Cơ sở lí thuyết thông tin

Chương 3: Mã hóa kênh
Mã Khối tuyến tính

TS. Phạm Hải Đăng

Phần 1: Khái niệm cơ bản

Mã kênh/Mã sửa lỗi

- Mã hóa kênh (channel Coding) hay còn gọi là mã sửa lỗi (Error Correction coding) là kỹ thuật khống chế, phát hiện và sửa lỗi trong quá trình truyền dữ liệu qua kênh có nhiễu.
- Mã sữa lỗi sử dụng thông tin dư thừa (redundancy) được mã hóa thêm vào dữ liệu phía bên phát. Thông tin dư thừa sẽ được phía thu sử dụng để sửa lỗi - mà không cần yêu cầu phát lại tin.

Phần 1: Khái niệm cơ bản

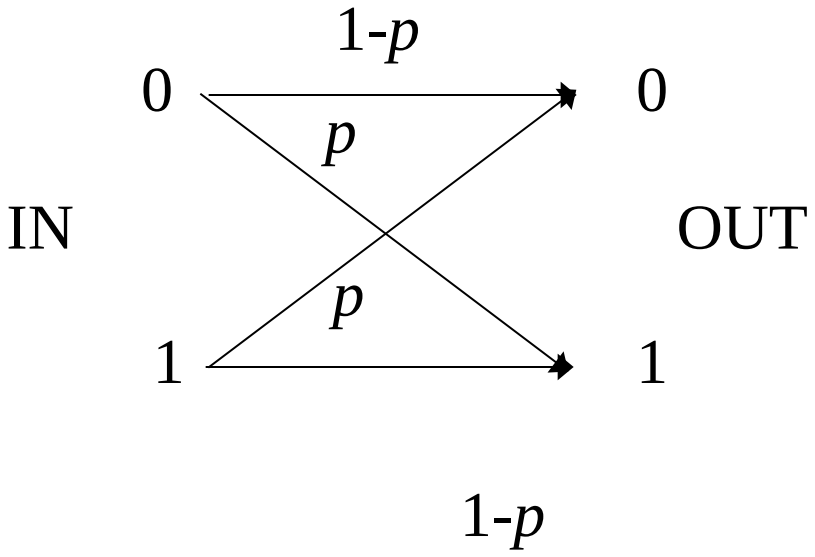
- Phân loại lỗi
  - Lỗi độc lập thống kê: Lỗi xuất hiện trong quá trình truyền tin trên kênh truyền, xuất hiện độc lập không liên quan tới nhau. Ví dụ: nhiễu Gaussian.
  - Lỗi chùm: Lỗi có phân bố liên hệ với nhau.

Phần 1: Khái niệm cơ bản

Ví dụ: Kênh truyền tin không nhớ

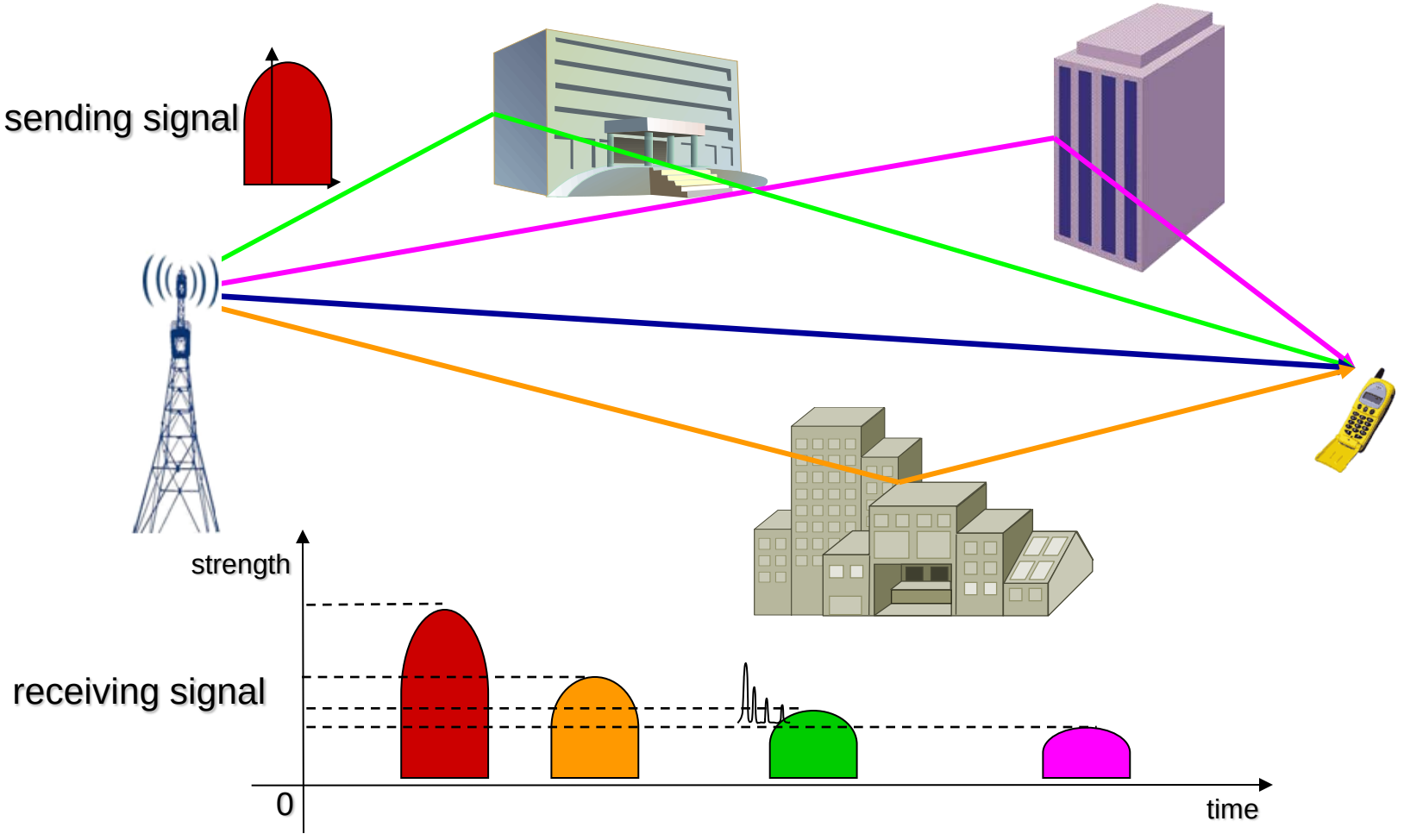
(Binary Symmetric Memoryless Channel).

- Lỗi xảy ra với bit "0" và "1" với cùng xác suất *p* (symmetric)
- Lỗi xảy ra ngẫu nhiên và độc lập giữa các bit (memoryless)

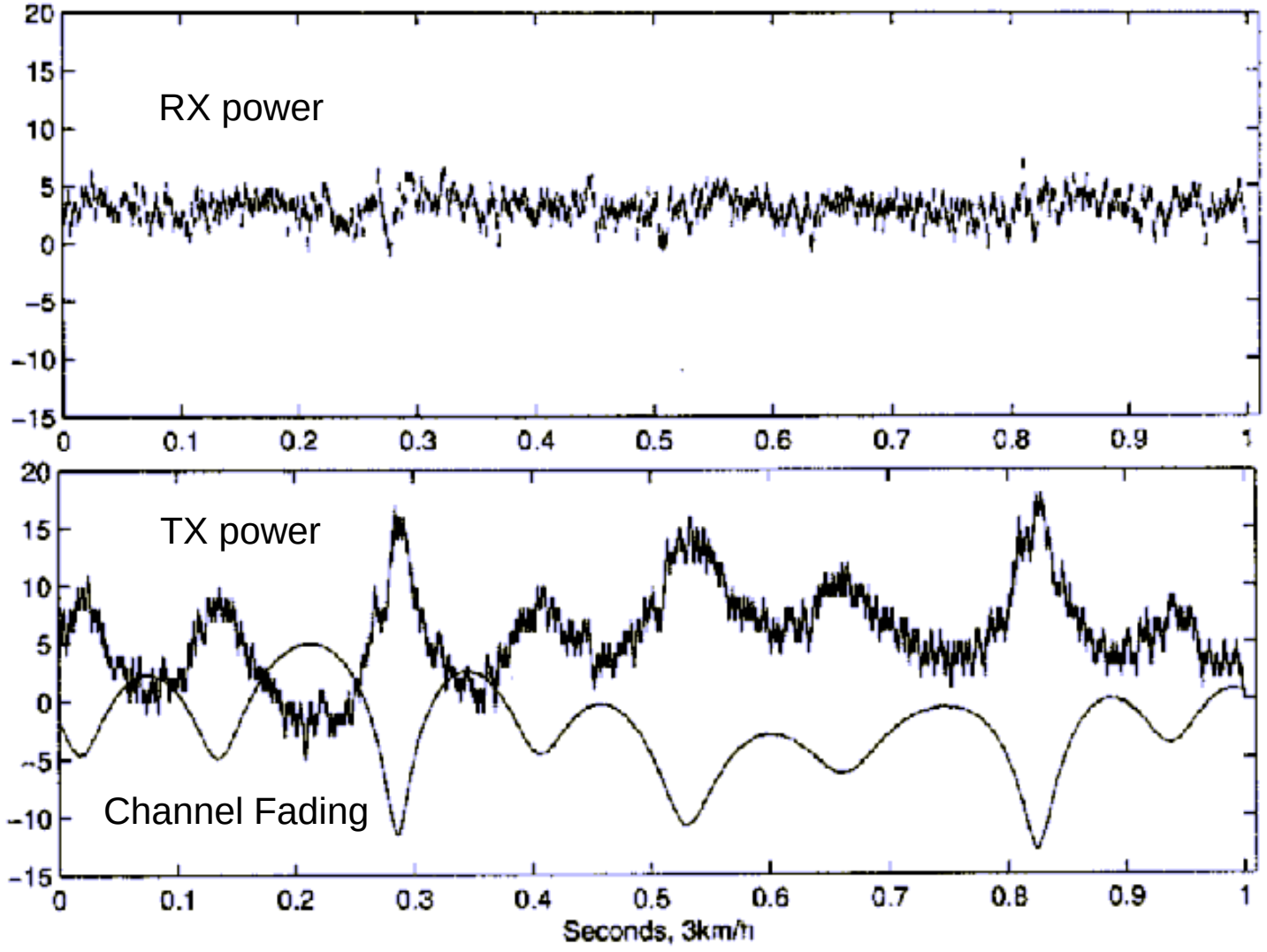


p là xác suất lỗi – BER
Bit Error Rate (BER)

Ví dụ: Kênh truyền tin đa đường



Ví dụ: Kênh truyền tin đa đường



Phần 1: Khái niệm cơ bản

Phân loại mã sửa lỗi:

- Mã khối (block codes): thông tin được mã hóa và chèn thêm phần dư thừa theo từng khối.
  - Mã khối
  - Mã khối tuyến tính
  - Mã vòng CRC
  - Mã BCH, Reed-Solomon, LDPC
- Mã chập (Convolutional codes): thông tin được biến đổi theo các hàm truyền đạt (phép tích chập). Không có giới hạn rõ ràng giữa thông tin và phần dư thừa.
  - Mã chập (covolutional codes)
  - Mã Turbo

Phần 2: Các khái niệm cơ bản của mã hóa sửa lỗi

- Tốc độ mã
- Khoảng cách Hamming (Hamming distance)
- Khoảng cách tối thiểu (minimum distance)
- Ma trận sinh, mã trận kiểm tra chẵn lẻ.

Tốc độ mã

Giả thiết $\mathbb{F}_2$ là tập hợp 2 phần tử '0' và '1'.

$\mathbb{F}_2^n$ biểu diễn vector *n* phần tử của $\mathbb{F}_2$

Số binary (n,k) là tập hợp 2^k điểm trong không gian $\mathbb{F}_2^n$

Mã (n,k) là mã chấp nhận k bit đầu vào và tạo ra n bit đầu ra.

Định nghĩa: tốc độ mã của mã (n,k) là

$$R = \frac{k}{n}$$

Ví dụ: Mã lặp (repetition code) $(n,1)$ nhận 1 bit đầu vào và tạo ra *n* bit lặp lại ở đầu ra. Tốc độ mã là $R = \frac{1}{n}$

Ma trận sinh

Với m biểu diễn thông tin (message).

C là từ mã (codeword) của mã lặp $(n,1)$

$$C = [m, m, m, \ldots m]$$

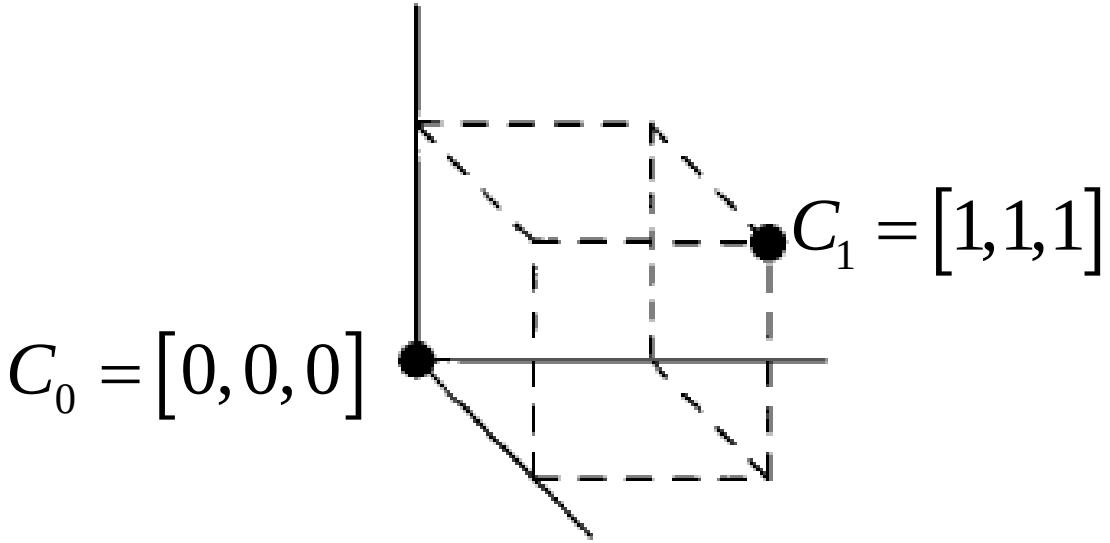
Quá trình mã hóa được biểu diễn dưới dạng ma trận. Ma trận sinh của mã lặp là G

$$C = mG$$

$$G = [1, 1, 1, \ldots, 1]$$

Phần 2: Các khái niệm cơ bản của mã hóa sửa lỗi

- Biểu diễn mã lặp (3,1) trong không gian



- Khoảng cách Hamming trong ma binary được tính bằng số các điểm khác biệt trong 2 từ mã.

$$C_1 = [1,0,1,1,1,0]$$
$$C_2 = [1,1,0,1,1,1]$$
$$d_{12} = 3$$

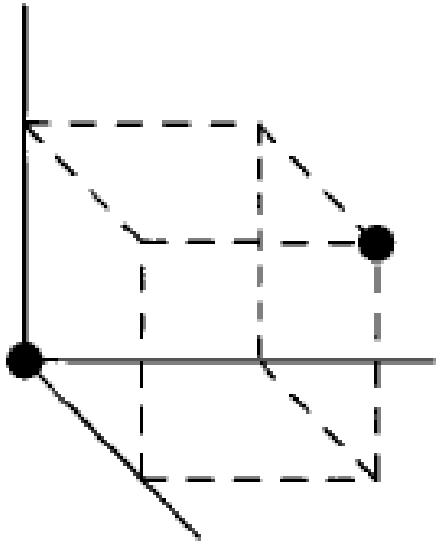
Phần 2: Các khái niệm cơ bản của mã hóa sửa lỗi

- Định nghĩa : Khoảng cách tối thiểu (min distance) là khoảng cách Hamming nhỏ nhất giữa 2 từ mã bất kì.

$$d_{\min} = \min_{\mathbf{c}_i, \mathbf{c}_j \in C, \mathbf{c}_i \neq \mathbf{c}_j} d_H(\mathbf{c}_i, \mathbf{c}_j).$$

- Phương pháp giải mã ML: tìm kiếm từ mã có khoảng cách gần nhất với từ mã thu được.

$$\hat{\mathbf{c}} = \arg \min_{\mathbf{c} \in C} d_H(\mathbf{r}, \mathbf{c})$$



Phần 2: Các khái niệm cơ bản của mã hóa sửa lỗi

- Liên hệ giữa khoảng cách Hamming tối thiểu và khả năng phát hiện và sửa lỗi.

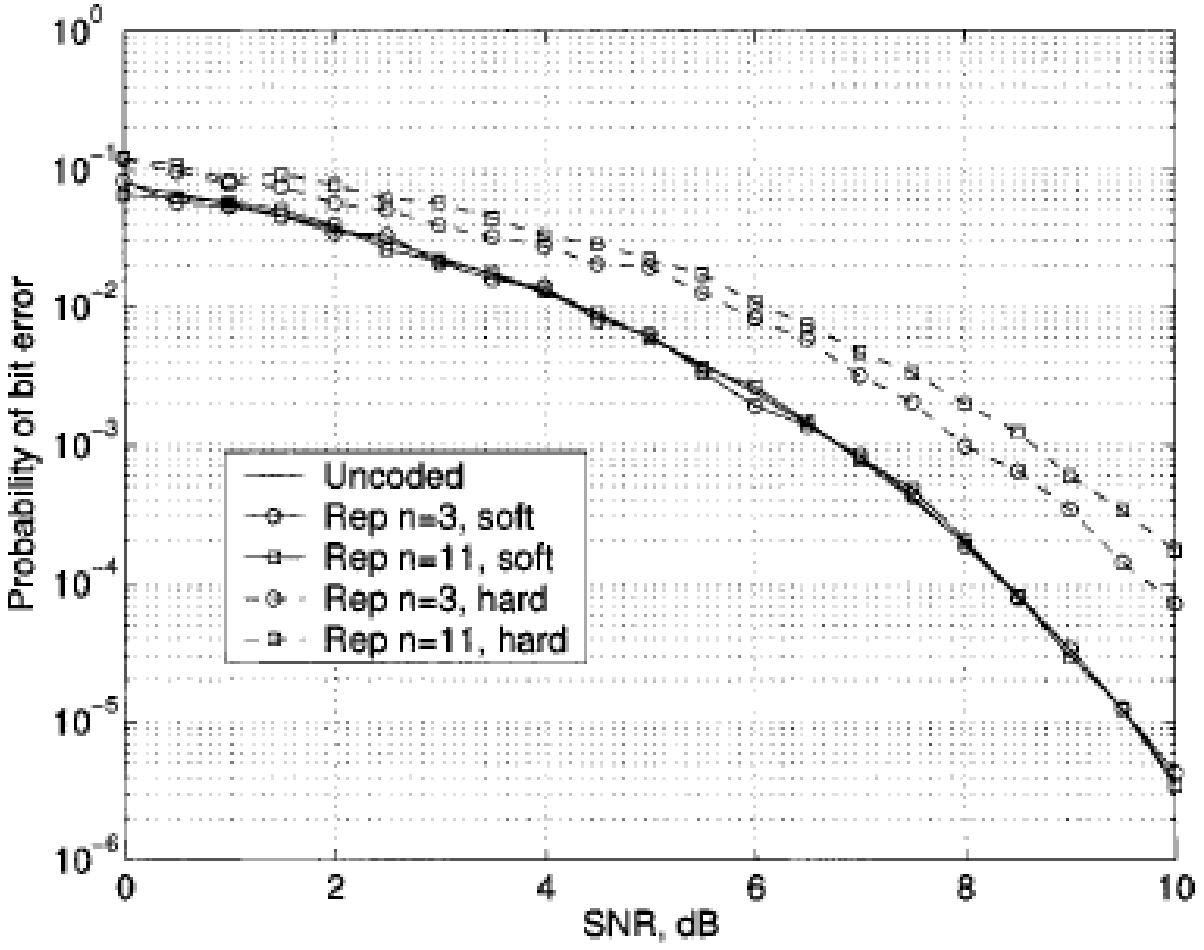
Với mã binary (n,k)

Khả năng phát hiện lỗi $d_{\min}-1$

Khả năng sửa lỗi $t=\left\lfloor \frac{d_{\min}-1}{2} \right\rfloor$

Phần 2: Các khái niệm cơ bản của mã hóa sửa lỗi

- Tỷ lệ lỗi bit (BER – Bit Error Rate) của mã lặp trong môi trường kênh AWGN (Additive White Gaussian Noise)



Phần 2: Các khái niệm cơ bản của mã hóa sửa lỗi

- Ví dụ mã kiểm tra chẵn
  - Trong trường hợp $n = k+1$, bản tin được bổ sung thêm 1 bit kiểm tra chẵn lẻ
  - Trong trường hợp số chẵn các bit '1', Bit kiểm tra chẵn lẽ có giá trị

$$q = \sum_{i=1}^{k} m_i \pmod 2$$

  - Trong trường hợp số lẻ các bit '1', bit kiểm tra có giá trị

$$1 - q$$

  - Bit kiểm tra chẵn lẽ được thêm vào đảm bảo số chẵn các bit '1' trong từ mã.
- Mã kiểm tra chẵn lẻ chỉ phát hiện được (tối đa) 1 lỗi, không có khả năng sửa lỗi.

Phần 2: Các khái niệm cơ bản của mã hóa sửa lỗi

- Ví dụ 1: Mã kiểm tra chẵn lẽ (6,5)
  - Bản tin m = (10110) => từ mã c=(101101)
  - Bản tin m = (11011) => từ mã c=(110110)

Phần 2: Các khái niệm cơ bản của mã hóa sửa lỗi

- Ví dụ 2: Bảng mã kiểm tra chẵn lẽ (4,3)

Dataword			Codeword			
0	0	0	0	0	0	0
0	0	1	0	0	1	1
0	1	0	0	1	0	1
0	1	1	0	1	1	0
1	0	0	1	0	0	1
1	0	1	1	0	1	0
1	1	0	1	1	0	0
1	1	1	1	1	1	1

Phần 3: Mã khối tuyến tính

- Mã khối (*n*,*k*) được biểu diễn dạng vector

  Bản tin d=(d_1 d_2....d_k)

  Từ mã c=(c_1 c_2.....c_n)

- Mã khối được xây dựng

  c=dG

  Với G là ma trận sinh

$$G = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ . & . & \dots & . \\ a_{k1} & a_{k2} & \dots & a_{kn} \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ . \\ a_k \end{bmatrix}$$

Phần 3: Mã khối tuyến tính

$$\mathrm{c} = \sum_{i=1}^{k} d_i \mathrm{a}_i$$

- Để đảm bảo 2 bản tin không có chung 1 từ mã (không thể giải mã/sửa lỗi), các hệ số a_i phải độc lập tuyến tính.
- Nếu c_i c_k là 2 từ mã bất kì

  $c = c_i + c_k$ cũng là 1 từ mã

- Hệ quả: khối gồm toàn bit '0' cũng là 1 từ mã

Khả năng sửa lỗi của mã khối tuyến tính

- Khoảng cách Hamming của mã khối tuyến tính là khoảng cách Hamming nhỏ nhất của các từ mã khác ‘0’
- Để tìm khoảng cách Hamming nhỏ nhất, cần tìm kiểm tra 2^k từ mã để tìm khoảng cách Hamming nhỏ nhất.

Ví dụ 1: mã khối tuyến tính

- Với mã (4,2), có ma trận sinh

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

$$a_1 = [1011]$$

$$a_2 = [0101]$$

- Với d=[1,1]

$$c = \begin{array}{cccc} 1 & 0 & 1 & 1 \\ + & 0 & 1 & 0 & 1 \\ \hline = & 1 & 1 & 1 & 0 \end{array}$$

Error Syndrome

- Để sửa lỗi mã khối tuyến tính, sử dụng phương pháp Error Syndrome
- Nếu $c_{r\ là}$ từ mã thu được ở phía thu, vector phát hiện lỗi (error syndrome) *s* của c_r

$$s = c_r H^T$$

- Nếu c_r bị lỗi, gọi e là vector lỗi

$$c_r = c + e$$

do đó

$$s = (c + e)\, H^T = cH^T + eH^T$$

$$s = 0 + eH^T$$

Vector syndrome có giá trị chỉ phụ thuộc vào vector lỗi e.

Error Syndrome

- Nhận xét: nếu cộng vector e với các từ mã khác thì vẫn thu được 1 vector syndrome.
  - Tổng cộng có $2^{(n-k)}$ syndromes, 2^n vector lỗi.
  - Ví dụ: với mã (3,2), có 2 syndromes và 8 vector lỗi e. Rõ ràng không thể sửa tất cả các lỗi trong trường hợp này.
  - Ví dụ: với mã (7,4) có 8 syndromes và 128 vector lỗi e.
  - Vì vậy, với 8 syndromes, ta cần bố trí các giá trị khác nhau để sửa được 7 lỗi (lỗi 1 bit) và 1 trường hợp không lỗi s=0.

Bảng liệt kê lỗi

- Bảng liệt kê lỗi được xây dựng như sau:

c_1 (all zero)	c_2	……	c_M	s_0
e_1	c_2+e_1	……	c_M+e_1	s_1
e_2	c_2+e_2	……	c_M+e_2	s_2
e_3	$\mathbf{c_2+e_3}$	……	c_M+e_3	s_3
…	……	……	……	…
e_N	c_2+e_N	……	c_M+e_N	s_N

← Các hàng đều có chung giá trị syndrome

Các hàng khác nhau có vector syndrome khác nhau

Bảng có 2^k cột (tương ứng với các từ mã hợp lệ) và 2^{n-k} hàng (số lượng các syndrome)

Phần 3: Mã khối tuyến tính – Giải mã sửa lỗi

- Bảng liệt kê lỗi được xây dựng theo cách
  - Liệt kê tất cả các lỗi 1 bit
  - Liệt kê tất cả các lỗi 2 bit
  - .....
- Kiểm tra đảm bác các lỗi được bổ sung vào bảng có vector syndrome khác nhau. Việc xây dựng bảng kết thúc khi sử dụng hết các vector syndrome.

- Để giải mã:
  - Tính vector syndrome theo công thức $s = 0 + eH^T$
  - Tra bảng tương ứng, tìm vector lỗi e tương ứng.
  - Cộng modulo vector e và từ mã thu được để giải mã

$$c = c_r + e$$

Phần 3: Mã khối tuyến tính – Mã hệ thống

- Mã hệ thống có phần thông tin và phần kiểm tra được phân tách trong từ mã, phần thông tin là không thay đổi so với ban đầu
- Ma trận sinh của mã hệ thống có dạng

$$G = \underbrace{\begin{bmatrix} 1 & 0 & .. & 0 \\ 0 & 1 & .. & 0 \\ .. & .. & .. & .. \\ 0 & 0 & .. & 1 \end{bmatrix}}_{k} \underbrace{\begin{bmatrix} p_{11} & p_{12} & .. & p_{1R} \\ p_{21} & p_{22} & .. & p_{2R} \\ .. & .. & .. & .. \\ p_{k1} & p_{k2} & .. & p_{kR} \end{bmatrix}}_{R} = [I \mid P]$$

$$R = n - k$$

Phần 3: Mã khối tuyến tính – Mã hệ thống

- Với mã hệ thống, ma trận kiểm tra được xây dựng

  $G = [\, I \,|\, P]$ and so $H = [-P^T \,|\, I]$

- Ví dụ: mã (7,4) với khoảng cách Hamming $d_{min} = 3$

$$G = [I \,|\, P] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \qquad H = [-P^T \,|\, I] = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

Tìm vector syndrome trong ví dụ

- Từ mã thu được c_r = [1101001]

$$s = c_r H^T = \begin{bmatrix}1 & 1 & 0 & 1 & 0 & 0 & 1\end{bmatrix} \begin{bmatrix}0 & 1 & 1\\ 1 & 0 & 1\\ 1 & 1 & 0\\ 1 & 1 & 1\\ 1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{bmatrix} = \begin{bmatrix}0 & 0 & 0\end{bmatrix}$$

Phần 4: Mã Hamming

- Mã Hamming là dạng đặc biệt của mã khối tuyến tính
- Với mỗi $r \geq 2$
  - Từ mã có độ dài $n = 2^r - 1$
  - Bản tin có độ dài $k = 2^r - r - 1$
  - Tốc độ mã $$R = \frac{k}{n} = 1 + \frac{r}{2^r - 1}$$
  - Khoảng cách Hamming $d_{\min} = 3$, khả năng sửa 1 lỗi
- Mã Hamming là mã có tốc độ mã *R* lớn nhất với cùng khoảng cách Hamming $d_{\min} = 3$

Phần 4: Mã Hamming (7,4)

- Mã Hamming (7,4) dạng hệ thống

$$G = [I \mid P] = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \qquad H = [-P^T \mid I] = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

- Mã Hamming (7,4) dạng không hệ thống

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

Phần 4: Mã Hamming (7,4)

- Ví dụ: mã Hamming (7,4) không hệ thống

  d = 1011

  c = 1110000

  + 0101010

  + 1101001

  = 0110011

  e = 0010000

  c_r = 0100011

  $s = c_r H^T = eH^T = 011$

- Chú ý: giá trị vector syndrome là vị trí lỗi.

Phần 4: Mã Hamming (7,4) – BER

- For a given channel bit error rate (BER), what is the BER after correction (assuming a memoryless channel, i.e., no burst errors)?
- To do this we will compute the probability of receiving 0, 1, 2, 3, …. errors
- And then compute their effect

Bit Error Rates after Decoding

- Example – A (7,4) Hamming code with a channel BER of 1%, i.e., $p = 0.01$

  P(0 errors received) = $(1 - p)^7 = 0.9321$

  P(1 error received) = $7p(1 - p)^6 = 0.0659$

$$\text{P(2 errors received)} = \frac{7 \times 6}{2} p^2 (1 - p)^5 = 0.002$$

P(3 or more errors) = 1 – P(0) – P(1) – P(2) = 0.000034

Bit Error Rates after Decoding

- Single errors are corrected, so,

  0.9321+ 0.0659 = 0.998 codewords are correctly detected

- Double errors cause 3 bit errors in a 7 bit codeword, i.e., (3/7)*4 bit errors per 4 bit dataword, that is 3/7 bit errors per bit.

  Therefore the double error contribution is 0.002*3/7 = 0.000856

Bit Error Rates after Decoding

- The contribution of triple or more errors will be less than 0.000034 (since the worst that can happen is that every databit becomes corrupted)
- So the BER after decoding is approximately 0.000856 + 0.000034 = 0.0009 = 0.09%
- This is an improvement over the channel BER by a factor of about 11

Perfect Codes

So,

$$2^R \geq 1+n \qquad \text{to correct up to 1 error}$$

$$\geq 1+n+\frac{n(n-1)}{2} \qquad \text{to correct up to 2 errors}$$

$$\geq 1+n+\frac{n(n-1)}{2}+\frac{n(n-1)(n-2)}{6} \qquad \text{to correct up to 3 errors}$$

If equality then code is Perfect

Only known perfect codes are SEC Hamming codes and TEC Golay (23,12) code ($d_{\min}=7$). Using previous equation yields

$$1+23+\frac{23(23-1)}{2}+\frac{23(23-1)(23-2)}{6}=2048=2^{11}=2^{(23-12)}$$