Chương 5: Mã hóa kênh

Chương 5: Mã hóa kênh

5.1. Mở đầu
5.2. Định lý Shannon 2
5.3. Luật giải mã
5.4. Giải mã theo đa số
5.5. Quãng cách Hamming
5.6. Giới hạn của độ dài từ mã
5.7. Xây dựng mã phát hiện sai/ sửa sai
5.8. Mã có tính chẵn
5.9. Mã Hamming
5.10. Mã vòng

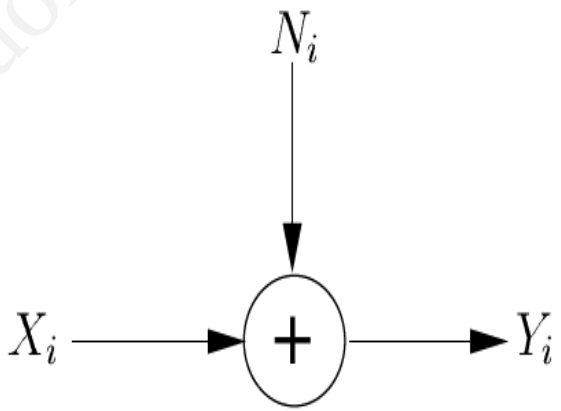
Nhắc lại

- Bài trước:
  - Mục đích của mã hóa nguồn?

  → Tìm phương pháp để biểu diễn bản tin với số ký hiệu mã sử dụng là tối thiểu (tối thiểu tài nguyên mã)
  - Mã hóa nguồn dùng cho kênh không nhiễu (Tốc độ lập tin cua nguồn < thông lượng của kênh)
- Nếu (Tốc đọ lập tin của nguồn > Thông lượng của kênh) thì mỗi đon vị thời gian sẽ có một lượng tin là R – C của nguồn tạo ra không thể chuyển được qua kênh. Khi truyền một phần lượng tin bị mất gây sai số hay nói khác kênh gây nhiễu thông tin được truyền.
  - → Cần một loại mã khac cho kênh có nhiễu
  - Mã này được gọi là mã kênh hay mã chống nhiễu

5.1. Mở đầu

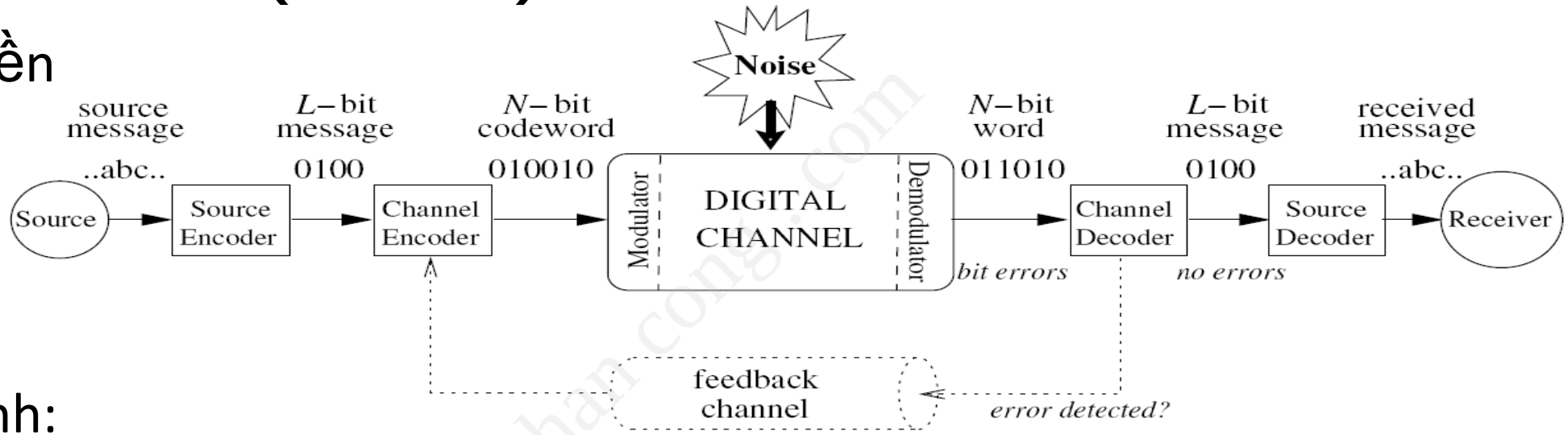
- Kênh chuyển tín hiệu (thông tin) vào thành tín hiệu (thông tin) ra và gây nhiễu tác động vào tín hiệu được truyền
  - Đầu ra = đầu vào + Nhiễu
  - Nhiễu tác động vào tín hiệu truyền qua kênh được coi là có phân bố Gaussian



$$Y_i = X_i + N_i$$

5.1. Mở đầu (Cont.)

- Hệ thống truyền



- Bộ mã hóa kênh:
  - Đầu vào của bộ mã hóa kênh là đầu ra của bộ mã nguồn
    - Có hai cách tổ chức đưa các ký hiệu vào:
      - Đưa trực tiếp đầu ra bộ mã nguồn vào đầu vào bộ mã kênh. Cách này được gọi là mã liên tục. Bộ mã hóa kênh liên tục nhận các ký hiệu mã vào và tạo các ký hiệu mã ở đầu ra.
      - Chia chuỗi mã ở đầu ra bộ mã nguồn thành từng chuỗi dài L ký hiệu mã gọi là tổ hợp mang tin và đưa từng tổ hợp mang tin dài L vào bộ mã hóa kênh. Theo cách này, mã được gọi là mã khối (mã từng khối L ký hiệu mã).
  - Đầu ra bộ mã hóa:
    - Với mã liên tục thì các ký hiệu mã được tạo ra liên tục nhau
    - Với mã khối thì một khối N ký hiệu mã được tao ra khi đầu vào là một tổ hợp mang tin dài L (N>L) .

5.1. Mở đầu (cont.)

- Bộ mã hóa kênh:
  - Thông thường mã khối được sử dụng để trình bày về lý thuyết mã hóa kênh
  - Nhiệm vụ của mã hóa kênh là đảm bảo truyền tin tin cậy trong trường hợp kênh có nhiễu (kênh gây ra sai thông tin truyền qua nó). Như vậy mã kênh phải đảm bảo phát hiện được sai và sửa được sai gây ra bởi kênh.
  - Khi tốc độ lập tin của nguồn lớn hơn thông lượng của kênh, để chống mất thông tin gây ra sai số, cần làm chậm tốc độ tạo tin của nguồn. Giải pháp của mã hóa kênh là thêm vào các ký hiệu mã không mang thông tin, gọi là các ký hiệu thừa hay ký hiệu kiểm tra.
    - Kênh vần truyền một lượng ký hiệu mã cố định trong một đơn vị thời gian, nhưng lượng tin trung bình chưa trong mỗi tin giảm đi do có các tin không chứa thông tin.

5.1. Mở đầu

- Coi tổ hợp mang tin L ký hiệu mã nguồn là tổ hợp mã m =(m1..mL) với r là cơ số mã. Số lượng tổ hợp mang tin sẽ là M = r lũy thừa L.
- Mỗi tổ hợp mang tin mi là chuỗi dài L ký hiệu mã nguồn, mỗi ký hiệu mã nguồn có chứa một lượng tin của nguồn bằng log(r) thường tính logr(r) =1 đơn vị thông tin tính theo cơ số r, hay đẳng xác suất. Các tổ hợp mang tin sẽ có xác suất bằng nhau.
- Bộ mã kênh chuyển mỗi tổ hợp mang tin thành dài L một từ mã chống nhiễu dài N, gọi là mã (N,L). Mã này có số từ mã bằng số tổ hợp mang tin và mọi từ mã có cùng xác suất xuất hiện.
- Số lượng ký hiệu thêm vào (ký hiệu thừa/ kiểm tra) RN = N – L.
- Tỷ số giữa số ký hiệu mã của tổ hợp mang tin chia cho số ký hiệu mã của từ mã được gọi là tốc độ mã hóa R. Với mã hóa kênh R = L/N

5.1. Mở đầu

- Mã kênh (mã chống nhiễu) sẽ sử dụng cùng cơ số mã r với mã nguồn. Số tổ hợp có thể của mã chống nhiễu sẽ là r lũy thừa N. Số từ mã là r lũy thừa L.
- Vì N>L nên mã chống nhiễu luôn có tổ hợp thừa, hay số tổ hợp thừa BN >0
- Tập các từ mã của mã chống nhiễu ký hiệu là A = {ai}, ai là một từ mã trong r lũy thừa L từ mã chống nhiễu dài N ký hiệu mã. từ mã ai sẽ được đưa vào đầu vào kênh.
- Tổ hợp dài N ký hiệu mã nhận được ở đầu ra kênh khi đưa từ mã ai vào kênh được ký hiệu là bj = ai + e. Tổ hợp mã e = (e1,..,eN) được gọi là tổ hợp gây sai đại diện cho nhiễu gây sai từ mã ai thành tổ hợp bj. mỗi ẹk là một ký hiệu mã, ek = không thì vị trí k không bị sai, ek= 1/../(r-1) thì ký hiệu thứ k của bj là bkj = aki + ek. Phép cộng theo mô đun cơ số r.
- Tập các tổ hợp nhận được bj (do từ mã ai sinh ra) là từ mã sẽ được ký hiệu BM
- Tập các tổ hợp nhận được bj (do từ mã ai sinh ra) không phải là từ mã được ký hiệu B’M

5.2. Định lý mã hóa của Shannon cho kênh có nhiễu

- Cho một kênh rời rạc có thông lượng C và nguồn vào của nó cũng rời rạc, có tốc độ lập tin R
  - Nếu R ≤ C thì sẽ tồn tại ít nhất một mã để truyền nguồn trên kênh với sai số bé tùy ý

→ Định lý này cho phép thực hiện truyền thông tin cậy qua kênh có nhiễu.

Tại sao?

5.3. Luật giải mã

- Giả sử ai là từ mã dài N ký hiệu mã được truyền vào kênh và đầu ra kênh sẽ nhận được tổ hợp dài N ký hiệu mã b. Tổ hợp b có thể là từ mã hoặc không.
- Bộ giải mã sẽ sử dụng luật giải mã D(.) để quyết định có phải ai đã được truyền khi nó nhận được b không. Ký hiệu ai = D(b).
- Giả sử p(b/ai) là xác suất nhận được b khi đầu vào kênh có ai được truyền vào.
- Với kênh không nhớ: p(b/ai) = p(b1/a1i)..p(b2/a2i). ở đây bj là ký hiệu thứ j của tổ hợp nhận được b, ạji là ký hiệu thứ j của từ mã ai.

5.3. Luật giải mã.

- Theo công thức Bayes:
  - p(ai/b) = p(b/ai)p(ai)/p(b)
- Nếu bộ giải mã giải mã ra ai khi nhận được b thì sẽ là giải mã đúng. Xác suất giải mã đúng sẽ là p(ai/b) tính ở trên. Nếu bộ giải mã giải mã ra từ mã khác ai sẽ là xác suất giải mã sai. Xác suất giải mã sai là 1- p(ai/b). Tối thiểu hóa xác suất giải mã sai sẽ tối thiểu hóa được giải mã sai. Để tối thiểu hóa xác suất giải mã sai cần phải cực đại hóa xác suất giải mã đúng.
- Luật giải mã sẽ là: khi nhận được b, từ mã ai sẽ được chọn là từ mã được truyền vào kênh sao cho cực đại hóa được xác suất p(ai/b). Để tối thiểu hóa sai giải mã, cần cực đại hóa xác suất giải mã đúng p(ai/b).
- Luật giải mã cực tiểu hóa sai số, theo công thức bayes) sẽ là:
- Chọn từ mã ai được truyền khi nhận được tổ hợp b, nếu xác suất p(b/ai)p(ai)/p(b) đạt cực đại

5.3. Luật giải mã

- Luật giải mã cực tiểu hóa sai số thường được trình bày ở dạng:
- Chọn từ mã ai được truyền khi nhận được tổ hợp mã b, nếu:
- p(b/ai)p(ai)/p(b) >= p(b/aj)p(aj)/p(b) với mọi aj khác ai
- p(b/ai), p(b/aj) là các xác suất truyền của kênh; p(ai), p(aj) là các xác suất của các từ mã đưa vào kênh (nguồn vào). p(b) là xác suất tổ hợp nhận được
- Luật giải mã cực tiểu hóa sai số chuyển về dạng sau do p(b) chung cả 2 vế:
- Chọn từ mã ai được truyền khi nhận được tổ hợp mã b, nếu :
- p(b/ai)(p(ai) >= p(b/aj)p(aj) với mọi ạj khác ai
- → Luật giải mã theo cực đâị hóa tương đồng giưa ai và b (Maximum Likelihood)
- Thường p(ai) = p(aj), luật giải mã chuyển thành:
- Chọn từ mã ai được truyền khi nhận được tổ hợp b, nếu:
- p(b/ai) >= p(b/aj) với mọi aj khác ai.
- → Luật giải mã theo cực đại hóa xác suất hậu nghiệm (Maximun Apriori Probability)

8.3. Luật giải mã (Cor

$$\mathbf{P} = \begin{bmatrix} 0.6 & 0.4 \\ 0.4 & 0.6 \end{bmatrix}$$

Code word	$P_N(\mathbf{a}_i)$
$\mathbf{a}_1 = (000)$	0.4
$\mathbf{a}_2 = (011)$	0.2
$\mathbf{a}_3 = (101)$	0.1
$\mathbf{a}_4 = (110)$	0.3

- Ví dụ:
  - Một kênh BSC có ma trận kênh P, L=2, N=3. Các từ mã và xác suất xuất hiện của chúng cho bởi bảng trên. Tổ hợp nhận được là b=111
  - Tính:

$$P_N(\mathbf{b}|\mathbf{a}_1) = P_N(111|000) = P(1|0) \times P(1|0) \times P(1|0) = 0.064$$
$$P_N(\mathbf{b}|\mathbf{a}_2) = P_N(111|011) = P(1|0) \times P(1|1) \times P(1|1) = 0.144$$
$$P_N(\mathbf{b}|\mathbf{a}_3) = P_N(111|101) = P(1|1) \times P(1|0) \times P(1|1) = 0.144$$
$$P_N(\mathbf{b}|\mathbf{a}_4) = P_N(111|110) = P(1|1) \times P(1|1) \times P(1|0) = 0.144$$

♆✸♎ ♍⇔ ♦♒☞ ♍♒♐■ ♋🗎🖃♋🗎🖃♋🗐

- Luật giải mã cực tiểu hóa sai số

$$P_N(\mathbf{b}|\mathbf{a}_1)P_N(\mathbf{a}_1) = 0.064 \times 0.4 = 0.0256$$
$$P_N(\mathbf{b}|\mathbf{a}_2)P_N(\mathbf{a}_2) = 0.144 \times 0.2 = 0.0288$$
$$P_N(\mathbf{b}|\mathbf{a}_3)P_N(\mathbf{a}_3) = 0.144 \times 0.1 = 0.0144$$
$$P_N(\mathbf{b}|\mathbf{a}_4)P_N(\mathbf{a}_4) = 0.144 \times 0.3 = 0.0432$$

Chọn a4

5.4. Giải mã theo đa số

- Là phương pháp giải mã khi truyền lặp
  - Luật: ký hiệu nào xuất hiện nhiều nhất trong chuỗi ký hiệu nhận được từ chuỗi ký hiệu truyền lặp cho 1 ký hiệu sẽ là ký hiệu được truyền.
- Mã lặp được thực hiện ở dạng mỗi ký hiệu mã đưa vào sẽ được lặp lại chính nó một số lần.
  - Ký hiệu (n,m) ở đây n là số lần lặp cho một ký hiệu mã, m là số ký hiệu mã của bản tin.
- Nếu một mã lặp nhị phân (n,1) được dùng, thì mỗi bít vào sẽ được chuyển thành một chuỗi n bit trùng với nó. Thường n = 2t +1, t là số nguyên tùy chọn.
- Mã lặp có thể phát hiên (n-1)/2 lỗi.

5.4. Giải mã theo đa số

- Thuật toán giải mã cho mã nhị phân (n,1):
- Vì n = 2t + 1 và giả thiết sai không vượt quá t vị trí, thì:
  - Nếu tổng vị trí của tổ hợp nhận được có giá trị bằng 1, dH < t (số 0 nhiều hơn) thì chuỗi (từ mã) được truyền là toàn 0, ký hiệu được truyền là 0.
  - Nếu tổng dH>t (số 1 nhiều hơn) thì từ mã được truyền là toàn 1, ký hiệu 1 được truyền.
- Ví dụ, mã nhị phân (5,1) và tổ hợp nhận được là b = 10110.
- Mã này có t = 2. Tổ hợp nhận được có dH =3. Vậy từ mã được truyền là 11111 và ký hiệu được truyền là 1.

5.5. Quãng cách Hamming

- Giả sử có hai từ mã dài N lký hiệu mã à a = a1..aN và b = b1..bN
- Quãng cách Hamming giữa a và b, ký hiệu là d(a,b), được định nghĩa là số vị trí có ký hiệu mã khác nhau giữa hai từ mã.
- Quãng cách Hamming là độ đo được định nghĩa trên tất cả các cặp tổ hợp mã cùng độ dài.
- Quãng cách Hamming thỏa mãn các luật sau:
  - d(a,b) >= 0
  - d(a,b) = d(b,a)
  - d(a,b) + d(b,c) >= d(a,c) (bất đẳng thức tam giác)

5.5. Quãng cách Hamming (cont.)

- Ví dụ: cho N = 8

$$\mathbf{a} = 11010001$$
$$\mathbf{b} = 00010010$$
$$\mathbf{c} = 01010011$$

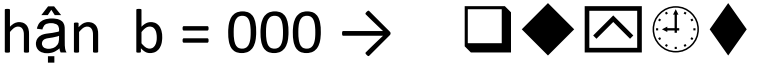
  - d(a,b) = 4, d(b,c) = 2, d(a,c) = 2
  - d(a,b) + d(b,c) = 4 + 2 ≥ d(a,c) = 2

5.5.1. Luật giải mã theo quãng cách Hamming

- Số sai của kênh, ký hiệu là t, được định nghĩa là số vị trí sai lớn nhất kênh có thể gây ra cho một từ mã được truyền qua kênh.
- Giả sử b là tổ hợp mã dài N nhận được khi truyền từ mã ai dài N qua kênh. Quãng cách Hamming giữa ai và b là d(ai,b) <= t. Quãng cách d(ai,b) = 0 khi ai = b hay kênh truyền không gây sai.
- Luật giải mã theo quãng cách Hamming là khi nhận được tổ hợp mã b và từ mã được truyền ai là (dựa theo luật giải mã cực đại hóa sự tương đồng):
- - Nếu b = ai (d(ai,b) = 0) thì giải mã ai =b
- - Nếu ai khác b thì với mọi aj khác ai sẽ chọn aj là từ mã được truyền, nếu
- d(ai,b) <= d(aj,b), sai giải mã hay chấp nhận đường truyền gây ra số vị trí sai t = d(ai,b)

5.5.1. Luật giải mã

Message ($L = 2$)	Code word ($N = 3$)
00	000
01	001
10	011
11	111

- Ví dụ:
  - Nếu nhận b = 000 →  a = 000
  - Nếu b ≠ 000 với sai quyết định t=1:

$\mathbf{b} = b_1 b_2 b_3$	Closest code word	Action
010	000 (b_2 in error), 011 (b_3 in error)	1-bit error detected
100	000 (b_1 in error)	1-bit error corrected
101	001 (b_1 in error), 111 (b_2 in error)	1-bit error detected
110	111 (b_3 in error)	1-bit error corrected

5.5.2. Quãng cách mã

- Quãng cách mã, ký hiệu d(Kn): Quãng cách Hamming cực tiểu giữa hai từ mã bất kỳ của bộ mã có từ mã dài N ký hiệu mã
  - d(Kn) = min (d(a,b)); Kn là bộ mã có các từ mã dài N ký hiệu mã
  - Ví dụ: Kn:

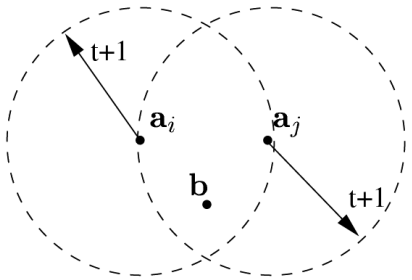
11010001
00010010
01010011

→ d(Kn) = 2

5.5.3. Phát hiện sai và sửa sai dùng quãng cách Hamming

- Phát hiện từ mã bị sai:
  - Mã khối, Kn, sẽ phát hiện được đến t sai khi và chỉ khi quãng cách mã thỏa mãn

    d(Kn) > t (5.1)
    - Công thức 5.1 là giới hạn về quãng cách mã của mã phát hiện được t sai.
    - Mã sẽ cho phép phát hiện đến t sai khi d(Kn) >= t+1.
  - Đồ hình minh họa phát hiện đến t sai khi d(Kn) = t+1:
    - ai, aj là hai từ mã dài N. Mỗi vòng tròn biểu thị không gian của các tổ hợp sai của mỗi từ mã khi bị sai ≤ t vị trí

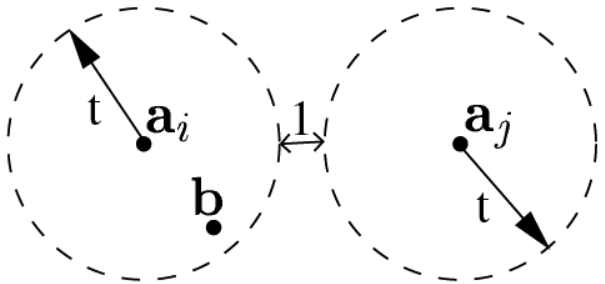


5.5.3. Phát hiện sai và sửa sai dùng quãng cách Hamming

- Mã khối, Kn, sửa được đến t sai khi và chỉ khi quãng cách mã thỏa mãn:

  $$d(Kn) > 2t \quad (5.2)$$

  - Công thức 5.2 là giới hạn về quãng cách mã để mã sửa được đến t sai.
  - Mã sẽ cho phép sửa đến t sai nêu d(Kn) >= 2t + 1
- Đồ hình minh họa mã sửa được đến t sai khi d(Kn) = 2t+1:
  - ai, aj là hai từ mã dài N, mỗi vòng tròn biểu diễn không gian các tổ hợp sai của mỗi từ mã khi bị sai ≤ t vị trí



5.5.3.

- Ví dụ:

Message	Code word
00	000
01	011
10	101
11	110

d(KN) = 2 → ♍♒🡕 ◘♒🡑♦ ♒♓🡔■ ⇨⌗🡖♍ 🗁 ♦♋♓ (t=1) vì yêu cầu d(KN) > t, và không sửa được sai vì yêu cầu d(kn) >= 2t +1

5.5.3.

- Ví dụ

Message	Code word
0	000
1	111

- d(KN) = 3 ➐ Phát hiện được đến 2 sai, sửa được 1 sai

5.6. Giới hạn về độ dài từ mã

- Bound of length of N-symbol codeword:
  - When transmitting a N-symbol codeword through channel with t-error, the number of errors would be:

$$N_{1E} = \sum_{i=1}^{t} C_N^i (r-1)^i$$

    - The channel with t-errors, it means the codeword may has from 1 to t error-positions
    - When the codeword has i error-positions, number of received error combinations will be C_N^i
      - $C_N^i = \frac{N!}{(N-i!)i!}$
    - Each error-position has (r-1) ways of errors
  - To detect the error, it needs to have enough number of "don't care" combination B_N

$$B_N \geq N_{1E} \rightarrow r^N - r^L \geq \sum_{i=1}^{t} C_N^i (r-1)^i \quad (8.3)$$

    - (8.3) is bound of length of N-symbol codeword of detection code
    - If r=2: → $r^N - r^L \geq \sum_{i=1}^{t} C_N^i$
    - If r=2, t=1 → $r^N - r^L \geq C_N^1$ → N ≥ L +1 → only need to add one symbol to the binary message to detect 1-error

5.6. Giới hạn về độ dài từ mã

- Bound of length of N-symbol codeword:
  - With correction code, received combination must be separated →error combinations are separated → number of the error combinations:

    $N_E = r^L \times N_{1E}$

    Where r^L is number of codewords
  - To correct the error, it needs to have enough number of "don't care" combination B_N

$$B_N \geq N_E \rightarrow r^N - r^L \geq r^L \sum_{i=1}^{t} C_N^i (r-1)^i$$

$$\rightarrow r^{N-L} - 1 \geq \sum_{i=1}^{t} C_N^i (r-1)^i$$

$$r^{N-L} \geq \sum_{i=0}^{t} C_N^i (r-1)^i$$

logarithm with base r:

$$\text{N-L} \geq log_r\left(\sum_{i=0}^{t} C_N^i (r-1)^i\right) \qquad (8.4)$$

- (8.4) is bound of length of N-symbol codeword of correction code
- If r = 2 → $\text{N-L} \geq log_r\left(\sum_{i=0}^{t} C_N^i\right)$
- If r =2, t =1 → $\text{N-L} \geq log_2(C_N^0 + C_N^1) = log_2$ (1 +N)
  - E.g: L=4 then N ≥ 7

5.7. Xây dựng mã phát hiện sai/ sửa sai

- Detection code construction:
  - Given L,t,r
  - Step 1: use (8.3) to calculate the length of codeword. Choose Nmin
  - Step 2: Choose N-symbol combination of 0 as first codeword. Continue find $(r^N - 1)$ N-symbol combinations as codewords so that minimum distance of code d satisfies (8.1)

- Correction code construction:
  - Given L,t,r
  - Step 1: use (8.4) to calculate the length of codeword. Choose Nmin
  - Step 2: Choose N-symbol combination of 0 as first codeword. Continue find $(r^N - 1)$ N-symbol combinations as codewords so that minimum distance of code d satisfies (8.2)

5.8. Mã Parity

- Binary code may detect 1-error
- Apply (8.3), the length of parity codeword N is length of message L plus 1
- To assure that d(Kn) ≥ 2, the added symbol must be:
  - If message has an even number of positions whose value is 1, added symbol =0
  - If message has an odd number of positions whose value is 1, added symbol =1

  → All codewords has even number of positions whose value is 1 (even codeword)
- To verify a binary combination is even or not,

$$\mathrm{P} = XOR_{j=1}^{L} m_{ij} \text{ where } m_{ij} \text{ is } j^{th} \text{ symbol in message } m_i$$

  - If P = 0 : even, P=1 : odd

5.8. Mã Parity

- Encoding algorithm:
  - Calculate P of message
  - Codeword is message m_i plus P where P called parity bit (PB)
- Decoding algorithm:
  - Calculate the syndrome S (sign to detect error, S ≤ 0: no error, S > 0 : error)
    - S = $XOR_{j=1}^{L} b_j$ where b_j is j^{th} symbol of received word b
      - S = 0: No error
      - S = 1: Error

5.8. Mã Parity

- Ví dụ:
  - Tập bản tin (tổ hợp có thể): {00,01,10,11}. L = 2
    - 00, 11: tổ hợp chẵn → P=0
    - 10,01: tổ hợp lẻ → P =1
  - Bộ mã sẽ là:

    000,110,101,011
  - Nếu nhận tổ hợp 010, thì s= 1, → ♦♋♓

8.9. Hamming code

- Linear binary block code proposed by R. Hamming
- Can correct 1-error
- Have largest length:
  - According to (8.4) N-L ≥
    - r = 2, t = 1 → N-L ≥ (1 +N) → ≥ 1 + N → N - 1
      - Nmax = - 1
- Hamming code uses linear space to represent code
  - Code that uses linear space called linear code

8.9. Hamming code (cont.)

- Linear space
  - A vector space over a field F is a set V together with two operations that satisfy the eight axioms listed below.
    - The first operation, called vector addition or simply addition +
      - $u, v \in V \rightarrow w = u + v \in V$
    - The second operation, called scalar multiplication .
      - $u \in F, v \in V \rightarrow w = u \cdot v \in V$

8.9. Hamming code (cont.)

- Linear space
  - Axioms:
    - Associativity of addition $u + (v + w) = (u + v) + w$
    - Commutativity of addition $u + v = v + u$
    - Identity element of addition There exists an element $0 \in V$, called the zero vector, such that $v + 0 = v$ for all $v \in V$.
    - Inverse elements of addition For every $v \in V$, there exists an element $-v \in V$, called the additive inverse of v, such that $v + (-v) = 0$.
    - Compatibility of scalar multiplication with field multiplication $a(bv) = (ab)v$
    - Identity element of scalar multiplication $1v = v$, where 1 denotes the multiplicative identity in F.
    - Distributivity of scalar multiplication with respect to vector addition $a(u + v) = au + av$
    - Distributivity of scalar multiplication with respect to field addition $(a + b)v = av + bv$

8.9. Hamming code (cont.)

- Linear space
  - If the element of V is N-dimension vector then V is called N-dimension vector space
    - a ∈ V then a = a1,a2,…,aN
    - ai has discrete values from 0 to r-1 → discrete space with base r
  - Generic matrix
    - Set of N independent elements of V called set of base elements
      - Base elements are denoted by g1,g2,..gN
    - Set of base elements can generate all elements of V
    - Arrange each N-dimension element in one row → N x N matrix whose rows are independent.
      - This matrix is called generic matrix (G)
    - a ∈ V if and only if a = C .G → a = → a = a1a2..aN
      - C is coefficient vector
      - In discrete space with base r: value of ci is 0/1/…/r-1
      - C has values
      - a = C .G can generate all N-dimension elements of space
    - If G is unit matrix
      - G is in canonical form
      - a is called systematic code
        - k first symbols are carrying information symbols, remaining symbols are checked symbols

8.9. Hamming code (cont.)

- Linear space
  - L-dimension subspace (L<N) is a subspace of N-dimension space.
    - Each element of L are N-dimension elements
    - Has maximum L independent elements
      - Can be considered as set of base elements of subspace
    - Generic matrix has L rows, N columns (
    - One element a ∈ if and only if a = C . while C=c1c2…cL
    - Number elements of subspace is
    - is in canonical form when its first (L x L) submatrix if unit matrix
  - N-L dimension subspace:
    - its elements are orthogonal with N-dimension subspace
    - Called orthogonal space
    - Generic matrix has (N-L) row, N columns ()
      - . = 0
      - a ∈ if and only if a. = 0
      - is called “check parity matrix”
    - is in canonical form when its first ((N-L) x (N-L)) submatrix if unit matrix

8.9. Hamming code (cont.)

- Linear code:
  - One codeword of linear code is mapped to one element of L-dimension subspace
  - Other elements of N-dimension space which don't belong to L-dimension subspace is "don't care combination"
  - With linear code: if *a* is codeword then *a* is generated by $a = C.G$

    or *a* satisfies $a. = 0$
  - To simplify is denoted by G, is denoted by H
  - To decode: when receive b, calculate syndrome S = b.
    - S = 0: no error
    - S > 0 : error
    - Since b = a + e where e = { ,…, } is "error combination", S = e.

      → e can be calculated using S

8.9. Hamming code (cont.)

- Hamming code:
  - To build Hamming code or to decode a codeword of Hamming code, Hamming uses only “check parity matrix” H
  - Hamming proposes: each column of check parity matrix is a (N-L) binary number
    - The value of binary number = order number of column
  - Hamming code is binary code that can correct 1-error
  - Length of Hamming code N = - 1
  - To build: Solve a=0 to determine codeword a
    - If a= is codeword needed to be built then a=0
    - a=0 is matrix equation which generates system of (N-L) first-order equations
      - ai = 0 when hi is the i-th row of matrix H
      - Systems of equations can only determine (N-L) ai, other L symbol ai of a will be given parameters
        - Given parameters are L-symbol message
        - ai are given parameters
          - Its position corresponds with column order of matrix H
            - The column has only value 1

8.9. Hamming code (cont.)

- Hamming code:
  - To decode:
    - Let b is received combination, need to calculate syndrome S =
    - If S = 0 → no error
    - If S ≠ 0 → S = e. = where is ith column of matrix H with the wrong position is i
      - binary number that has value = i → Syndrome indicates wrong position

8.9. Hamming code (cont.)

- Example
  - L = 4, t = 1, r =2
  - Let message m = {m1,m2,m3,m4}
  - N is calculated by N = 2^ (L – 1) - 1 → N = 7
  - Check matrix (check parity matrix):
    - $H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$
  - Position 1,2,4 of matrix H has only one position that has value = 1

→a = (x,y,m1,z,m2,m3,m4)

→then a = {z+m2+m3+m4, y+m1+m3+m4,x+m1+m2+m4} = {0,0,0}

8.9. Hamming code (cont.)

- x = m1+ m2+ m4
- y = m1+m3+m4
- z= m1+m2+m3

→ a = {m1+m2+m4, m1+m3+m4, m1,m1+m2+m3,m2,m3,m4}

This gives us the code

$$\{0000000, 0001111, 0010110, 0011001, 0101010, 0101101, 0110011, 0111100,$$
$$1001011, 1001100, 1010101, 1011010, 1100110, 1101001, 1110000, 1111111\}.$$

To illustrate the error-correction procedure, suppose the third bit of the code word 1100110 gets corrupted, giving 1110110. The syndrome is $1110110H^T = 011$, indicating the third bit, as expected.

Similarly, the syndrome of 1110010 is $1110010H^T = 110$, indicating that the sixth bit has been corrupted, and that the correct code word is 1110000.

- Input: L-symbol message
- Output: N-symbol codeword

8.10.Cyclic code

- 8.8.1
- 8.8.2

8.8.1 Galois field

- Field: field is a set of elements and operations of addition and multiplication. The operations must follows rules below
  - Closed: Closure impliles that the sum and product of any two elements in the field are also elements of the field
  - Commutative (ab = ba and a+b = b+a)
  - Associative (a(bc) = (ab)c, and a + (b + c) = (a + b) + c)
  - Distributive law relates multiplication and addition: a(b + c) = ab + ac.
  - Has additive and multiplicative identities (0 and 1) such that a + 0 = a and 1a = a for any element in the field.
  - Elements of a field must have additive and multiplicative inverses. The additive inverse of a is an element b such that a+b = 0 and the multiplicative inverse of a is an element c such that ac = 1.
  - E.g:
    - set of real numbers and addition, multiplication creates field.

8.8.1 Galois field

- Finite field:
  - Denoted by Zp that contains
    - The set of integers {0, 1, .., p−1}
    - Modulo p arithmetic.
    - p is a prime number
- Galois field: GF () contains
  - p is prime number
  - n is arbitrary positive integer
  - Each element is denoted by polynomial *a1+ a2+….+ aN* where the coefficients ai take on values in the set {0, 1, ..., p − 1}.
  - To add two polynomials, for each power of x present in the summands, just add the corresponding coefficients modulo p
  - a(x)= *a1+ a2+….+ aN*
  - c(x) = a(x)+b(x) = (a1b1) + (a2b2)x + …+ (*aN bN)*
  - ai = ai+bi if ai+bi <p
    = ai +bi –p if ai+bi >=p
  - Multiplication of two polynomials is done by multilication in modulo where is modulo polynomial a(x) x b(x) modulo ( ) = reminder of ((a(x) x b(x))/)

8.8.2 Definition

- *Cyclic code uses Galois Field GF()*
- *Codeword a is considered as polynomials*
  - *E.g. a = {,,…,} is considered as a(x)= + +….+*
- *Multiplication is calculated in modulo 1*
- *Multiple with x is equivalence to right shift its coefficients*

  $xa(x) = a0 +$ a1x^1 $+ a2x^2 +….+a(N-1) +aN(x^n -1)$

  $xa(x)\ modulo(x^n -1) = aN +$ a0x^1$+a1x^2 +….+$ a(N-1)
- Cyclic code *is a linear code with the property that any cyclic shift of a code word is also a code word*
- *A cyclic code has a unique non-zero polynomial of minimal degree*
  - *This polynomial is called generator polynomial with degree r:*
    g(x) = g0 + g1x^1 +…+ grx^r
  - *g(x) is the generator polynomial of a cyclic code if and only if it is a factor of* (X^N-*1)*
  - *The remainder of division between arbitrary codeword and g(x) =0*
    - *If c(x) is codeword then c(x) = m(x) g(x)*

8.8.2. Definition(Cont.)

- Generator matrix:

$$G = \begin{bmatrix} g_0 & g_1 & \dots & g_r & 0 & 0 & \dots & 0 \\ 0 & g_0 & \dots & g_{r-1} & g_r & 0 & \dots & 0 \\ & & & \vdots & & & & \\ 0 & \dots & 0 & g_0 & g_1 & \dots & g_r & 0 \\ 0 & \dots & 0 & 0 & g_0 & \dots & g_{r-1} & g_r \end{bmatrix}$$

  - G is a cyclic matrix (each row is obtained by shifting the previous row one column to the right).

8.8.2. Definition(Cont.)

- Since g(x) is the factor of *-1), that*

  -1) = g(x) h(x)

  Where h(x) is called check parity matrix

  - *If c(x) is codeword then c(x) h(x) = m(x) g(x) h(x) modulo -1) = 0*
  - h(x) = x +…+

- Check parity matrix H: is a cyclic matrix (each row is obtained by shifting the previous row one column to the right).
  - First row is h(x)

8.8.3.Encoding and decoding

- Encoding process is multiple generator polynomial g(x) with carrying information (message) polynomial m(x)
  - c(x) = m(x) g(x)
- Decoding process:
  - Syndrome S is remainder of division between received polynomial r(x) and g(x)
    - S = r(x) mod g(x) modulo -*1)*
    - *If S = 0 → codeword*
    - *If S* 0 → S = e(x) mod g(x) modulo -*1)*
      - *Can find error polynomial e(x) from S*

8.8.3.Encoding and decoding (Cont.)

- If generator matrix G is transformed into canonical form, codeword is in systematic form
  - c(x) = m(x) + d(x)

  Where d(x) is a polynomial has degree of n-k-1
- Since c(x) mod g(x) modulo-*1) = 0,*

  then d(x) = m(x) mod g(x) modulo-*1)*

8.8.4. Cyclic Redundancy Check Codes

- Is cyclic systematic code
- Used for send or store the information
- Codeword c(x) = m(x) – crc
  - crc = m(x) mod g(x) modulo – 1)
- Decoding
  - Let r(x) = m’(x) – crc’ where m’(x) = m(x) + (x); crc’=crc+ (x)
    - (x) first L symbol of e(x)
    - (x) remaining N-L symbols of e(x)
  - S= m’(x) mod g(x) modulo – 1) – crc’
    - S= 0 → no error
    - S 0 → S = (x) mod g(x) modulo – 1) - (x)
      - Calculate (x), (x) from S