords. “Lower bound”.
- Gave decoding “procedure”.
Volume Bound

- **Hamming Ball:** \( B(x, r) = \{ w \in \{0, 1\}^n \mid \Delta(w, x) \leq r \} \).

- **Volume:** \( \text{Vol}(r, n) = |B(x, r)| \). (Notice volume independent of \( x \) and \( \Sigma \), given \( |\Sigma| = q \).)

- **Hamming (/Volume/Packing) Bound:**
  - Basic Idea: Balls of radius \( t \) around codewords of a \( t \)-error correcting code don’t intersect.
  - Quantitatively: \( 2^k \cdot \text{Vol}(t, n) \leq 2^n \).
  - For \( t = 1 \), get \( 2^k \cdot (n + 1) \leq 2^n \) or \( k \leq n - \log_2(n + 1) \).

Decoding the Hamming code

- Can recognize codewords? Yes - multiply by \( H_t \) and see if 0.

- What happens if we send codeword \( c \) and \( i \)th bit gets flipped?

- Received vector \( r = c + e_i \).

- \( r \cdot H = c \cdot H + e_i \cdot H \)
  \[ = 0 + h_i \]
  \[ = \text{binary representation of } i. \]

- \( r \cdot H \) gives binary rep’n of error coordinate!

Rest of the course

- More history!
- More codes (larger \( d \)).
- More lower bounds (will see other methods).
- More algorithms - decode less simple codes.
- More applications: Modern connections to theoretical CS.
Applications of error-correcting codes

• Obvious: Communication/Storage.

• Algorithms: Useful data structures.

• Complexity: Pseudorandomness ($\varepsilon$-biased spaces, $t$-wise independent spaces), Hardness amplification, PCPs.

• Cryptography: Secret sharing, Cryptoschemes.

• Central object in extremal combinatorics: relates to extractors, expanders, etc.

• Recreational Math.