PHỤ LỤC 8B1

MÃ HOÁ KÊNH KIỂM SOÁT LỖI

3.1. MỞ ĐẦU

❖ **Chức năng:** Mã hoá kênh kiểm soát lỗi là *quá trình xử lý tín hiệu số* nhằm đạt được truyền tin số tin cậy bằng cách *bổ xung có hệ thống* các ký hiệu dư vào luồng tin phát để phát hiện lỗi và sửa lỗi.

❖ **Vị trí:** Sau nguồn tin và trước điều chế sóng mang.

❖ **Nhận xét:** Khi thết kế hệ thống truyền dẫn số cần lưu ý hai thông số: Thông số tín hiệu phát và độ rộng băng tần của kênh truyền dẫn. Hai thông số này cùng với mật độ phổ công suất tạp âm thu xác định E_b/N_0.

- ✓ Do BER là một *hàm đơn trị* của E_b/N_0, nên khi E_b/N_0 cố định thì có thể cải thiện BER bằng cách dùng mã hoá kênh.
- ✓ Dùng mã hoá kênh kiểm soát lỗi *để dung hoà* giữa BER yêu cầu và E_b/N_0 dB (giảm công suất phát, giảm giá thành phần cứng như sử dụng anten kích thước nhỏ, tái sử dụng tần số....) chẳng hạn dùng mã hoá kênh kiểm soát lỗi trong các hệ thống thông tin di động.
- ✓ Để đánh giá lượng bit dư bổ sung phục vụ cho việc phát hiện và sửa lỗi của mã được định nghĩa bởi thông số tỉ lệ mã $r = \frac{R_b}{R_c}$ (Do bổ sung các bit dư vào bản tin phát dẫn đến luồng bit ra bộ lập mã có tốc độ bít R_c cao hơn tốc độ bit đầu vào R_b, tăng độ rộng băng tần $\Rightarrow$ ảnh hưởng đến hiệu quả sử dụng phổ tần hệ thống).

❖ Các cơ chế phát hiện và sửa lỗi

- ✓ Phát hiện bản tin bị lỗi, sau đó yêu cầu phía phát phát lại bản tin bị lỗi, vì thế cần có một kênh hồi tiếp.
- ✓ Phát hiện và sửa lỗi ở phía thu.

❖ Mục đích của mã hoá kênh kiểm soát lỗi

- ✓ Xác định đoạn nào của luồng số thu chứa lỗi. Thông báo cho nơi gửi hay nơi nhận về lỗi.
- ✓ Giảm thiểu xác suất không phát hiện lỗi.
- ✓ Giảm được BER (hay xác suất lỗi) với một E_b/N_0 định trước.
- ✓ Với BER cho trước giảm được E_b/N_0, lượng giảm này được gọi là độ lợi của mã đối với xác suất lỗi.

3.2. CÁC NGUYÊN TẮC MÃ HOÁ KÊNH KIỂM SOÁT LỖI

Nếu một tin có độ dài k bit $\Rightarrow$ có 2^k bản tin có thể có. Mong muốn thay thế từng bản tin nói trên bằng một từ mã tương ứng. Theo đó cần sử dụng các bit dư vì thế độ dài từ mã sẽ dài hơn độ dài bản tin.

Để phân tích khả năng phát hiện và sửa lỗi của mã trước hết định nghĩa khoảng cách Hamming giữa hai từ mã.

❖ **Định nghĩa khoảng cách Hamming:** *Khoảng cách Hamming giữa hai từ mã bất kỳ có cùng độ dài được định nghĩa là số vị trí mà ở đó chúng khác nhau.*

❖ Khả năng phát hiện và sửa lỗi

Nếu ký hiệu

d_m là khoảng cách Hamming cực tiểu giữa các từ mã có thể có trong tập mã.

t_{Detec} là số lỗi phát hiện được

t_{Corr} là số lỗi có thể sửa được

thì chúng phải thoả mãn các biểu thức (3.1) và (3.2) dưới đây.

➢ **Khả năng phát hiện t_{Detec} lỗi**

$$\begin{aligned} & d_m = t_{Detec} + 1 \\ \Leftrightarrow\ & t_{Detec} = d_m - 1 \end{aligned} \qquad (3.1)$$

➢ **Khả năng sửa t_{Corr} lỗi**

$$\begin{aligned} & d_m \geq 2t_{Corr} + 1 \\ \Leftrightarrow\ & t_{Corr} \leq \frac{d_m - 1}{2} \end{aligned} \qquad (3.2)$$

Mã kênh được phân thành hai loại mã khối tuyến tính và mã xoắn.

3.3. MÃ KHỐI TUYẾN TÍNH

❖ Khái niệm

✓ ***Khối bản tin:*** Luồng thông tin được chia thành các ***khối*** có độ dài bằng nhau được gọi là các khối bản tin. Độ dài k bit

✓ ***Từ mã (Codeword):*** Các bit nhận được ở đầu ra của bộ lập mã tương ứng với mỗi bản tin đầu vào được gọi là từ mã. Độ dài n bit

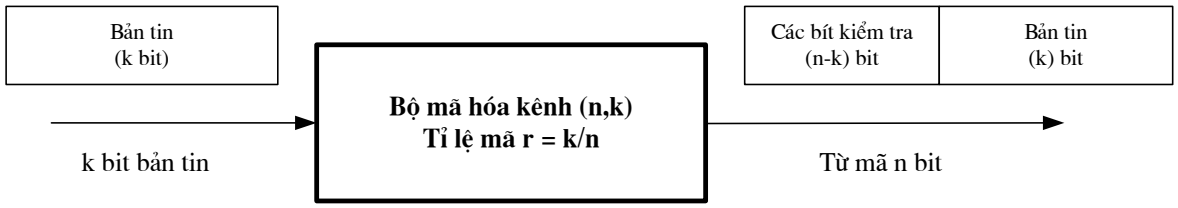
✓ ***Các bit kiểm tra:*** là các bit dư được bổ xung vào các khối bản tin theo một *thuật toán nhất định* tuỳ vào loại mã được dùng. Độ dài (n-k) bit.

- ✓ Mã khối được gọi là ***tuyến tính*** nếu thoả mãn sự kết hợp tuyến tính bất kỳ hai từ mã nào đó cũng là một từ mã thuộc mã đó. Trong trường hợp cơ hai tổng của hai từ mã bất kỳ cũng là một từ mã.

❖ Các thông số đặc trưng cho mã khối tuyến tính

- ✓ Độ dài khối bản tin k.
- ✓ Độ dài từ mã n.
- ✓ Khoảng cách Hamming cực tiểu d_m.
- ✓ Tỉ lệ mã $r = \frac{k}{n}$

Sơ đồ khối tổng quát của bộ mã hoá khối tuyến tính (n,k) được cho ở hình 3.4.



Hình 3.4. Sơ đồ bộ mã hóa khối tuyến tính (n,k)

Hoạt động của bộ lập mã có thể được biểu diễn ***toán học*** ở dạng ma trận tạo mã **G** hay đa thức tạo mã **g(x)**.

Tóm tắt: Lập mã khối thực hiện ánh xạ (sắp xếp) chuỗi **k** bit đầu vào thành chuỗi **n** bit đầu ra có các đặc điểm sau:

- ✓ Từ mã đầu ra bộ lập mã **C** chỉ phụ thuộc vào chuỗi bit đầu vào **m** hiện thời và ma trận tạo mã **G** (hay đa thức tạo mã g(x)) mà không phụ thuộc vào chuỗi đầu vào trước đó.
- ✓ Trong các mã khối tuyến tính, các từ mã tạo thành không gian con **k** chiều trong không gian **n** chiều (n,k).
- ✓ Các mã khối tuyến tính được mô tả dưới dạng ma trận tạo mã **G** có kích thước k×n, mỗi *từ mã* đầu ra C được viết dưới dạng.

$$C_{1\times n} = \underbrace{m_{1\times k}.G_{k\times n}}_{\text{số cột của ma trận m phải bằng số hàng của ma trận G}}$$

Số cột của ma trận m phải bằng số hàng của ma trận G

Trong đó **m** là chuỗi dữ liệu cơ hai đầu vào bộ mã hoá có độ dài **k** bit, lưu ý chỗi **n** bit 0 cũng là một từ mã thuộc mã khối tuyến tính (n,k).

3.3.1. Ma trận tạo mã và ma trận kiểm tra chẵn lẻ

❖ Ma trận tạo mã G

Nếu đưa vào bộ mã hóa một khối bản tin **k** bit, ***thì*** sẽ nhận được ở đầu ra bộ mã hóa một từ mã có độ dài **n** bit ở dạng chuỗi bit sau

$$\overbrace{c = \left[c_0, c_1, \ldots, c_{n-1}\right]_{1\times n}}^{\text{Hướng truyền} \Rightarrow}$$

$$= \underbrace{\underbrace{b_0, b_1, .., b_{n-k-1}}_{(n-k)\text{ bít kiểm tra chẵn lẻ}}, \underbrace{m_0, m_1, \ldots, m_{k-1}}_{\text{k bit bản tin}}}_{\text{n bit dầu ra bộ mã hoá}}$$

trong đó m_j là các bit của khối bản tin, b_i là các bit kiểm tra, các bit có chỉ số cao là các bit có nghĩa lớn hơn và được truyền trước.

$\Rightarrow$ ***<u>Như vậy:</u>*** *đầu vào bộ mã hóa có 2^k khối bản tin có thể có tương ứng ở đầu ra bộ lập mã có 2^k từ mã được sử dụng trong số 2^n từ mã có thể có (tương ứng với mỗi khối bản bin k bit đầu vào nhận được một từ mã n bit đầu ra)*

✓ ***Biểu diễn k bit bản tin vào dạng vector $1\times k$***

$$m = \left[m_0, m_1, \ldots, m_{k-1}\right]_{1\times k} \qquad \textbf{(3.3)}$$

✓ ***Biểu diễn n-k bit chẵn lẻ vào dạng vector $1\times(n-k)$***

$$b = \left[b_0, b_1, \ldots, b_{n-k-1}\right]_{1\times(n-k)} \qquad (3.4)$$

✓ ***Biểu diễn n bit ra ở dạng vector $1\times n$***

$$c = \left[c_0, c_1, \ldots, c_{n-1}\right]_{1\times n} \qquad (3.5)$$

trong đó

$$c_i = \begin{cases} \underbrace{b_i}_{(n-k)\text{ bit}} & i = 0,1,2,\ldots,n-k-1 \\ \underbrace{m_{i+k-n}}_{\text{k bit}} & i = n-k, n-k+1, \ldots, n-1 \end{cases} \qquad (3.6)$$

Các vector nói trên là các ***vector <u>hàng</u>***, các bit có chỉ số cao hơn là các bit có trọng số cao và được *<u>truyền trước</u>*, **(n-k)** bit kiểm tra chẵn lẻ được xác định bằng *tổng tuyến tính* của **k** bit bản tin theo tương quan tổng quát sau

$$\underbrace{b_i}_{\text{bit kiểm tra thứ i}} = \underbrace{p_{i0}m_0 + p_{i1}m_1 + \ldots + p_{ij}m_j + \ldots + p_{i,k-1}m_{k-1}}_{(n-k)\text{ bit kiểm tra chẵn lẻ là tổng tuyến tính của k bit bản tin}}$$

$$= \sum_{j=0}^{k-1} p_{ij}m_j \qquad (3.7)$$

trong đó

$$p_{ij} = \begin{cases} 1 & \text{Nếu } b_i \text{ phụ thuộc vào } m_j \\ 0 & \text{Nếu } b_i \text{ không phụ thuộc vào } m_j \end{cases}$$

và $\begin{cases} i = 0,1,...,n-k-1 \\ j = 0,1,2,...,k-1 \end{cases}$

Biểu diễn các phương trình trên gọn hơn ở dạng ma trận sau

$$b = mP$$

$$b_{1\times(n-k)} = m_{1\times k} P_{k\times(n-k)} \qquad \textbf{(3.8)}$$

trong đó **P** là ma trận k × (n-k) xác định bởi

$$P = \underbrace{\begin{bmatrix} p_{0,0} & p_{1,0} & \cdots & p_{n-k-1,0} \\ p_{0,1} & p_{1,1} & \cdots & p_{n-k-1,1} \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ p_{0,k-1} & p_{1,k-1} & \cdots & p_{n-k-1,k-1} \end{bmatrix}_{k\times(n-k)}}_{\text{Các phần tử ma trận } p_{ij}\text{, trong đó i là chỉ số cột, j là chỉ số hàng}} \qquad (3.9)$$

với i = 0, 1, 2,..,n-k-1 là chỉ số ***cột*** và j = 0, 1, 2,....k-1 là chỉ số ***hàng***.

$$b_{1\times(n-k)} = m_{1\times k} P_{k\times(n-k)}$$

$$= \underbrace{[m_0, m_1, ..., m_{k-1}]_{1\times k} \underbrace{\begin{bmatrix} p_{0,0} & p_{1,0} & \cdots & p_{n-k-1,0} \\ p_{0,1} & p_{1,1} & \cdots & p_{n-k-1,1} \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ .. & .. & .. & .. \\ p_{0,k-1} & p_{1,k-1} & \cdots & p_{n-k-1,k-1} \end{bmatrix}}_{k\times(n-k);\ \text{Các phần tử ma trận } p_{ij} \text{ trong đó i là chỉ số cột, j là chỉ số hàng}}}_{b_i = \sum_{j=0}^{k-1} p_{ij} m_j}$$

Từ các phương trình (3.3) – (3.7) biểu diễn **c** là một vector ***hàng*** được phân đoạn theo **m** và **b** như sau:

$$c = \underbrace{\left[\underbrace{b}_{(n-k)\text{ bit kiểm tra}} : \underbrace{m}_{k\text{ bit bản tin}}\right]_{1\times n}}_{\text{từ mã dầu ra n bit}} \tag{3.10}$$

Thay (3.8) vào (3.10) và đưa thừa số chung **m** ra ngoài thì

$$c_{1\times n} = m_{1\times k}\underbrace{\left[P_{k\times(n-k)} \quad I_{k\times k}\right]}_{G_{k\times n}} \tag{3.11}$$

trong đó I_k là ma trận đơn vị

$$I_k = \begin{bmatrix} 1 & 0 & . & . & 0 \\ 0 & 1 & . & . & 0 \\ . & . & . & . & . \\ . & . & . & 1 & 0 \\ 0 & 0 & . & 0 & 1 \end{bmatrix}_{k\times k} \tag{3.12}$$

✓ ***Ma trận tạo mã G được định nghĩa ở dạng sau***

$$G = \left[\underbrace{P}_{k\times(n-k)} \quad : \quad \underbrace{I_k}_{k\times k}\right]_{k\times n} \tag{3.13}$$

Khi này có thể viết biểu thức 3.11 ở dạng sau

$$C_{1\times n} = \underbrace{m_{1\times k}.G_{k\times n}}_{\text{số cột của ma trận m phải bằng số hàng của nma trận G}} \tag{3.14}$$

❖ Ma trận kiểm tra chẵn lẻ H

(Parity Check Matrix)

Giả sử **H** là ma trận (n-k)×n được xác định bởi

$$H = \left[I_{n-k} \quad \underbrace{P^T}_{\text{kích thước (n-k)×k}}\right]_{(n-k)\times n} \tag{3.15}$$

Trong đó P^T là ma trận chuyển vị của P có kích thước (n-k)×k, nhận được từ P bằng cách chuyển đổi cột thành hàng. Tương ứng có thể nhân các ma trận phân đoạn như sau

$$\begin{aligned} HG^T &= \left[I_{n-k} : P^T\right]\begin{bmatrix} P^T \\ \dots \\ I_k \end{bmatrix} \\ &= I_{n-k}P^T + P^T I_k \\ &= P^T + P^T \\ &= 0 \end{aligned} \tag{3.16}$$

Vậy: $H.G^T = 0 \Rightarrow G.H^T = 0$

Sau khi nhân cả hai vế của phương trình (3.14) với H^T nhận được

$$c.H^T = mG.H^T = 0 \qquad (3.17)$$

Ma trận **H** được gọi là ma trận kiểm tra chẵn lẻ

Như vậy, Ma trận tạo mã G được dùng để tạo mã ở phía phát, ma trận kiểm tra chẵn lẻ H được dùng để giải mã ở phía thu.

❖ Syndrome và phát hiện lỗi

Syndrome & Error Detection

➢ Khái niệm

Xét **mã** (n, k) với ma trận sinh **G** tương ứng có ma trận kiểm tra **H**. Khi phát một từ mã qua kênh có tạp âm, từ mã thu **y** là tổng của vector phát **c** và vector mẫu lỗi **e** nghĩa là

$$y = c + e \qquad (3.18)$$

Vector phát đi: $c = (c_0, c_1, c_2....c_{n-1})$

Vector mẫu lỗi: $e = (e_0, e_1, e_2....e_{n-1})$

Nếu $e_i = 1$ thì vector thu bị lỗi ở vị trí i, ngược lại nếu $e_i = 0$, vector thu không bị lỗi ở vị trí thứ i đó (i=1,2..,n).

Máy thu có nhiệm vụ giải mã vector **c** từ vector thu **y** (khôi phục **c** từ **y**). Để giải mã thực hiện tính toán Syndrome. Đặc điểm của Syndrome là chỉ phụ thuộc vào mẫu lỗi **e** mà không phụ thuộc vào từ mã **c** được phát và được xác định bởi:

$$S = y.H^T = e.H^T + \underbrace{C.H^T}_{=0} = e.H^T \qquad (3.20)$$

➢ Các thuộc tính của Syndrome

- ✓ **Thuộc tính 1:** *Syndrome chỉ phụ thuộc vào mẫu lỗi chứ không phụ thuộc vào từ mã được phát.*
- ✓ **Thuộc tính 2:** *Tất cả các mẫu lỗi khác nhau nhiều nhất một từ mã đều có cùng Syndrome.*

Với **k** bit bản tin ta có 2^k vector từ mã khác nhau được biểu thị bởi c_i, $i = \overline{0,2^k - 1}$.

Tương ứng với một mẫu lỗi **e** bất kỳ có thể định nghĩa 2^k vector $\mathbf{e_i}$ khác nhau như sau:

$$e_i = e + c_i, \qquad i = \overline{0,2^k - 1} \qquad (3.21)$$

chỉ khác nhau nhiều nhất một từ mã.

Tập các vector $\{e_i,\ i = \overline{0,2^k - 1}\}$ theo định nghĩa trên được gọi là Coset của mã. Nói cách khác, ***một Coset có đúng*** 2^k ***phần tử khác nhau nhiều nhất một từ mã***. Số các tổ hợp từ mã có thể có của một mã khối tuyến tính (n,k) là 2^n trong đó chỉ

có một bộ 2^k từ mã được dùng. Vậy tổng số bộ 2^k từ mã có thể có là 2^{n-k} (nghĩa là có 2^{n-k} bộ 2^k từ mã) và tương ứng sẽ có 2^{n-k} Coset có thể có (trong đó chỉ có một Coset là tương ứng với bộ từ mã được dùng).

- ✓ **Thuộc tính 3:** *Syndrome S là tổng các cột của ma trận H tương ứng với nơi xẩy ra lỗi.*
- ✓ **Thuộc tính 4:** *Bằng cách giải mã Syndronme, một mã khối tuyến tính (n,k) có thể sửa được **t** lỗi ở một từ mã, nếu **n** và **k** thoả mãn giới hạn Hamming sau đây.*

$$2^{n-k} \geq \sum_{i=0}^{t} \binom{n}{i} \tag{3.22}$$

trong đó $\binom{n}{i}$ là hệ số nhị thức được tính bởi.

$$\binom{n}{i} = \frac{n!}{(n-i)!i!} \tag{3.23}$$

Mã cơ hai thoả mãn giới hạn Hamming theo *dấu bằng* được gọi là *mã hoàn hảo*.

⇒ **Thấy rõ**, một mình Syndrome không thể xác định được mẫu lỗi **e**. Thực tế nói chỉ nhận ra được Coset chứa mẫu lỗi. Mẫu lỗi có khả năng xẩy ra nhất trong Coset được đặc trưng bởi Syndrome **S** là mẫu lỗi có xác suất lớn nhất (trong trường hợp coi rằng tạp âm của kênh là cộng).

⇒ **Vậy**, ***có thể tiến hành thuật toán giải mã như sau***:

- *Tínhnh Syndrome* $S = y.H^T$ *cho vector **y** thu.*
- *Trong tập Coset được đặc trưng bởi Syndrome trên, **chọn** mẫu lỗi có xác suất lớn nhất gọi là* $\boldsymbol{e_0}$.
- *Tính vector mã cho đầu ra*

$$c' = y + e_0$$

❖ Trọng lượng Hamming cực tiểu

Trọng lượng Hamming cực tiểu của ***một từ mã*** là tổng số các vị trí bit khác không trong từ mã xét hay nói cách khác là khoảng cách Hamming giữa một từ mã khác không với từ mã toàn không.⇒ Trọng lượng Hamming của ***mã*** là...

Do tính chất của mã khối tuyến tính là tổng (hoặc hiệu, ở số học Modul2 thì phép cộng & trừ là như nhau) hai từ mã bất kỳ luôn bằng một ***từ mã thứ ba*** của ***mã*** ⇒ nên trọng lượng Hamming cực tiểu của các từ mã khác không của mã khối tuyến tính chính bằng khoảng cách Hamming cực tiểu.

Note:

Thông số quan trọng của mã khối tuyến tính (xác định khả năng hiệu chỉnh lỗi của nó) là khoảng cách (Hamming) cực tiểu của **mã**, được định nghĩa là khoảng cách Hamming cực tiểu giữa hai từ mã riêng biệt ***bất kỳ***. Khoảng cách cực tiểu của mã được ký hiệu d_{min}

$$d_{min} = \min_{i \neq j} d_H(c_i, c_j)$$

Đối với các mã tuyến tính, khoảng cách cực tiểu bằng với trọng lượng cực tiểu của mã được định nghĩa là

$$w_{min} = \min_{c_i \neq 0} w(c_i)$$

Nghĩa là số các bit "1" ít nhất trong ***bất kỳ*** từ mã khác không nào đó.

Ví dụ 3.1: *Xét một họ mã được gọi là mã Hamming có các thông số sau*

Độ dài từ mã: $n = 2^m - 1$

Số các bit bản tin: $k = (2^m - 1) - m$

Số các bit chẵn lẻ: $m = n - k$

Trong đó $m \geq 3$.

Cụ thể xét mã Hamming (7,4) có n=7; k=4 tương ứng có m= 3.

Ma trận tạo **mã** của mã này phải có cấu trúc phù hợp với (3.13) và có dạng.

$$G = \left[\begin{array}{ccc:cccc} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ \underbrace{1}_{P} & \underbrace{0}_{} & \underbrace{1}_{} & \underbrace{0}_{} & \underbrace{0}_{} & \underbrace{0}_{} & \underbrace{1}_{I_k} \end{array}\right] \qquad (3.24)$$

Ma trận kiểm tra chẵn lẻ có dạng

$$H = \left[\begin{array}{ccc:cccc} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ \underbrace{0}_{I_{n-k}} & \underbrace{0}_{} & \underbrace{1}_{} & \underbrace{0}_{} & \underbrace{1}_{} & \underbrace{1}_{} & \underbrace{1}_{P^T} \end{array}\right] \qquad (3.25)$$

Các từ mã được tạo ra theo (3.14) **C=mG** và được cho ở bảng 3.1:

Bảng 3.1. Các từ mã của mã Hamming (7,4)

Bản tin	Từ mã	Trọng lượng	Bản tin	Từ mã	Trọng lượng
0000	000**0000**	0	1000	110**1000**	3
0001	101**0001**	3	1001	011**1001**	4
0010	111**0010**	4	1010	001**1010**	3
0011	010**0011**	3	1011	100**1011**	4
0100	011**0100**	3	1100	101**1100**	4
0101	110**0101**	4	1101	000**1101**	3
0110	100**0110**	3	1110	010**1110**	4
0111	001**0111**	4	1111	111**1111**	7

Cho thấy, trọng lượng nhỏ nhất của các từ mã khác không là 3 $\Rightarrow$ khoảng cách Hamming cực tiểu là d_{min} =3. Tất nhiên các mã Hamming có thuộc tính là khoảng cách Hamming cực tiểu luôn bằng 3 ***không phụ thuộc*** vào **m**. Do d_{min} =3 nên theo (3.1) và (3.2) các mã này chỉ có thể ***sửa được một lỗi*** và ***phát hiện được hai lỗi***.

$$C_{1\times n} = m_{1\times k}.G_{k\times n} = \left[\underbrace{b}_{(n-k)\text{ bit kiểm tra}} : \underbrace{m}_{k\text{ bit bản tin}}\right]$$

$$\Rightarrow C_i = m_i.G, \qquad i = 0,1,...,2^k - 1$$

Ví dụ: $c_{1\times 7} = [1\ \ 0\ \ 0\ \ 1]_{1\times 4}\left[\begin{array}{cccc:ccc} 1 & 1 & 0 & 1 & 0 & 0 & 0\\ 0 & 1 & 1 & 0 & 1 & 0 & 0\\ 1 & 1 & 1 & 0 & 0 & 1 & 0\\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{array}\right]_{4\times 7} = \left[\underbrace{011}_{b}\ \ \underbrace{1001}_{m}\right]$

(P: cột 1–3; I_k: cột 4–7)

❖ Quan hệ giữa d_{min} và H

Do c là một từ mã thuộc mã nên $c.H^T = mGH^T = 0 \Rightarrow$ Xét mã Hamming (7,4), tồn tại 16 từ mã thuộc mã **đều** làm cho $s = c.H^T = 0$ trong đó có

Bẩy từ mã có trong lượng = 3

Bẩy từ mã có trọng lượng = 4

Một từ mã có trọng lượng = 7

Một từ mã có trọng lượng = 0

$\Rightarrow d_{min}$ =3 $\Rightarrow$ Mối quan hệ giữa d_{min} và H là: d_{min} là *số cột nhỏ nhất* của ma trận kiểm tra chẵn lẻ H mà khi cộng chúng với nhau bằng 0. Chẳng hạn cho từ mã c = **0110100**.

Sử dụng công thức $c.H^T = mGH^T = 0$

$$C.H^T = \begin{bmatrix}0 & 1 & 1 & 0 & 1 & 0 & 0\end{bmatrix}_{1\times 7} \begin{bmatrix} \underbrace{1 & 0 & 0}_{I_{n-k}} : \underbrace{0 & 1 & 1 & 1}_{P^T} \\ 0 & 1 & 0 : 1 & 1 & 1 & 0 \\ 0 & 0 & 1 : 0 & 1 & 1 & 1 \end{bmatrix}^T$$

sau đó loại bỏ các cột của **H** tương ứng các vị trí có bit bằng 0 của ***từ mã*** nhận được tổng của các cột thứ 2 và thứ 3 và thứ 5 của **H** như sau:

$$\begin{bmatrix}0\\1\\0\end{bmatrix} + \begin{bmatrix}0\\0\\1\end{bmatrix} + \begin{bmatrix}0\\1\\1\end{bmatrix} = \begin{bmatrix}0\\0\\0\end{bmatrix}$$

$\Rightarrow$ **Vậy Syndrome = [0 0 0]**.

Nếu thực hiện cho 14 từ mã khác không còn lại *thuộc mã* thì Syndrome đều bằng 0.

❖ Quan hệ giữa Syndrome mẫu lỗi e

Xét mã Hamming (7,4). Vì mã này có d_{min} =3 nên chỉ có thể sửa được một lỗi $\Rightarrow$ nên với các mẫu lỗi đơn, áp dụng thuộc tính 3 (*Syndrome là tổng các cột của ma trận H tương ứng với nơi xẩy ra lỗi*) $\Rightarrow$ cho phép xác định được quan hệ giữa Syndrome và mẫu lỗi. Chẳng hạn

Nếu truyền c = [1 1 1 1 **1** 1 1] qua kênh, phía thu nhận được y = [1 1 1 1 **0** 1 1], xảy ra lỗi ở vị trí thứ 5 của c $\Rightarrow$ theo thuộc tính 3 thì Syndrome sẽ là cột thứ 5 của ma trận H nghĩa là

$$S = y.H^T = \begin{bmatrix}1 & 1 & 1 & 1 & 0 & 1 & 1\end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 : & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 : & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 : & 0 & \underbrace{1}_{\text{Note}} & 1 & 1 \end{bmatrix}^T$$

$$= \begin{bmatrix}1\\0\\0\end{bmatrix} + \begin{bmatrix}0\\1\\0\end{bmatrix} + \begin{bmatrix}0\\0\\1\end{bmatrix} + \begin{bmatrix}1\\1\\0\end{bmatrix} + \begin{bmatrix}1\\1\\1\end{bmatrix} + \begin{bmatrix}1\\0\\1\end{bmatrix}$$

$$= \underbrace{\begin{bmatrix}0\\1\\1\end{bmatrix}}_{\text{Cột thứ 5 của H}}$$

khi biết được Syndrome thì biết được vector mẫu lỗi là e = [0 0 0 0 **1** 0 0] $\Rightarrow$ sửa được lỗi ở vị trí 5 này $y_{cor} = y + e = [1\ 1\ 1\ 1\ 0\ 1\ 1] + [0\ 0\ 0\ 0\ 1\ 0\ 0] =$ **[1 1 1 1 1 1 1]**

Nếu phát c = [**0**111001] qua kênh, phía thu nhận được y = [**1**111001] $\Rightarrow$ s = [1 0 0] là hàng thứ nhất của H^T (cột thứ nhất của H) $\Rightarrow$ xác định được e = [1 0 0 0 0 0 0] $\Rightarrow$ sửa được y_{cor} = [**0**111001].

Tương tự xét tất cả các trường hợp còn lại ⇒ Quan hệ này đối với mã Hamming (7.4) được cho ở bảng 3.2 ⇒ Từ mã sai là từ mã không thuộc mã đó, khi này S = $y.H^T \neq 0$ vì $c.H^T = 0$

Bảng 3.2. Bảng giải mã cho mã Hamming (7.4)

Syndrome	Mẫu lỗi
000	**0000000**
100	1000000
010	0100000
001	0010000
110	0001000
011	0000100
111	0000010
101	0000001

3.3.2. Đa thức tạo mã

Mã vòng (*Cyclic Codes*) là một ***tập con*** của ***mã khối tuyến tính***, được dùng rộng rãi để phát hiện lỗi và sửa lỗi trong các hệ thống truyền thông. Lí do chính mà được dùng rộng rãi là dễ thực hiện bằng các mạch ghi dịch hồi tiếp tuyến tính (LFSR: Linear Feedback Shift Register).

❖ **Định nghĩa:** Một mã được gọi là mã vòng nếu thỏa mãn 2 thuộc tính cơ bản sau:

✓**Thuộc tính tuyến tính:** *Tổng của hai từ mã cũng là một từ mã.*

✓**Thuộc tính vòng:** *Mọi sự dịch vòng của một từ mã cũng sẽ là một từ mã.*

❖ Biểu diễn từ mã

➢ ***Dưới dạng vector gồm n thành phần:***

$$c = (c_0, c_1, c_2, ..., c_{n-1}) \tag{3.26}$$

➢ ***Dưới dạng đa thức bậc (n-1):***

$$\begin{aligned} c(x) &= c_0 + c_1 x + c_2 x^2 + ... + c_{n-1} x^{n-1} \\ &= \sum_{i=0}^{n-1} c_i x^i \end{aligned} \tag{3.27}$$

trong đó c_i=0/1 và mỗi luỹ thừa của x tương ứng với dịch vòng 1 bit theo thời gian. Vì thế khi nhân đa thức (3.27) với x có nghĩa là dịch vòng một bít sang phải, vì vậy x^{n-1}phải bằng 1 và bít ngoài cùng bên phải chuyển thành bit ngoài cùng bên trái. Quá trình nhân và dịch vòng như vậy được biểu diễn như sau:

$$x.c(x) \bmod (x^n - 1) = c_{n-1} + c_0 x + ... + c_{n-2}.x^{n-1} \quad (3.28)$$

trong đó **mod** có nghĩa là chia chỉ lấy phần dư.

Tương tự nếu nhân biểu thức (3.27) với x^2 có nghĩa dịch vòng hai bit sang phải và quá trình này được biểu diễn như sau:

$$x^2.c(x) \bmod (x^n - 1) = \underbrace{c_{n-2} + c_{n-1}x + ... + c_{n-3}.x^{n-1}}_{\text{chỉ có } c_i \text{ dịch chuyển còn bậc của x không dịch}} \quad (3.29)$$

Một mã vòng (n,k) được đặc tả bởi tập đầy đủ các đa thức bậc (n-1) hay thấp hơn và nhận một đa thức bậc thấp nhất (n-k) làm thừa số. Thừa số đặc biệt này được gọi là đa thức tạo mã **g(x)** của mã này. Đa thức tạo mã **g(x)** tương ứng với ma trận tạo mã **G** được đề cập ở phần 3.3.1 được biểu diễn như sau:

Đa thức tạo mã: Ký hiệu g(x) là đa thức tạo mã được biểu diễn bởi.

$$\begin{aligned} g(x) &= g_0 + g_1 x + g_2 x^2 + ... + g_{n-k}.x^{n-k} \\ &= \sum_{i=0}^{n-k} g_i x^i \end{aligned} \quad (3.10)$$

Đa thức bản tin: Ký hiệu m(x) là đa thức của khối bản tin k bit được biểu diễn bởi:

$$\begin{aligned} m(x) &= m_0 + m_1 x + m_2 x^2 + ... + m_{k-1} x^{k-1} \\ &= \sum_{i=0}^{k-1} m_i x^i \end{aligned} \quad (3.11)$$

❖ Các thuộc tính của đa thức tạo mã của mã vòng

- ✓ **<u>Thuộc tính 1:</u>** *Đa thức tạo mã của một mã vòng (n,k) là đơn nhất ở ý nghĩa rằng nó là đa thức từ mã bậc (n-k) cực tiểu duy nhất.*
- ✓ **<u>Thuộc tính 2</u>**: *Mọi bội số của đa thức tạo mã g(x) là một đa thức từ mã mới, xác định như sau:*

$$c(x) = a(x).g(x) \bmod (x^n - 1)$$

Trong đó a(x) là một đa thức của x.

Hay: *Một đa thức bậc n-1 hay thấp hơn là một đa thức từ mã khi và chỉ khi nó là bội số của g(x)*

⇒ Sử dụng thuộc tính 2 để xác định đa thức tạo mã c(x): *<u>Nếu</u>* cho một đa thức tạo mã g(x) để mã hóa một khối bản tin $(m_0, m_1, m_2, ..., m_{k-1})$ vào một mã vòng hệ thống ở dạng

$$\begin{aligned} c &= (c_0, c_1, c_2, ..., c_{n-1}) \\ &= \underbrace{\left(\underbrace{b_0, b_1, .., b_{n-k-1}}_{(n-k)\text{bít kiểm tra chẵn lẻ}}, \underbrace{m_0, m_1, ..., m_{k-1}}_{\text{k bit bản tin}} \right)}_{\text{n bit đầu ra bộ mã hoá}} \end{aligned}$$

và cho đa thức bản tin được định nghĩa bởi

$$m(x) = m_0 + m_1 x + m_2 x^2 + ... + m_{k-1} x^{k-1} = \sum_{i=0}^{k-1} m_i x^i \quad (3.32)$$

thì theo cấu trúc được định nghĩa cho một ***từ mã*** thì bit đầu của bản tin, m_{k-1}, là hệ số của x^{n-1} trong đa thức của ***từ mã***. Để đảm bảo được yêu cầu này nhân đa thức bản tin m(x) với x^{n-k} nhận được

$$x^{n-k}.m(x) = m_0 x^{n-k} + m_1.x^{n-k+1} + ... + m_{k-1}.x^{n-1} \quad (3.33)$$

Nếu chia đa thức $x^{n-k}.m(x)$ cho đa thức tạo mã g(x), thì nhận được thương a(x) và phần dư b(x) như sau:

$$\frac{x^{n-k}m(x)}{g(x)} = a(x) + \frac{b(x)}{g(x)} \quad (3.34)$$

hay:

$$x^{n-k}.m(x) = a(x).g(x) + b(x) \quad (3.35)$$

trong đó :

$$a(x) = a_0 + a_1 x + a_2 x^2 + ... + a_{k-1}.x^{k-1} = \sum_{i=0}^{k-1} a_i.x^i \quad (3.36)$$

và

$$b(x) = b_0 + b_1 x + b_2 x^2 + ... + b_{n-k-1}.x^{n-k-1} = \sum_{i=0}^{n-k-1} b_i.x^i \quad (3.37)$$

Trong số học modul 2 thì **b(x)=-b(x)**, nên có thể sắp xếp lại (3.35) lại như sau

$$b(x) + x^{n-k}.m(x) = a(x).g(x) \quad (3.38)$$

Đa thức này là ***bội số của g(x)***. Vì thế theo ***thuộc tính 2*** nó biểu thị một đa thức từ mã của mã tuần hoàn (n,k) được tạo bởi g(x).

Vậy từ mã đầu ra bộ lập mã được biểu diễn dưới dạng đa thức sau

$$c(x) = \underbrace{b(x)}_{\substack{\text{Bậc luôn nhỏ hơn} \\ \text{bậc của g(x): (n-k)}}} + \underbrace{x^{n-k}.m(x)}_{\substack{\text{Chỉ chứa các x có luỹ thừa} \\ \text{lớn hơn (n-k)}}} \quad \textbf{(3.39)}$$

Bậc của phần dư b(x) luôn nhỏ hơn bậc của số chia: **n-k**. Thành phần $x^{n-k}.m(x)$ chỉ chứa các x có luỹ thừa bằng hay lớn hơn **n-k**, $\Rightarrow$ việc cộng hai thành phần này như ở phương trình (3.39) sẽ không làm thay đổi bất cứ thành phần nào trong biểu thức.

Tóm tắt: Từ kết quả phân tích trên có thể tổng kết các bước trong quá trình mã hoá cho một mã vòng (n,k) như sau:

- ***Nhân đa thức bản tin m(x) với*** x^{n-k} ***nhận được*** $x^{n-k}.m(x)$

- ***Chia*** $x^{n-k}.m(x)$ ***cho g(x) để được phần dư b(x).***
- ***Cộng b(x) với*** $x^{n-k}.m(x)$ ***để nhận được đa thức từ mã c(x).***

✓ **Thuộc tính 3:** *Đa thức tạo mã g(x) và đa thức kiểm tra chẵn lẽ h(x) là các thừa số của đa thức* $1+x^n$ *như sau:*

$$h(x).g(x) = 1 + x^n \tag{3.40}$$

Trong đó h(x) là một đa thức chẵn lẻ được định nghĩa bởi :

$$h(x).g(x) \bmod (x^n - 1) = 0 \tag{3.41}$$

hay từ thuộc tính 2 nhận được

$$h(x).c(x) \bmod (x^n - 1) = 0 \tag{3.42}$$

Thuộc tính này cung cấp cơ sở để chọn đa thức *tạo mã* hay đa thức *kiểm tra chẵn lẻ*. Chẳng hạn có thể phát biểu rằng nếu đa thức g(x) là một đa thức bậc n-k đồng thời là một thừa số của $1+x^n$ thì nó là đa thức tạo mã của một mã vòng (n,k). Tương tự có thể phát biểu rằng nếu h(x) là một đa thức bậc k và đồng thời là thừa số của $1+x^n$ thì nó là đa thức kiểm tra chẵn lẻ của một mã vòng (tuần hoàn) (n,k).

Mọi thừa số của $1+x^n$ có bậc (n-k) (số các bít chẵn lẻ) đều có thể được sử dụng như một đa thức tạo mã. Khi giá trị n lớn, đa thức $1+x^n$ có thể có nhiều thừa số bậc (n-k). Chúng có thể tạo ra mã vòng tốt hoặc không tốt.

Ví du 3.2: Các mã Hamming

Để minh hoạ việc trình bầy mã vòng ở dạng đa thức, ta khảo sát quá trình tạo ra mã vòng (7,4) cho mã Hamming. Với độ dài từ mã n=7, thực hiện khai triển đa thức $1+x^7$ vào **ba** đa thức thừa số tối giản như sau

$$x^7 + 1 = (1+x)(1+x^2+x^3)(1+x+x^3)$$

Đa thức tối giản là đa thức không thể phân chia thành thừa số bằng cách sử dụng các đa thức có các hệ số là mã cơ số hai. Một đa thức tối giản bậc **m** được gọi là đa thức nguyên thuỷ nếu tồn tại quan hệ $n = 2^m - 1$, trong đó **n** là bậc của đa thức $1+x^n$ chia hết cho đa thức nguyên thuỷ. Vậy ở ví dụ đang xét chỉ có hai đa thức nguyên thuỷ là $(1+x^2+x^3)$ và $(1+x+x^3)$.

Xác định các đa thức g(x) và h(x)

Nếu chọn đa thức tạo mã **g(x)** với bậc bằng *số các bit chẵn lẻ (n-k)* bởi

$$g(x) = 1 + x + x^3$$

thì đa thức kiểm tra chẵn lẻ **h(x)** sẽ là

$$h(x) = (1+x)(1+x^2+x^3)$$
$$= 1 + x + x^2 + x^4$$

có bậc bằng *số các bit bản tin (k bit).*

Quá trình tạo từ mã c(x) theo đa thức tạo mã g(x)

Xét các thủ tục tạo mã cho khối bản tin **m = 1001** bằng cách sử dụng đa thức tạo mã nói trên.

Viết khối bản tin như sau:

$$m(x) = 1 + x^3$$

Nhân đa thức bản tin với $x^{n-k} = x^3$ *nhận được*

$$x^3.m(x) = x^3\left(1 + x^3\right) = x^3 + x^6$$

Chia đa thức trên cho đa thức tạo mã g(x) để nhận được phần dư b(x), kết quả được

$$\frac{\overbrace{x^3 + x^6}^{x^{n-k}m(x)}}{\underbrace{1 + x + x^3}_{g(x)}} = \underbrace{x + x^3}_{a(x)} + \frac{\overbrace{x + x^2}^{b(x)}}{\underbrace{1 + x + x^3}_{g(x)}}$$

Đa thức tạo mã c(x) được xác định theo (3.39):

$$c(x) = b(x) + x^{n-k}.m(x)$$
$$= \underbrace{x + x^2}_{b(x)} + \underbrace{x^3 + x^6}_{x^{n-k}.m(x)}$$

Kết quả ta được từ mã là $c = \overbrace{\underbrace{0\ 1\ 1}_{(n-k)=3 \text{ bit kiểm tra chẵn lẻ}}\ \underbrace{1\ 0\ 0\ 1}_{k=3 \text{ bit của khối bản tin}}}^{0.x^0+1.x^1+1.x^2+1.x^3+0.x^4+0.x^5+1.x^6}$ là từ mã có trong bảng 3.1 cho mã Hamming (7,4) .

⇒ **Tổng quát hoá:** *Mọi mã vòng được tạo ra bởi một đa thức nguyên thuỷ là một mã Hamming có khoảng cách hamming cực tiểu* $d_{min}=3$.

Quan hệ giữa g(x) với G và h(x) với H

Sẽ thấy rằng, đa thức tạo mã **g(x)** và đa thức kiểm tra chẵn lẻ **h(x)** xác định ***đơn trị*** ma trận tạo mã **G** và ma trận kiểm tra chẵn lẻ **H** từ ví dụ này.

Xây dựng ma trận tạo mã G từ đa thức tạo mã g(x).

Để xây dựng ma trận tạo mã **G** kích thước **4×7** ta bắt đầu bằng việc khảo sát bốn đa thức gồm đa thức tạo mã **g(x)** và ba đa thức dịch vòng của đa thức tạo mã **g(x)**

$$g(x) = 1 + x + x^3$$
$$x.g(x) = x + x^2 + x^4$$
$$x^2g(x) = x^2 + x^3 + x^5$$
$$x^3g(x) = x^3 + x^4 + x^6$$

Từ ***thuộc tính dịch vòng*** thấy rõ các đa thức g(x); xg(x);x^2g(x) và x^3g(x) là các ***đa thức từ mã*** của mã Hanning (7,4). Nếu sử dụng các ***hệ số*** của các đa thức nói trên cho các hàng của ma trận 4×7 nhận được.

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

Để nhận được *dạng hệ thống*, cộng hàng thứ nhất vào hàng thứ 3 rồi cộng tổng của hai hàng đầu với hàng thứ 4, ta được.

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

ma trận này giống hệt như ma trận ở ví dụ 3.1.

Xây dựng ma trận kiểm tra chẵn lẻ H từ đa thức kiểm tra h(x).

Trước hết, xét đa thức đảo của h(x) được định nghĩa là $x^k.h(x^{-1})$, các đa thức này cũng là các đa thức thừa số của $1+x^n$. Đối với ví dụ đang xét, ma trận **H** có kích thước là 3×7, vì vậy cần phải thiết lập **ba** đa thức gồm đa thức $x^4.h(x^{-1})$ và hai đa thức dịch vòng của nó như sau.

$$\begin{aligned} x^4h(x^{-1}) &= 1+x^2+x^3+x^4 \\ x^5.h(x^{-1}) &= x+x^3+x^4+x^5 \\ x^6h(x^{-1}) &= x^2+x^4+x^5+x^6 \end{aligned}$$

Nếu sử dụng các hệ số của các đa thức này cho các hàng của ma trận 3×7 nhận được ma trận **H** như sau:

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

Biến đổi ma trận H vào dạng hệ thống. Muốn vậy, cộng dòng thứ 3 vào dòng thứ nhất kết quả thế vào dòng thứ nhất nhận được H ở dạng hệ thống giống như **H** ở ví dụ 3.1 như sau:

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

Sơ đồ bộ mã hoá vòng

Trong phần trước đã xét các thủ tục mã hoá vòng (n,k) gồm ba bước sau:

- ✓ *Nhân đa thức bản tin m(x) với* x^{n-k} *nhận được* $x^{n-k}.m(x)$

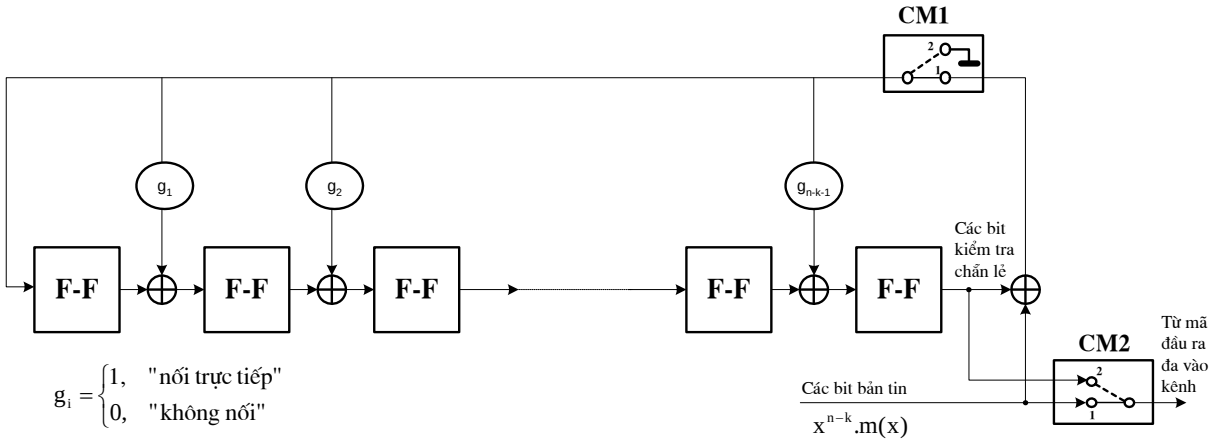
- ✓ *Chia* $x^{n-k}.m(x)$ *cho g(x) để được phần dư b(x).*
- ✓ *Cộng b(x) với* $x^{n-k}.m(x)$ *để nhận được đa thức từ mã c(x).*

Bộ mã hoá thực hiện ba bước này được cho ở hình 3.5 gồm một thanh ghi dịch **(n-k)** tầng có mạch *hồi tiếp tuyến tính*.

$$c = (c_0, c_1, c_2, ..., c_{n-1})$$

$$= \underbrace{\Big(\underbrace{b_0, b_1, .., b_{n-k-1}}_{\substack{(n-k)\text{ bít kiểm tra chẵn lẻ được xác định bởi nội dung của LFSR được dịch vào kênh ở n-k chu kỳ xung đồng hồ sau}}} , \underbrace{m_0, m_1, ..., m_{k-1}}_{\substack{\text{k bit bản tin được dịch vào kênh trong k chu kỳ xung đồng hồ đầu đồng thời được dịch vào LFSR}}} \Big)}_{\text{n bit đầu ra bộ mã hoá}}$$

Hướng truyền⇒



Hình 3.5. Bộ lập mã vòng (n,k) với đa thức tạo mã

$$g(x) = 1 + g_1.x + g_2.x^2 + ... + g_{n-k-1}.x^{n-k-1} + x^{n-k}$$

❖ Cấu tạo

- ✓ Các Flip-Flop (hay các Triger D, các phần tử trễ). Các Flip-Flop này nhận một trong hai trạng thái 0 hay 1. Một đồng hồ ngoài (không vẽ trong hình) điều khiển hoạt động tất cả các Flip-Flop.
- ✓ Một tập các mạch cộng Modul2 thực hiện cộng Modul2 giữa đầu ra của Flip-Flop với mạch hồi tiếp.
- ✓ Các bộ nhân để nhân chúng với nhau. $g_i = \begin{cases} 1, & \text{"nối trực tiếp"} \\ 0, & \text{"không nối"} \end{cases}$
- ✓ Hai chuyển mạch CM1 và CM2.

Mỗi khi có sườn tăng của xung đồng hồ, nội dung của thanh ghi dịch được dịch đi một vị trí theo chiều mũi tên (dịch sang phải).

❖ Hoạt động

Trước khi dịch k bit bản tin vào thanh ghi dịch, nội dung của thanh ghi dịch này được xoá về không.

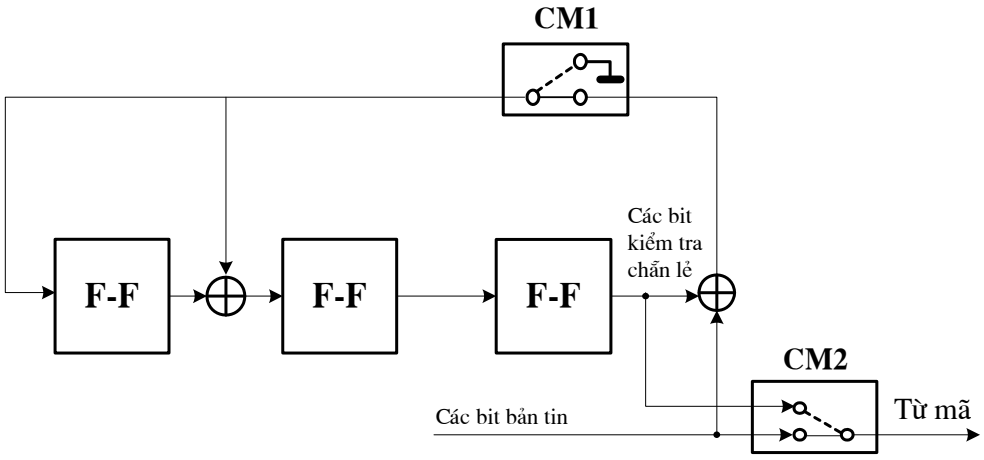
- ✓ ***Trong khoảng thời gian k chu kỳ xung đồng hồ đầu**: k bit bản tin (m_0, m_1,.....,m_{k-1}) tương ứng với $x^{n-k}.m(x)$ được dịch lần lượt vào kênh **đồng thời** chúng cũng được dịch vào thanh ghi dịch hồi tiếp tuyến tính LFSR để tạo ra (n-k) bit kiểm tra chẵn lẻ (các bit chẵn lẻ này có cùng giá trị như các hệ số của phần dư b(x))⇒ Vì vậy các khoá chuyển mạch CM1, CM2 đều ở vị trí dưới (vị trí 1).*
- ✓ ***Trong khoảng thời gian (n-k) chu kỳ xung đồng hồ tiếp theo: N**ội dung của thanh ghi dịch hồi tiếp tuyến tính LFSR chứa (n-k) bit kiểm tra chẵn lẻ được dịch vào kênh truyền ⇒ Vì vậy, c*ác khoá chuyển mạch CM1 và CM2 chuyển lên vị trí trên (vị trí 2).

⇒ *Kết quả từ mã đầu ra có cấu trúc* $c = \underbrace{\underbrace{b_0, b_1, .., b_{n-k-1}}_{\text{(n-k) bít kiểm tra chẵn lẻ được xác định bởi nội dung của LFSR được dịch vào kênh ở n-k chu kỳ xung đồng hồ sau}} , \underbrace{m_0, m_1, ..., m_{k-1}}_{\text{k bit bản tin được dịch vào kênh trong k chu kỳ xung đồng hồ đầu đồng thời được dịch vào LFSR}}}_{\text{Hướng truyền} \Rightarrow}$

Từ mã n bit đầu ra bộ mã hoá

Thí dụ 3.3: Bộ lập mã vòng Hamming (7,4)

Sơ đồ bộ lập mã vòng Hamming (7,4) có đa thức tạo mã $g(x) = 1 + x + x^3$ được cho ở hình 3.6:



Hình 3.6. Bộ lập mã vòng (7,4) với $g(x) = 1 + x + x^3$

Để mô tả hoạt động của bộ lập mã này xét chuỗi bản tin đầu vào là (1001). Thay đổi nội dung của thanh ghi dịch sau mỗi lần dịch vào một bit thông tin được cho ở bảng 3.3. Sau bốn lần dịch nội dung của thanh ghi dịch và các bit chẵn lẻ tương ứng là (0111). Vậy nếu gắn các bit này vào các bit tin nhận được từ mã (**011**1001) kết quả này giống hệt ở ví dụ 3.1.

Bảng 3.3. Nội dung thanh nghi dịch cho ví dụ 3.3 khi bản tin vào **(1001)**

Dịch	Bit vào	Nội dung thanh ghi
1	1	110
2	0	011
3	0	111
4	1	**011**

❖ Syndrome

Giả sử từ mã $c = (c_0,c_1,c_2,....,c_{n-1})$ được truyền qua kênh có tạp âm, thu được từ mã $y = (y_0,y_1,y_2,...,y_{n-1})$. Để giải mã từ mã thu này trước hết là tính Syndrome cho từ mã thu. Nếu Syndrome bằng không thì từ mã thu không bị lỗi, ngược lại nếu Syndrome khác không thì từ mã này bị mắc lỗi cần phải sửa nó. Đối với mã vòng ở dạng hệ thống thì có thể tính toán Syndrome dễ dàng.

Giả sử từ mã thu được trình bày ở dạng đa thức sau.

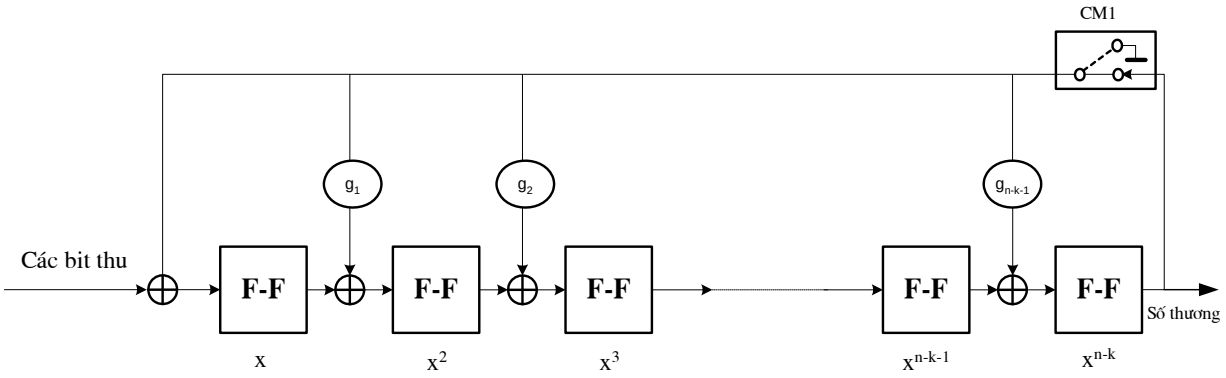
$$y = y_0 + y_1x + ... + y_{n-1}x^{n-1} \qquad (3.43)$$

Syndrome được tính bằng cách chia đa thức y(x) cho đa thức tạo mã g(x). Giả sử a(x) là thương còn s(x) là phần dư của kết quả chia, vì thể y(x) được biểu diễn như sau:

$$y(x) = a(x)g(x) + s(x) \qquad (3.44)$$

phần dư s(x) là một đa thức bậc ***(n-k-1)*** hay thấp hơn được gọi là đa thức Syndrome. Nếu đa thức s(x) khác không thì từ mã thu bị lỗi.

Sơ đồ bộ tính toán Syndrome giống như sơ đồ bộ lập mã chỉ khác ở chỗ các bit của từ mã thu được đưa vào (n-k) tầng của thanh ghi dịch có hồi tiếp (xem hình 3.7). Sau khi các bit của từ mã thu được dịch hết vào thanh nghi dịch, thì nội dung của thanh nghi dịch này xác định Syndrome S. Một khi biết được S sẽ xác định được mỗi lỗi và đưa ra được quyết định sửa như đã xét ở phần trước.

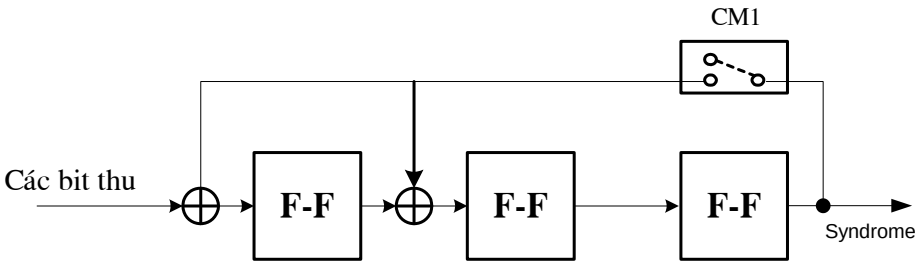


Hình 3.7. Bộ tính Syndrome của mã vòng (n,k) dựa trên đa thức tạo mã

$$g(x) = 1 + g_1.x + g_2.x^2 + ... + g_{n-k-1}.x^{n-k-1} + x^{n-k}$$

Thí dụ 3.4: Bộ tính Syndrome cho mã vòng (7,4)

Đối với mã Hamming vòng được tạo bởi bộ tạo mã $g(x)=1+x+x^3$ sơ đồ tính Syndrome được cho ở hình 3.8. Giả sử từ mã phát là (*011*1001) và từ mã thu là (011**0**001); nghĩa là bít giữa bị sai. Nội dung thanh ghi dịch của bộ tính Syndrome trong trường hợp này được cho ở bảng 3.4.



Hình 3.8. Bộ tính Syndrome cho mã (7,4) được tạo bởi đa thức $g(x)=1+x+x^3$.

Bảng 3.4. Nội dung ở bộ tính Syndrome ở hình 3.4 khi từ mã thu (0110001)

Dịch	Bit vào	Nội dung thanh ghi
		000 (trạng thái đầu)
1	1	100
2	0	010
3	0	001
4	0	110
5	1	111
6	1	001
7	0	**110**

Sau bẩy lần dịch Syndrome được xác định bằng **110**. Vì Syndrome khác không nên từ mã thu bị mắc lỗi. Từ bảng 3.2 nhận được mẫu lỗi tương ứng là 000**1**000 nghĩa là bit giữa bị lỗi.

3.3.3. Một số dạng mã vòng quan trọng

❖ Mã kiểm tra vòng dư

Các dạng mã vòng rất phù hợp cho việc *phát hiện lỗi vì hai lý do:* ***(1)*** chúng có khả năng phát hiện nhiều tổ hợp lỗi. ***(2)*** thực hiện các mạch điện lập mã và phát hiện lỗi đơn giản. Một dạng mã vòng thường dùng cho việc phát hiện lỗi là mã kiểm tra vòng dư CRC (Cyclic Redundance Check).

Định nghĩa *cụm lỗi* CRC độ dài B ở từ mã thu n bit là một dãy B bit trong đó các bit đầu, các bit cuối và một số bất kỳ bit trung gian thu bị mắc lỗi. Trong trường hợp các mã CRC cơ hai (n,k) có khả năng phát hiện các mẫu lỗi sau:

1. *Tất cả các cụm lỗi CRC độ dài (n-k) hay nhỏ hơn.*
2. *Một phần của các cụm lỗi có độ dài (n-k+1); phần này bằng* $1-2^{-(n-k-1)}$.
3. *Một phần của các cụm lỗi có độ dài lớn hơn (n-k+1); phần này bằng* $1-2^{-(n-k)}$
4. *Mọi tổ hợp của* $d_{min}-1$ *(hay nhỏ hơn) lỗi*
5. *Tất cả các mẫu lỗi có số lỗi lẻ nếu số các hệ số khác không của đa thức tạo mã g(x) chẵn.*

Bảng 3.5 trình bày đa thức tạo mã của ba mã CRC được công nhận là chuẩn quốc tế.

Bảng 3.5. Các mã CRC

Mã	Đa thức tạo mã
CRC-12	$1+x+x^2+x^3+x^{11}+x^{12}$
CRC-16	$1+x^2+x^{15}+x^{16}$
CRC-CCITT	$1+x^5+x^{12}+x^{16}$

❖ Mã có độ dài cực đại

Đối với số nguyên dương m≥3, tồn tại một mã có độ dài cực đại với các thông số sau:

Độ dài khối: $n=2^m-1$

Số bit bản tin: $k=m$

Khoảng cách tối thiểu: $d_{min}=2^m-1$

Đa thức tạo mã độ dài cực đại được hình thành theo:

$$g(x)=\frac{1+x^n}{h(x)} \qquad (3.45)$$

trong đó h(x) là một đa thức nguyên thuỷ bất kỳ bậc m. **Vì** *mọi mã vòng* được tạo ra bởi đa thức nguyên thuỷ *là mã Hamming* có độ dài tối thiểu là *ba* **nên** các mã có độ dài cực đại là kép đôi của mã Hamming. Mã độ dài cực đại còn được gọi là mã ngẫu nhiên vì các đặc tính tương quan và phổ của nó giống như tạp âm trắng.

❖ Mã Golay

Mã Golay (23,12) là một mã cơ hai đặc biệt quan trọng, nó có khả năng sửa mọi tổ hợp ba hay ít hơn các lỗi ngẫu nhiên trong một khối 23 bit. Mã có khoảng cách cực tiểu bằng 7. Ngoài ra đây là một mã hoàn hảo vì nó thoả mãn giới hạn Hamming ở

phương trình (3.2) đối với $t_{cor} = 3$ở dấu bằng. Mã Golay (23,12) là mã duy nhất được biết là mã hoàn hảo cơ số hai sửa 3 lỗi.

Đa thức tạo mã cho mã Golay có thể là:

$$g_1(x) = 1 + x^2 + x^4 + x^5 + x^6 + x^{10} + x^{11} \quad (3.46)$$

hay

$$g_2(x) = 1 + x + x^5 + x^6 + x^7 + x^9 + x^{11} \quad (3.47)$$

Cả hai đều là thừa số của đa thức $1 + x^{23}$

$$1 + x^{23} = (1 + x).g_1(x).g_2(x)$$

Rất tiếc không thể *tổng quát hoá* mã Golay cho mọi tổ hợp các thông số mã ***n*** và ***k***.

❖ Các mã Bose-Chaudhuri-Hocquenqhem (BCH)

Một trong các lớp mã khối tuyến tính mạnh và quan trọng nhất là các mã BCH, đây là các mã vòng có đa dạng các thông số. Các mã BCH chung nhất được đặc trưng như sau: Đối với các số nguyên m (m≥3) bất kỳ và $t < \frac{2^m - 1}{2}$ tồn tại một mã cơ hai với các thông số sau:

Độ dài khối: $n = 2^m - 1$

Số bit bản tin: $k \geq n - m.t$

Khoảng cách tối thiểu: $d_{min} \geq 2.t - 1$

Mỗi mã BCH là một mã sửa t lỗi ở chỗ có thể phát hiện sửa được t lỗi ngẫu nhiên trên một từ mã. Cũng có thể trình bày các mã sửa lỗi đơn Hamming như là các mã BCH.

Các mã BCH cho phép linh hoạt chọn lựa các thông số như: độ dài từ mã và tỷ lệ mã. Ngoài ra, ở các độ dài khối vài trăm hoặc ít hơn các mã BCH nằm trong số các mã có độ dài khối và tỷ lệ mã như nhau được biết đến nhiều nhất.

Bảng 3.6 trình bày các thông số của mã và các đa thức tạo mã cho các mã khối cơ hai BCH có độ dài đến $2^5 - 1$.

Bảng 3.6. Các mã BCH cơ hai có độ dài đến $2^5 - 1$

n	k	t	Các đa thức tạo mã
7	4	1	1 011
15	11	1	10 011
15	7	2	111 010 001
15	5	3	10 100 110 111
31	26	1	100 101
31	21	2	11 101 101 001

31	16	3	1 000 111 110 101 111
31	11	5	101 100 010 011 011 010 101
31	6	7	11 001 011 011 110 101 000 100 111

Lưu ý: n = độ dài khối.

k = số các bit thông tin

t = số các lỗi có thể phát hiện được cực đại.

Các hệ số bậc cao của đa thức tạo mã g(x) ở phía phát.

Thí dụ: Giả thiết, muốn xây dựng một đa thức tạo mã cho mã BCH(15,7). Từ bảng trên ta có (111 010 001) cho các hệ số của đa thức tạo mã. Vậy

$$g(x) = x^8 + x^7 + x^6 + x^4 + 1$$

❖ Các mã Reed-Solomon

Các mã Reed-Solomon là một phân lớp quan trọng của các mã BCH không phải cơ hai; các mã này thường được viết tắt là các mã RS. Bộ mã hoá cho một mã RS khác với bộ mã hoá cơ hai ở chỗ nó tác động lên nhiều bit chứ không phải bit riêng lẻ. Chẳng hạn bộ mã hoá cho một mã RS(n,k) trên cơ sở các ký hiệu m bit nhóm các luồng số liệu cơ hai thành các khối, mỗi khối có độ dài km bit. Mỗi khối sẽ được xử lý như là k ký hiệu. Thuật toán mã hoá k ký hiệu thành n ký hiệu bằng cách cộng thêm (n-k) ký hiệu dư. Như vậy, từ mã chứa nm bit. Khi m là một nhận được từ luỹ thừa hai của một số nguyên, các ký hiệu m bit nói trên được gọi là các byte. Giá trị phổ biến của m = 8 tất nhiên các mã RS 8 bit là các mã rất mạnh.

Một mã RS hiệu chỉnh t_{corr} lỗi có các thông số sau:

Độ dài khối: $n = 2^m - 1$ *các ký hiệu*

Kích thước bản tin: k *ký hiệu*

Kích thước kiểm tra chẵn lẻ: $n - k = 2.t_{corr}$ *ký hiệu*

Khoảng cách cực tiểu: $d_{min} = 2.t_{corr} + 1$ *ký hiệu*

Độ dài của mã RS nhỏ hơn một so với kích thước của một ký hiệu mã và khoảng cách cực tiểu lớn hơn một so với các ký hiệu chẵn lẻ. Tất nhiên khoảng cách cực tiểu luôn bằng khoảng cách thiết kế của mã. Có thể chỉ ra rằng khong có mã khối tuyến tính nào có khoảng cách cực tiểu lớn hơn (n-k+1). Một mã khối tuyến tính (n,k) có khoảng cách cực tiểu bằng (n-k+1) được gọi là mã với khoảng cách cực đại có thể tách được. Vì thế, các mã RS là các mã với khoảng cách cực đại có thể phân tách được. Ngoài ra, các mã RS cho phép chọn lựa nhiều tỷ số mã khác nhau để tối ưu chất lượng. Sau cùng đã có các kỹ thuật giải mã hiệu quả cho mã RS.

Ví du 3.5:

Ta khảo sát một mã RS hiệu chỉnh lỗi đơn có một byte (ký hiệu) 2 bit. Nghĩa là, t_{corr}=1 và m=2. Bằng cách ký hiệu bốn ký hiệu có thể có là 0,1,2,3 ta có thể viết lại dạng trình bầy cơ số hai của chúng như sau:

0: 00

1: 01

2: 10

3: 11

Mã này có các thông số sau:

$n = 2^2 - 1 = 3$ byte =6 bit

$n - k = 2$ byte = 4 bit

Tỉ lệ mã $r = k/n = 1/3$

Mã có thể hiệu chỉnh mọi lỗi cụm đồng pha (đoạn một ký hiệu) có độ dài 2.

3.4. MÃ XOẮN

3.4.1. Mở đầu

❖ Các thông số đặc trưng cho mã khối tuyến tính gồm hai thông số n, k, và một ma trận G hay đa thức tạo mã. Trong đó

- ✓ k là độ dài của khối bit bản tin đầu vào bộ lập mã
- ✓ n là độ dài (tổng số ký hiệu) đầu ra của bộ lập mã
- ✓ r = k/n là tỉ lệ mã, đánh giá lượng dư bổ sung

❖ Các thông số đặc trưng cho mã xoắn bởi ba thông số n, k và K. Trong đó

- ✓ **k** là số bit dịch vào bộ lập mã tại cùng một thời điểm (k bit đồng thời vào bộ lập mã).
- ✓ **n** là số bit ở đầu ra bộ lập mã khi cho k bit đồng thời vào bộ lập mã.
- ✓ **K** là độ dài hạn chế thể hiện số lần dịch cực đại của một nhóm k bit bản tin vào mà nhóm k bit này vẫn còn gây ảnh hưởng tới đầu ra bộ lập mã.
- ✓ **r** = k/n là tỉ lệ mã (k<n), tuy nhiên n không xác định độ dài từ mã như trường hợp bộ lập mã khối.
- ✓ Đặc tính quan trọng của mã xoắn khác biệt so với mã khối là bộ lập mã của chúng có nhớ.

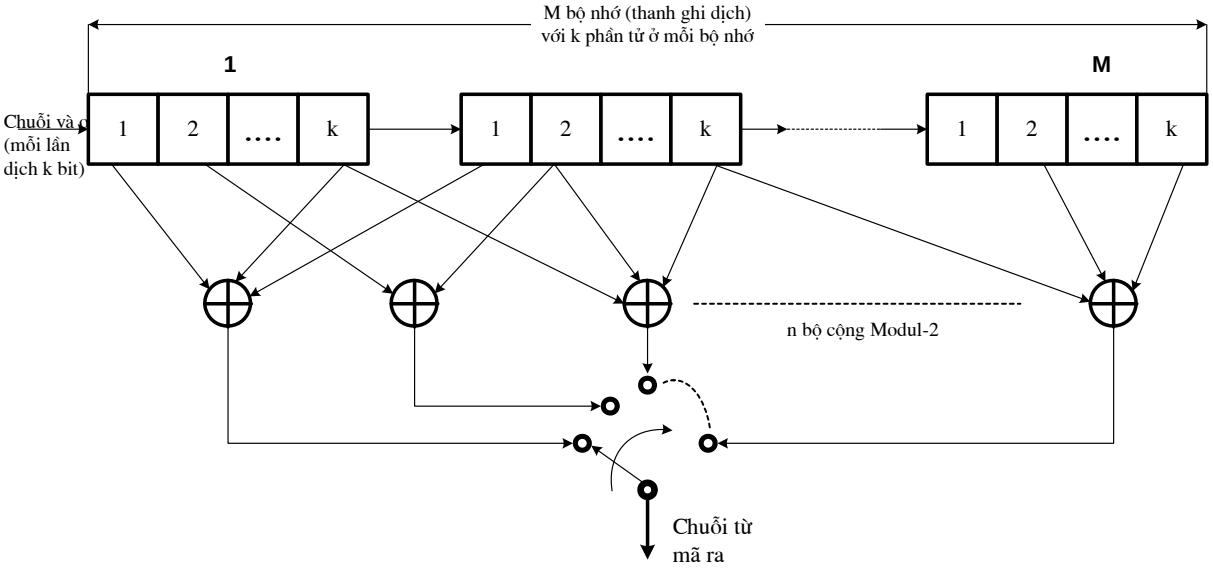
3.4.2. Lập mã xoắn

❖ Sơ đồ khối tổng quát

Sơ đồ tổng quát của một bộ lập mã xoắn được cho ở hình 3.9. Theo đó, bộ lập mã xoắn gồm: ***(1)*** M bộ nhớ (tầng nhớ) mỗi tầng có k phần tử nhớ (k bit ở mỗi tầng). ***(2)*** n bộ tạo hàm đại số tuyến tính ở dạng cộng Modul-2 tương ứng với số các bit đầu ra

bộ lập mã cho mỗi lần dịch k bit tin vào đồng thời. ***(3)*** quan hệ giữa K và M được xác định là: $K = \begin{cases} M+1, & \text{nếu dầu vào bộ nhớ thứ nhất dược nối dến bộ cộng co số hai} \\ M, & \text{nếu dầu vào bộ nhớ thứ nhất không dược nối dến bộ cộng co số hai} \end{cases}$

trong đó độ dài hạn chế K được định nghĩa là số lần dịch cực đại của một nhóm k bit bản tin vào đồng thời mà nhóm k bit này vẫn còn ảnh hưởng tới đầu ra bộ lập mã.



Hình 3.9. Sơ đồ tổng quát của một bộ lập mã xoắn với tỷ lệ mã k/n

❖ Nguyên lý hoạt động

Mỗi khi dịch đồng thời k bit tin vào bộ lập mã hoá xoắn, được mã hoá thành n bit đầu ra, mối quan hệ giữa k bit tin đầu vào và n bit đầu ra được xác định bởi một trong các phương pháp sau: ***(1)*** ma trận tạo mã; ***(2)*** chuỗi tạo mã; ***(3)*** đa thức tạo mã; ***(4)*** biểu đồ hình cây; ***(5)*** biểu đồ hình lưới; ***(6)*** biểu đồ trạng thái. Đặc tính quan trọng của mã xoắn khác biệt so với mã khối là bộ lập mã của chúng có nhớ $\Rightarrow$ vì thế quá trình tạo ra n bit đầu ra *không chỉ* phụ thuộc vào k bit tin đầu vào đồng thời *mà còn* phụ thuộc vào (K-1) tập hợp k bit đầu vào trước đó. Hay nói cách khác n bit đầu ra (được lấy mẫu từ các đầu ra của n bộ cộng Modul-2 tương ứng) ***không chỉ*** phụ thuộc vào k bit vào đồng thời ***mà còn*** phụ thuộc vào trạng thái trước đó của bộ lập mã (là một trạng thái thuộc một trong số $2^{(K-1)k}$ trạng thái có thể có của bộ lập mã).

Nếu k=1 (mỗi lần dịch vào một bit)$\Rightarrow$ bộ lập mã xoắn có M tầng nhớ trở thành thành ghi dịch M tầng. Mỗi khi dịch một bit tin vào bộ lập mã thì tất cả các bit trước đó được dịch sang bên phải một tầng, tuỳ theo cấu trúc cụ thể bộ lập mã (được xác định bởi đa thức tạo mã) mà nhận được n bit cụ thể được truyền vào kênh.

Vì vậy có thể dùng các phương pháp kể trên (ma trận tạo mã, đa thức tạo mã, chuỗi tạo mã, biểu đồ hình cây hoặc hình lưới và biểu đồ trạng thái) để nghiên cứu hoạt động của bộ lập mã, trong đó dùng các ký hiệu chỉ số sau:

- ✓ $\ell = 1,2,...,L$: là số thứ tự của chuỗi bit vào có độ dài **L**
- ✓ $q = 1,2,...,k$: là số thứ tự đầu vào

- ✓ $j = 1,2,...,n$: là số thứ tự đầu ra.
- ✓ $p = 0,1,2,...,M$: là số thứ tự phần tử nhớ (p = 0 tương ứng với đầu vào của bộ nhớ thứ nhất)
- ✓ i : là thời điểm xét.

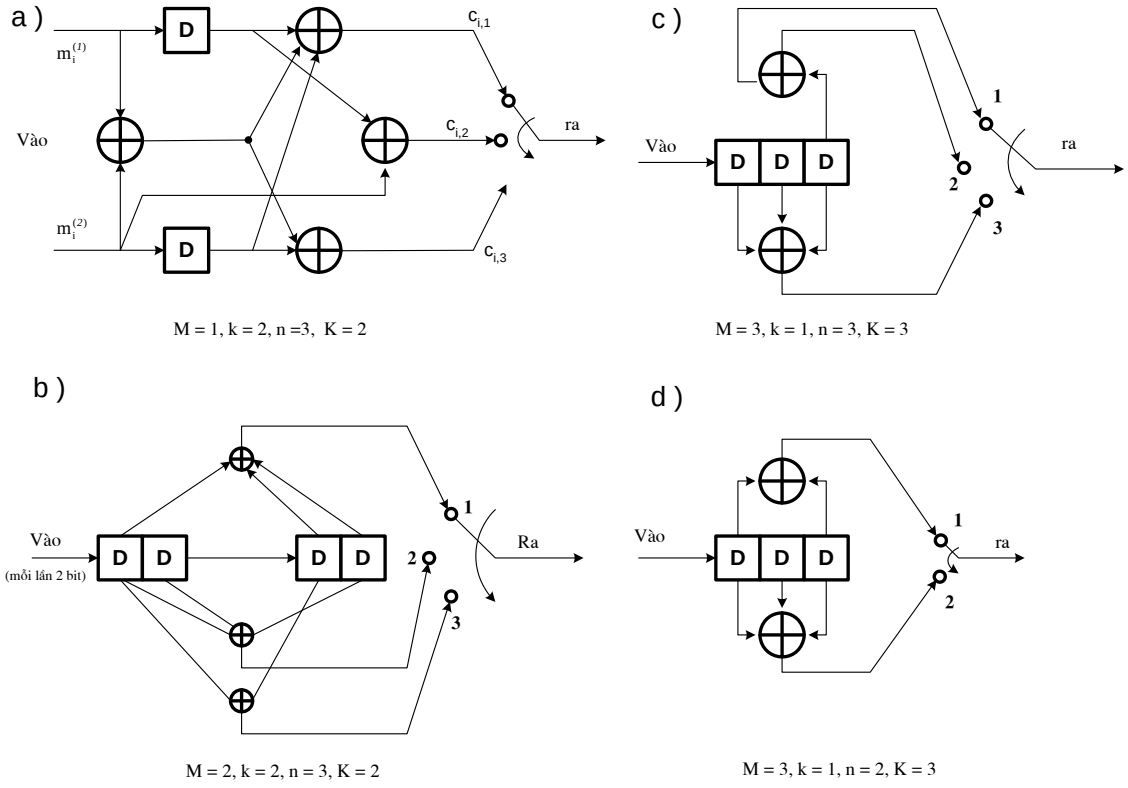
$$p = \begin{cases} 0,1,2,...,M, & \text{khi dầu vào bộ lập mã dược nối dến nhánh cộng Modul2} \\ 1,2,...,M, & \text{khi dầu vào bộ lập mã không dược nối dến nhánh cộng Modul2} \end{cases}$$

❖ Xác định các bit độn

- ✓ *Độn loại 1*: Mục đích làm cho độ dài bản tin vào là số nguyên lần k. Theo đó, nếu độ dài chuỗi bit tin vào không phải là bội số của k thì phải độn các bit "0" để độ dài là bội số của k khi này độ dài bản tin đầu vào trên mỗi đầu vào q đều bằng **L** $\Rightarrow$ chuỗi bit bản tin đầu vào là $L_m = L \times k$
- ✓ *Độn loại 2*: Mục đích là đưa bộ lập mã về trạng thái khởi đầu toàn không (*các đoạn bản tin dài L bit không ảnh hưởng lên nhau*), muốn vậy phải độn (K-1)k bit "0" vào cuối chuỗi bản tin trước (đầu bản tin tiếp theo) khi bản tin tiếp theo được nạp vào các tầng nhớ của bộ lập mã.

Lưu ý : Số lượng các bit "0" được độn phụ thuộc vào các thông số đặc trưng cụ thể của sơ đồ lập mã và việc độn được thực hiện trước khi đưa chuỗi bit tin vào bộ lập mã.

Thí dụ: Bộ lập mã xoắn được cho ở các hình 3.10a,b,c,d.



Hình 3.10. Các thí dụ về các bộ lập mã xoắn

3.4.3. Ma trận tạo mã

❖ Mục đích

Nghiên cứu trinhg bầy hoạt động bộ lập mã xoắn bằng cách sử dụng ma trận tạo mã khi cho sơ đồ bộ lập mã

❖ Cách tiếp cận

Để trình bày hoạt động bộ lập mã xoắn (nghĩa là tìm chuỗi bit đầu ra bộ lập mã khi cho chuỗi bit đầu và sơ đồ bộ lập mã) ở *dạng ma trận* giống như đối với mã khối tuyến tính, ta cần phải xác định ma trận tạo mã G và ma trận kiểm tra H như sau:

<u>Bước 1</u>: Xác định ma trận tạo mã từ sơ đồ bộ lập mã gồm

✓***Biểu diễn phương trình vào/ra của sơ đồ lập mã đã cho.***

✓***Biểu diễn phương trình vào ra theo toán tử trễ x.***

✓***Xác định kích thước và giá trị cụ thể cho các phần tử của ma trận tạo mã G***

Xác xác định kích thước và các phương trình vào/ra theo các phần tử của G.

Từ các phương trình tổng quát

$$c_i = m.G; \quad s = y.H^T \text{ và } c_i H^T = 0$$

trong đó $m = \left(m_i^{(1)},...,m_i^{(q)},...,m_i^{(k)}\right)_{1\times k}$ là k bit vào

$c_i = \left(c_{i,1},...,c_{i,j},...,c_{i,n}\right)_{1\times n}$ là n bit đầu ra của bộ mã hóa tại thời điểm i.

Xác định kích thước và các phần tử của ma trận tạo mã G tương ứng với các thông số đặc trưng cụ thể của sơ đồ lập mã đã cho bằng cách

$$G_p = \begin{bmatrix} g_{p,1}^{(1)}, & g_{p,2}^{(1)} & ,....., & g_{p,j}^{(1)} & ,..., & g_{p,n}^{(1)} \\ g_{p,1}^{(2)}, & g_{p,2}^{(2)} & ,..., & g_{p,j}^{(2)} & ,..., & g_{p,n}^{(2)} \\ : & : & : & : & : & : \\ g_{p,1}^{(q)}, & g_{p,2}^{(q)} & ,..., & g_{p,j}^{(q)} & ,..., & g_{p,n}^{(q)} \\ : & : & : & : & : & : \\ g_{p,1}^{(k)}, & g_{p,2}^{(k)} & ,..., & g_{p,j}^{(k)} & ,..., & g_{p,n}^{(k)} \end{bmatrix}$$

trong đó k, n, q,p là các giá trị cụ thể tương ứng với sơ đồ đã cho, khi này tìm được các phương trình vào/ra bộ lập mã theo các phần tử của m và các phần tử của ma trận tạo mã G nghĩa là

$$c_i = \left(m_i^{(1)},...,m_i^{(q)},...,m_i^{(k)}\right)_{1\times k} \times \begin{bmatrix} g_{p,1}^{(1)}, & g_{p,2}^{(1)} & ,...., & g_{p,j}^{(1)} & ,...., & g_{p,n}^{(1)} \\ g_{p,1}^{(2)}, & g_{p,2}^{(2)} & ,...., & g_{p,j}^{(2)} & ,...., & g_{p,n}^{(2)} \\ : & : & : & : & : & : \\ g_{p,1}^{(q)}, & g_{p,2}^{(q)} & ,...., & g_{p,j}^{(q)} & ,...., & g_{p,n}^{(q)} \\ : & : & : & : & : & : \\ g_{p,1}^{(k)}, & g_{p,2}^{(k)} & ,...., & g_{p,j}^{(k)} & ,...., & g_{p,n}^{(k)} \end{bmatrix}_{k\times n}$$

$$= \left(c_{i,1},...,c_{i,j},...,c_{i,n}\right)_{1\times n}$$

$$c_{i,1} = m_i^{(1)}.g_{p,1}^{(1)} + m_i^{(2)}.g_{p,1}^{(2)} + ... + m_i^{(q)}.g_{p,1}^{(q)} + ... + m_i^{(k)}.g_{p,1}^{(k)}$$

$$c_{i,2} = m_i^{(1)}.g_{p,2}^{(1)} + m_i^{(2)}.g_{p,2}^{(2)} + ... + m_i^{(q)}.g_{p,2}^{(q)} + ... + m_i^{(k)}.g_{p,2}^{(k)}$$

:

$$c_{i,j} = m_i^{(1)}.g_{p,j}^{(1)} + m_i^{(2)}.g_{p,j}^{(2)} + ... + m_i^{(q)}.g_{p,j}^{(q)} + ... + m_i^{(k)}.g_{p,j}^{(k)}$$

:

$$c_{i,n} = m_i^{(1)}.g_{p,n}^{(1)} + m_i^{(2)}.g_{p,n}^{(2)} + ... + m_i^{(q)}.g_{p,n}^{(q)} + ... + m_i^{(k)}.g_{p,n}^{(k)}$$

Giá trị cụ thể cho các phần tử của ma trận tạo mã G

Bằng cách so sánh các phương trình vào ra của sơ đồ theo các phần tử $g_{p,j}^{(q)}$ của ma trận tạo mã G với các phương trình vào/ra theo toán tử trễ x, ta sẽ tìm được giá trị cụ thể cho các phần tử của ma trận tạo mã đặc trưng cho sơ đồ lập mã đã cho.

<u>Bước 2:</u> Ma trận kiểm tra chẵn lẻ H

Vì mối quan hệ giữa ma trận kiểm tra chẵn lẻ H và ma trận tạo mã G thông qua ma trận P nghĩa là

$$G = \left[I_k : P_{k\times(n-k)}\right]_{k\times n}$$

$$H = \left[P_{(n-k)\times k}^T : I_{n-k}\right]_{(n-k)\times n}$$

trong đó I_k là ma trận đơn vị k x k, lưu ý ở đây ma trận I_k đứng trước P **vì** các m_i có chỉ số i thấp là các bit vào trước, nên ta tìm được P bằng cách biến đổi G về dạng hệ thống, sau đó tìm H

<u>Bước 3:</u> Xác định dạng hồi tiếp của sơ đồ lập mã

Bằng cách dùng phương trình $c_i H^T = 0$, ta sẽ tìm được phương trình thiết kế sơ đồ hồi tiếp của bộ lập mã.

<u>Ví dụ:</u> Trình bầy hoạt động của bộ lập mã được cho ở hình 3.10a bằng phương pháp ma trận tạo mã.

<u>Bước 1</u>: Xác định ma trận tạo mã từ sơ đồ bộ lập mã gồm

✓***Biểu diễn phương trình vào/ra của sơ đồ lập mã đã cho.***

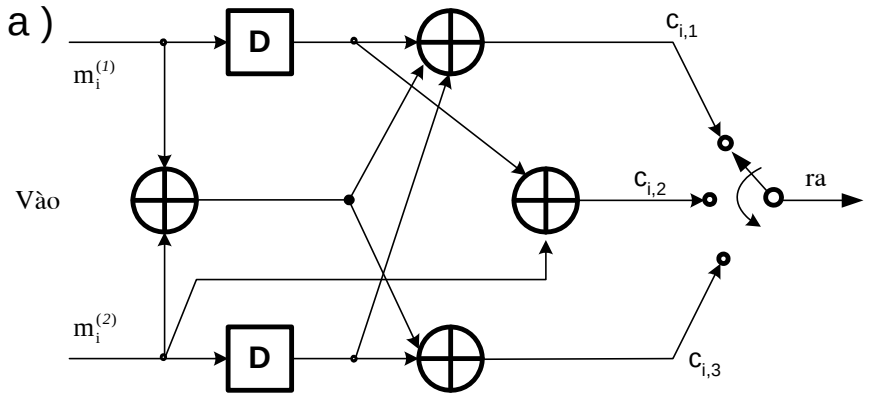
$$\begin{aligned} c_{i,1} &= m_i^{(1)} + m_i^{(2)} + m_{i-1}^{(1)} + m_{i-1}^{(2)} \\ c_{i,2} &= m_{i-1}^{(1)} + m_i^{(2)} \\ c_{i,3} &= m_i^{(1)} + m_i^{(2)} + m_{i-1}^{(2)} \end{aligned} \qquad (3.48)$$

trong đó: $m_{i-p}^{(q)}$ là các bit bản tin vào bộ nhớ p từ các đầu vào q=1, 2 ở thời điểm (i-p), p là số thứ tự nhớ (p =1, 2), $c_{i,j}$ là các bit đầu ra của các bộ cộng modulo-2 thứ j (j =1, 2, 3) ở thời điểm i, dấu cộng biểu thị cộng Modul-2.

✓***Biểu diễn phương trình vào ra theo toán tử trễ x.***

Nếu coi x là toán tử trễ thì các phương trình (3.48) được viết dưới dạng

$$\begin{aligned} c_{i,1} &= \underbrace{m_i^{(1)}(1+x)}_{m_i^{(1)}+m_{i-1}^{(1)}} + \underbrace{m_i^{(2)}(1+x)}_{m_i^{(2)}+m_{i-1}^{(2)}} \\ c_{i,2} &= \underbrace{m_i^{(1)}.x}_{m_{i-1}^{(1)}} + m_i^{(2)} \\ c_{i,3} &= m_i^{(1)} + \underbrace{m_i^{(2)}(1+x)}_{\substack{m_i^{(2)}+m_{i-1}^{(2)}=m_i^{(2)}+m_i^{(2)}.x \\ =m_i^{(2)}(1+x)}} \end{aligned} \qquad (3.49a)$$



M = 1, k = 2, n =3, K = 2

Hình 3.10 a. Bộ lập mã xoắn

✓***Xác định kích thước và giá trị cụ thể cho các phần tử của ma trận tạo mã G***

Xác xác định kích thước và các phương trình vào/ra theo các phần tử của G.

Từ các phương trình tổng quát và các thông số đặc trưng cho sơ đồ này (k=2, n=3,M=1,K=2), nhận được kích thước và các phần tử của ma trận tạo mã G tương ứng cũng như các phương trình vào ra được viết theo các phần tử của ma trận G như sau

$$\begin{aligned} c_i &= \left(m_i^{(1)}, m_i^{(2)}\right)_{1\times 2} \times \begin{bmatrix} g_{p,1}^{(1)} & g_{p,2}^{(1)} & g_{p,3}^{(1)} \\ g_{p,1}^{(2)} & g_{p,2}^{(2)} & g_{p,3}^{(2)} \end{bmatrix}_{2\times 3} \\ &= \left(c_{i,1}, c_{i,2}, c_{i,3}\right)_{1\times 3} \end{aligned}$$

$$\begin{aligned} c_{i,1} &= m_i^{(1)}.g_{p,1}^{(1)} + m_i^{(2)}.g_{p,1}^{(2)} \\ c_{i,2} &= m_i^{(1)}.g_{p,2}^{(1)} + m_i^{(2)}.g_{p,2}^{(2)} \\ c_{i,3} &= m_i^{(1)}.g_{p,3}^{(1)} + m_i^{(2)}.g_{p,3}^{(2)} \end{aligned} \qquad (3.49b)$$

Xác định giá trị cụ thể cho các phần tử của ma trận tạo mã G: Bằng cách so sánh các phương trình vào ra của sơ đồ theo các phần tử của ma trận tạo mã G với các phương trình vào ra theo toán tử trễ x rút ra được giá trị cụ thể cho các phần tử của ma trận tạo mã cần tìm.

So sánh ptr(3.49) với pt(3.49b) ta được

$$c_{i,1} = \underbrace{m_i^{(1)}(1+x)}_{m_i^{(1)}+m_{i-1}^{(1)}} + \underbrace{m_i^{(2)}(1+x)}_{m_i^{(2)}+m_{i-1}^{(2)}} \quad \Leftrightarrow c_{i,1} = m_i^{(1)}.g_{p,1}^{(1)} + m_i^{(2)}.g_{p,1}^{(2)}$$

$$c_{i,2} = \underbrace{m_i^{(1)}.x}_{m_{i-1}^{(1)}} + m_i^{(2)} \quad \Leftrightarrow c_{i,2} = m_i^{(1)}.g_{p,2}^{(1)} + m_i^{(2)}.g_{p,2}^{(2)}$$

$$c_{i,3} = m_i^{(1)} + \underbrace{m_i^{(2)}(1+x)}_{\substack{m_i^{(2)}+m_{i-1}^{(2)}=m_i^{(2)}+m_i^{(2)}.x \\ =m_i^{(2)}(1+x)}} \quad \Leftrightarrow c_{i,3} = m_i^{(1)}.g_{p,3}^{(1)} + m_i^{(2)}.g_{p,3}^{(2)}$$

$$G = \begin{bmatrix} g_{p,1}^{(1)} & g_{p,2}^{(1)} & g_{p,3}^{(1)} \\ g_{p,1}^{(2)} & g_{p,2}^{(2)} & g_{p,3}^{(2)} \end{bmatrix}_{2\times 3} = \begin{bmatrix} 1+x & x & 1 \\ 1+x & 1 & 1+x \end{bmatrix}_{2\times 3} \qquad (3.50)$$

<u>Bước 2</u>: Ma trận kiểm tra chẵn lẻ H

Do mối quan hệ giữa ma trận kiểm tra chẵn lẻ H và ma trận tạo mã G thông qua ma trận P như sau. Biến đổi G vào dạng hệ thống nhận được

$$G = \begin{bmatrix} 1 & 0: & (1+x)^{-1} + x^2(1+x)^{-2} \\ 0 & 1: & x(1+x)^{-1} \end{bmatrix}_{2\times 3} = \left[I_{2\times 2} : P_{2\times 1} \right]_{2\times 3} \qquad (3.51)$$

Ma trận kiểm tra chẵn lẻ có dạng:

$$H = \left[\underbrace{P_{2\times 1}^T}_{(n-k=1)\times k} : \underbrace{I_1}_{(n-k)=1} \right]_{1\times 3} = \left[\underbrace{(1+x)^{-1} + x^2(1+x)^{-2} \quad x(1+x)^{-1}}_{P_{2\times 1}^T} : \underbrace{1}_{I_1} \right]_{1\times 3}$$

Loại bỏ mũ âm bằng cách nhân ma trận trên với $(1+x)^2$ ta được:

$$H = \left[1 + x + x^2, \quad x(1+x), \quad (1+x)^2 \right] \qquad (3.52)$$

<u>Bước 3</u>: Xác định dạng hồi tiếp của sơ đồ lập mã

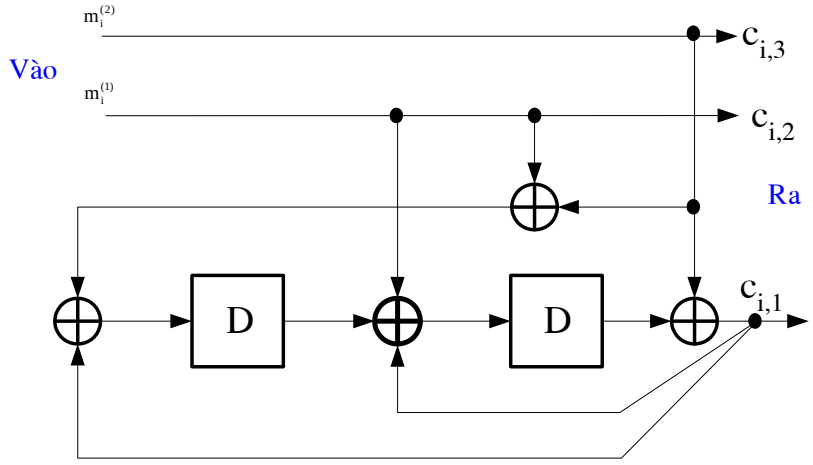
Bằng cách dùng phương trình $c_i H^T = 0$, tìm được phương trình thiết kế sơ đồ hồi tiếp của bộ lập mã.

Từ phương trình $c_i H^T = 0$, nhận được phương trình:

$$c_{i,1} + c_{i-1,1} + c_{i-2,1} + c_{i-1,2} + c_{i-2,2} + c_{i,3} + c_{i-2,3} = 0 \qquad (3.53)$$

trong đó $c_{i-p,j}$ là bit đầu ra j tại thời điểm i-p

Từ phương trình này ta có thể thiết kế dạng hồi tiếp của bộ tạo mã ở hình 3.11



Hình 3.11. Dạng hồi tiếp của bộ lập mã xoắn

Cả sơ đồ truyền thẳng và hồi tiếp đều được sử dụng để trình bày mã xoắn.

3.4.4. Chuỗi tạo mã và đa thức tạo mã

❖ Mục đích

Trình bầy hoạt động của bộ lập mã xoắn dựa vào chuỗi tạo mã (đa thức tạo mã).

❖ Cách tiếp cận

Để tìm được các bit đầu ra của bộ lập mã cho trường hợp đơn giản (k=1) nghĩa là mỗi lần chỉ đưa 1 bit tin vào bộ lập mã, sau đó tổng quát hoá cho trường hợp đưa k bit đồng thời vào bộ lập mã. Vì vậy, trước hết xác định chuỗi tạo mã (đa thức tạo mã), sau đó tìm chuỗi bit đầu ra (đa thức đầu ra) bộ lập mã dựa vào chuỗi tạo mã (đa thức tạo mã).

✓ Chuỗi tạo mã

Để tính toán đáp ứng ở đầu ra của bộ cộng Modul-2 thứ j lên đầu vào q có thể sử dụng chuỗi tạo mã sau:

$$g_j^{(q)} = \left(g_{0,j}^{(q)}, g_{1,j}^{(q)}, ..., g_{p,j}^{(q)}, ..., g_{M,j}^{(q)}\right) \qquad \textbf{(3.54)}$$

trong đó $g_{p,j}^{(q)}$ là đáp ứng của đầu ra bộ nhớ thứ p lên đầu vào q của bộ cộng j bằng 0 nếu đầu ra của bộ nhớ không nối đến bộ cộng và bằng 1 nếu đầu này được nối đến bộ cộng (đầu vào bộ nhớ ***đầu*** tương ứng với p=0) nghĩa là

$$g_{p,j}^{(q)} = \begin{cases} 0, & \text{nếu dầu ra bộ nhớ p không nối đến bộ cộng Modul2 thứ j} \\ 1, & \text{nếu dầu ra bộ nhớ p được nối đến bộ cộng Modul2 thứ j} \end{cases}$$

✓ Đa thức tạo mã

Có thể biểu diễn đáp ứng xung ở đầu ra của bộ cộng modul-2 thứ j ở dạng đa thức tạo mã cho đầu vào q của bộ lập mã như sau:

$$g_j^{(q)}(x) = g_{0,j}^{(q)} + g_{1,j}^{(q)}x + ... + g_{p,j}^{(q)}x^p + ... + g_{M,j}^{(q)}x^M = \sum_{p=0}^{M} g_{p,j}^{(q)}x^p \quad \textbf{(3.55)}$$

trong đó hệ số $g_{p,j}^{(q)} = 0/1$ giống như đối với chuỗi tạo mã.

❖ Sử dụng chuỗi tạo mã để trình bày hoạt động bộ lập mã

➢ ***Trường hợp mỗi lần đưa 1 bit tin vào bộ lập mã (k=1, r = 1/n):***

Hoạt động của bộ lập mã xoắn với k=1, r = 1/n trên cơ sở chuỗi tạo mã được cho bởi phương trình (3.54). Giả sử luồng bit tin đầu vào được chia thành các đoạn có độ dài L, được phân cách nhau bởi các đoạn đuôi K-1 bit "0" để các đoạn này không ảnh hưởng lên nhau, có thể biểu diễn chuỗi bit ở dạng sau:

$$m = \left(\overset{\Leftarrow \text{Hướng truyền}}{m_0, ..., m_\ell, ..., m_{L-1}} \right) \quad (3.56)$$

trong đó m_ℓ là các bit của khối bản tin L bit, $\ell = 0$ là *bit đầu tiên vào bộ lập mã.*

Vì trong trường hợp này q=1 *nên* có thể viết lại phương trình (3.54) như sau:

$$g_j^{(q)}\Big|_{q=1} = g_j = \left(g_{0,j}, g_{1,j}, ..., g_{p,j}, ..., g_{M,j}\right) \quad (3.57)$$

Ký hiệu chuỗi bit ra của các nhánh cộng modul-2 thứ j là c_j như sau:

$$c_j = \left(c_{0,j}, c_{1,j}, ..., \underbrace{\overset{\Leftarrow \text{Hướng truyền}}{c_{i,j}}}_{\substack{\text{bit ra thứ i} \\ \text{của nhánh thứ j}}}, ..., c_{L+M-1,j} \right) \quad \textbf{(3.58)}$$

trong đó j =1, 2, ….n; $c_{i,j}$ là bit ra i của nhánh j, i = 0,1,...,L+M-1 với i=0 tương ứng với đầu vào của bộ lập mã được nối đến nhánh cộng modul-2 và M là số bộ nhớ của bộ lập mã. Vậy trong mọi trường hợp đều nhận được số bit ở đầu ra nhánh cộng bằng **(L+K-1)** bit.

Các phần tử $c_{i,j}$ của chuỗi c_j này được xác định theo công thức ***tích chập*** sau:

$$c_{i,j} = \sum_{p=0}^{M} g_{p,j}.m_{i-p}, \qquad \begin{matrix} i = 0,1,2,..., L+M-1 \\ j = 1,2,..., n \end{matrix} \quad \textbf{(3.59)}$$

trong đó m_{i-p}=0 đối với p > i và (p = 0, i = 0) tương ứng với đầu vào của bộ lập mã được nối đến nhánh cộng modul-2. Chuỗi đầu ra của bộ lập mã là ghép xen của các chuỗi c_j.

➢ ***Trường hợp tổng quát mỗi lần đưa k bit tin đồng thời vào bộ lập mã (r = k/n):***

Nếu ký hiệu k bit đồng thời đưa vào bộ nhớ p ở thời điểm i ở dạng vector sau:

$$m_{i-p} = \left(m_{i-p}^{(1)}, m_{i-p}^{(2)}, ..., m_{i-p}^{(q)}, ..., m_{i-p}^{(k)}\right) \quad (3.60)$$

trong đó $m_{i-p}^{(q)}$ là bit thứ q vào bộ nhớ p tại thời điểm xét i.

Thì các đầu ra tương ứng ở thời điểm i được xác định như sau:

$$c_i = \sum_{p=0}^{M} m_{i-p} G_p \quad (3.61)$$

trong đó $\mathbf{G_p}$ là vector hay *ma trận kết nối* biểu thị sự kết nối giữa các đầu ra của *bộ nhớ thứ p* (p = 0 tương ứng với đầu vào bộ nhớ thứ nhất) với **các** bộ cộng modul-2 ở đầu ra của bộ lập mã được xác định như sau:

$$G_p = \begin{bmatrix} g_{p,1}^{(1)}, & g_{p,2}^{(1)} & ,...., & g_{p,j}^{(1)} & ,..., & g_{p,n}^{(1)} \\ g_{p,1}^{(2)}, & g_{p,2}^{(2)} & ,..., & g_{p,j}^{(2)} & ,..., & g_{p,n}^{(2)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ g_{p,1}^{(q)}, & g_{p,2}^{(q)} & ,..., & g_{p,j}^{(q)} & ,..., & g_{p,n}^{(q)} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ g_{p,1}^{(k)}, & g_{p,2}^{(k)} & ,..., & g_{p,j}^{(k)} & ,..., & g_{p,n}^{(k)} \end{bmatrix} \quad (3.62)$$

trong đó các phần tử $g_{p,j}^{(q)}$ của ma trận tương ứng với đáp ứng đầu ra nhánh cộng j của bộ nhớ p lên bit thứ q trong số k bit vào đồng thời.

❖ Sử dụng đa thức tạo mã để trình bày hoạt động bộ lập mã

Để sử dụng phương trình (3.55) vào việc trình bày hoạt động bộ lập mã xoắn, cần phải viết các đa thức bản tin, đa thức tạo mã, đa thức chuỗi bit ra của bộ lập mã cho hai trường hợp dưới đây.

➢ ***Trường hợp mỗi lần đưa 1 bit tin vào bộ lập mã (k=1, r = 1/n):***

✓ *Đa thức bản tin:* Giả sử chuỗi bit tin đầu vào được chia thành các đoạn dài L bit với đuôi là K-1 bit "0" như đã xét ở trên, từ ptr.(3.56) có thể viết *chuỗi bản tin* này vào dạng *đa thức bản tin* như sau:

$$\begin{aligned} m(x) &= m_0 + m_1 x + ... + m_\ell x^\ell + ... + m_{L-1} x^{L-1} \\ &= \sum_{\ell=0}^{L-1} m_\ell x^\ell \end{aligned} \quad (3.63)$$

✓ *Đa thức tạo mã* $g_j^{(q)}(x)\big|_{q=1} = g_j(x)$*:* Đáp ứng của nhánh cộng j (đa thức tạo mã ở nhánh thứ j $\Leftrightarrow$ $g_j^{(q)}(x)\big|_{q=1} = g_j(x)$) lên chuỗi bit vào trong trường hợp này có thể biểu diễn ở dạng đa thức tạo mã như sau:

$$\begin{aligned} g_j(x) &= g_{0,j} + g_{1,j} x + ... + g_{p,j} x^p + ... + g_{M,j} x^M \\ &= \sum_{p=0}^{M} g_{p,j} x^p \end{aligned} \quad (3.64)$$

trong đó $g_{p,j} = 0/1$ tương ứng với phần tử nhớ p không nối hoặc có nối vào bộ cộng thứ j.

✓ *Đa thức chuỗi bit mã đầu ra thứ j bộ lập mã:* Đa thức của chuỗi bit ra nhánh j như sau:

$$\begin{aligned} c_j(x) &= m(x).g_j(x) \\ &= c_{0,j} + c_{1,j}x + ... + c_{i,j}x^i + ... + c_{L+M-1,j}x^{L+M-1} \\ &= \sum_{i=0}^{L+M-1} c_{i,j}x^i \end{aligned} \quad (3.65)$$

trong đó hệ số $c_{i,j} = 0/1$ tương ứng với bit i ở đầu ra j.

➢ ***Trường hợp tổng quát mỗi lần đưa đồng thời k bit tin vào bộ lập mã (r = k/n):***

✓ *Đa thức bản tin:* ***Nếu*** mỗi lần đồng thời đưa k bit vào bộ lập mã, ***thì*** có thể biểu diễn chuỗi bản tin cho đầu vào thứ q (q=1,2,..,k) có độ dài L đầu ở dạng đa thức bản tin sau:

$$\begin{aligned} \mathbf{m}^{(q)}(x) &= m_0^{(q)} + m_1^{(q)}x + ... + m_\ell^{(q)}x^\ell + ... + m_{L-1}^{(q)}x^{L-1} \\ &= \sum_{\ell=0}^{L-1} m_\ell^{(q)}x^\ell \end{aligned} \quad (3.66)$$

✓ *Đa thức tạo mã*: được cho ở phương trình (3.55)

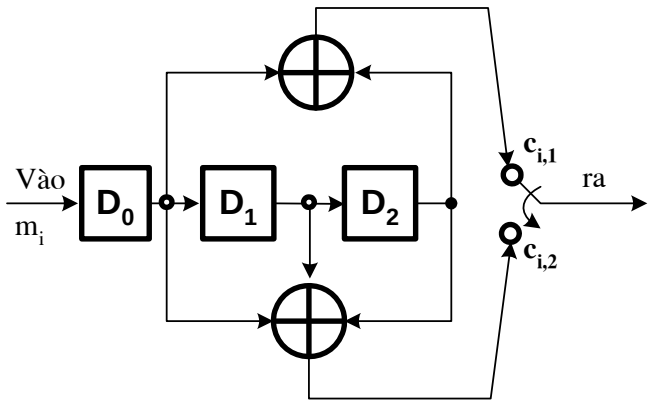
✓ *Đa thức chuỗi bit mã đầu ra thứ j bộ lập mã:* Đa thức chuỗi bit mã đầu ra nhánh j khi này được xác định như sau:

$$\mathbf{c}_j(x) = \sum_{q=1}^{k} \mathbf{m}^{(q)}(x).\mathbf{g}_j^{(q)}(x) \quad (3.67)$$

trong đó j = 1,2,..., n, $\mathbf{g}_j^{(q)}(x)$ là đa thức tạo mã *cho chuỗi vào đầu q*, nhánh ra j.

Chuỗi bit đầu ra của bộ lập mã là ghép xen của n các chuỗi nhánh.

Thí dụ 3.6: Tìm chuỗi bit ở đầu ra của bộ tạo mã ở hình 3.10d



M = 3, k = 1, n = 2, K = 3

khi chuỗi bit đầu vào có dạng:

$$\mathbf{m} = (m_0, m_1, m_2, m_3, m_4) = (10011)$$

trong đó m_0 là bit đầu tiên vào bộ lập mã.

Trong thí dụ này L = 5, M=3, K= 3, L+M-1 = 7, độn loại hai là (K-1) bit "0" để cho các đoạn bản tin không ảnh hưởng lên nhau, r = 1/2. Vì vậy, sau khi độn 2 bit "0", thì bản tin khi này có độ dài là 7 bit được xác định như sau:

$$m=(m_0, m_1, m_2, m_3, m_4, m_5, m_6) = (1001\mathbf{00})$$

❖ Chuỗi tạo mã

➢ Xác định chuỗi tạo mã

Từ các ptr.(3.54) có thể viết:

$$g_j^{(q)} = \left(g_{0,j}^{(q)}, g_{1,j}^{(q)}, ..., g_{p,j}^{(q)}, ..., g_{M,j}^{(q)}\right)$$

trong đó $g_{p,j}^{(q)}$ là đáp ứng của đầu ra bộ nhớ thứ p lên đầu vào q của bộ cộng j được xác định bởi $g_{p,j}^{(q)} = \begin{cases} 0, & \text{nếu dầu ra bộ nhớ p không nối dến bộ cộng Modul2 thứ j} \\ 1, & \text{nếu dầu ra bộ nhớ p dược nối dến bộ cộng Modul2 thứ j} \end{cases}$

do k=1 $\Rightarrow$ chỉ có một đầu vào q=1.

đầu vào bộ lập mã không được nối đến bộ cộng Modul2 $\Rightarrow$ p=1,2,..,M

vì vậy áp dụng ptr(3.54) vào trường hợp này nhận được

$$g_j^{(q)} = \left(g_{0,j}^{(q)}, g_{1,j}^{(q)}, ..., g_{p,j}^{(q)}, ..., g_{M,j}^{(q)}\right)\Big|_{q=1, M=3, n=2}$$

$$\Rightarrow g_j = \left(g_{1,j}, g_{2,j}, g_{3,j},\right)\Big|_{n=2}$$

$$\Rightarrow \begin{cases} g_1 = \left(g_{1,1}, g_{2,1}, g_{3,1},\right) = \underbrace{(1,0,1)}_{\text{Nhánh trên}} \\ g_1 = \left(g_{1,2}, g_{2,2}, g_{3,2},\right) = \underbrace{(1,1,1)}_{\text{Nhánh dưới}} \end{cases}$$

➢ Xác định các chuỗi bit ra của các nhánh cộng modul2 thứ j và chuỗi bit ra

✓ *Chuỗi bit ra của các nhánh cộng Molul2 thứ j được xác định theo ptr(3.58)*

$$c_j = \left(c_{0,j}, c_{1,j}, ..., c_{i,j}, ..., \underbrace{c_{L+M-1,j}}_{=7}\right)$$

$$= \underbrace{\left(c_{1,j}, ..., c_{i,j}, ..., c_{7,j}\right)}_{\substack{\text{Do dầu vào không dược nối dến} \\ \text{bộ cộng Modul2 nên i=1,2,..,7}}}$$

trong đó $c_{i,j}$ là bit ra thứ i của nhánh thứ j và được xác định bởi

$$c_{i,j} = \sum_{p=1}^{M} g_{p,j}, m_{i-p}, \qquad i = 1,2,...,7;$$

$$j = 1,2$$

$$p = 1,2,..., M$$

$$m_{i-p} = 0 \text{ khi } i - p < 0 \Leftrightarrow i < p$$

+ Đối với nhánh trên (j=1)

$$c_1 = \left(c_{1,1}, c_{2,1}, c_{3,1}, c_{4,1}, c_{5,1}, c_{6,1}, c_{7,1}\right)$$

+ Đối với nhánh dưới (j=2)

$$c_1 = \left(c_{1,2}, c_{2,2}, c_{3,2}, c_{4,2}, c_{5,2}, c_{6,2}, c_{7,2}\right)$$

✓*Chuỗi bit đầu ra của bộ lập mã c là ghép xen của các chuỗi bit của các n nhánh cộng modul2, trường hợp này là*

$$c = \left(\underbrace{c_{1,1}c_{1,2}}_{1}, \underbrace{c_{2,1}, c_{2,2}}_{2}, \underbrace{c_{3,1}, c_{3,2}}_{3}, ..., \underbrace{c_{7,1}, c_{7,2}}_{7} \right)$$

➢**Tính các giá trị** $c_{i,j}$**.**

Đối với nhánh trên (j=1), ta được

Triển khai ptr.(3.59) cho nhánh trên trong trường hợp này (lưu ý p bắt đầu từ 1 vì đầu vào của bộ nhớ thứ nhất không được nối đến nhánh cộng) nhận được:

$$c_{1,1} = \sum_{p=1}^{3} g_{p,1} m_{1-p} = g_{1,1} m_{1-1} + g_{2,1} \underbrace{m_{1-2}}_{=0} + g_{3,1} \underbrace{m_{1-3}}_{=0} = 1.1 + 0.0 + 1.0 = 1$$

$$c_{2,1} = \sum_{p=1}^{3} g_{p,1} m_{2-p} = g_{1,1} m_{2-1} + g_{2,1} m_{2-2} + g_{3,1} \underbrace{m_{2-3}}_{=0} = 1.0 + 0.1 + 1.0 = 0$$

$$c_{3,1} = \sum_{p=1}^{3} g_{p,1} m_{3-p} = g_{1,1} m_{3-1} + g_{2,1} m_{3-2} + g_{3,1} m_{3-3} = 1.0 + 0.0 + 1.1 = 1$$

$$c_{4,1} = \sum_{p=1}^{3} g_{p,1} m_{4-p} = g_{1,1} m_{4-1} + g_{2,1} m_{4-2} + g_{3,1} m_{4-3} = 1.1 + 0.0 + 1.0 = 1$$

$$c_{5,1} = \sum_{p=1}^{3} g_{p,1} m_{5-p} = g_{1,1} m_{5-1} + g_{2,1} m_{5-2} + g_{3,1} m_{5-3} = 1\times1 + 0\times1 + 1\times0 = 1$$

$$c_{6,1} = \sum_{p=1}^{3} g_{p,1} m_{6-p} = g_{1,1} \underbrace{m_{6-1}}_{\text{bit chèn}} + g_{2,1} m_{6-2} + g_{3,1} m_{6-3} = 1\times0 + 0\times1 + 1\times1 = 1$$

$$c_{7,1} = \sum_{p=1}^{3} g_{p,1} m_{7-p} = g_{1,1} \underbrace{m_{7-1}}_{\text{bit chèn}} + g_{2,1} \underbrace{m_{7-2}}_{\text{bit chèn}} + g_{3,1} m_{7-3} = 1\times0 + 0\times0 + 1\times1 = 1$$

Vậy chuỗi bit ở đầu ra của nhánh trên (j=1) là:

c_1 = (1011111)

Tương tự đối với nhánh dưới j=2, ta được

$$c_{1,2} = \sum_{p=1}^{3} g_{p,1} m_{1-p} = g_{1,2} m_{1-1} + g_{2,2} \underbrace{m_{1-2}}_{\substack{=0 \text{ do} \\ i-p<0}} + g_{3,2} \underbrace{m_{1-3}}_{\substack{=0 \text{ do} \\ i-p<0}} = 1\times1 + 1\times0 + 1\times0 = 1$$

$$c_{2,2} = \sum_{p=1}^{3} g_{p,1} m_{2-p} = g_{1,2} m_{2-1} + g_{2,2} m_{2-2} + g_{3,2} \underbrace{m_{2-3}}_{\substack{=0 \text{ do} \\ i-p<0}} = 1\times0 + 1\times1 + 1\times0 = 1$$

$c_{3,2} = g_{1,2}\, m_2 + g_{2,2}\, m_1 + g_{3,2}\, m_0 = 1\times0 + 1\times0 + 1\times1 = 1$

$c_{4,2} = g_{1,2}\, m_3 + g_{2,2}\, m_2 + g_{3,2}\, m_1 = 1\times1 + 1\times0 + 1\times0 = 1$

$c_{5,2} = g_{1,2}\, m_4 + g_{2,2}\, m_3 + g_{3,2}\, m_2 = 1\times1 + 1\times1 + 1\times0 = 0$

$c_{6,2} = g_{1,2}\, m_5 + g_{2,2}\, m_4 + g_{3,2}\, m_3 = 1\times0 + 1\times1 + 1\times1 = 0$

$c_{7,2} = g_{1,2}\, m_6 + g_{2,2}\, m_5 + g_{3,1}\, m_4 = 1\times0 + 1\times0 + 1\times1 = 1$

Vậy chuỗi bit ở đầu ra của nhánh dưới sẽ là:

c_2 = (1111001)

Chuỗi bit ở đầu ra của bộ lập mã là ghép chung của hai chuỗi c_1, c_2 như sau:

c = (11 01 11 11 10 10 11)

có 14 bit.

❖ Đa thức tạo mã:

✓ *Đa thức bản tin:* Từ ptr (3.63) ta có

$$m(x) = \sum_{\ell=0}^{L-1} m_\ell x^\ell = m_0 + m_1 x + ... + m_\ell x^\ell + ... + m_{L-1} x^{L-1}$$

với m = (10011)

$\Rightarrow m(x) = 1 + x^3 + x^4$

✓ *Đa thức tạo mã* $g_j^{(q)}(x)\big|_{q=1} = g_j(x)$ *:* Từ ptr (3.64) ta có

$$g_j(x) = g_{0,j} + g_{1,j} x + ... + g_{p,j} x^p + ... + g_{M,j} x^M = \sum_{p=0}^{M} g_{p,j} x^p$$

trong đó $g_{p,j} = 0/1$ tương ứng với phần tử nhớ p không nối hoặc có nối vào bộ cộng thứ j và cấu trúc của bộ lập mã. Do đầu vào không được nối đến bộ cộng mdul2 nên p=1,2,..,M; j=1,2 nên nhận được

$g_1(x) = x + x^3$

$g_2(x) = x + x^2 + x^3$

✓ *Đa thức chuỗi bit mã đầu ra thứ j bộ lập mã như sau:*

$$c_j(x) = m(x).g_j(x) = \underbrace{c_{0,j}}_{=0} + c_{1,j} x + ... + c_{i,j} x^i + ... + c_{L+M-1,j} x^{L+M-1} = \sum_{i=1}^{L+M-1} c_{i,j} x^i \qquad (3.65)$$

trong đó hệ số $c_{i,j} = 0/1$ tương ứng với bit i ở đầu ra j.

$c_1(x) = (1 + x^3 + x^4)(x + x^3) = x + x^3 + x^4 + x^5 + x^6 + x^7$

hay: $c_1 = (c_{1,1}, c_{2,1}, c_{3,1}, c_{4,1}, c_{5,1}, c_{6,1}, c_{7,1}) = (1\ 0\ 1\ 1\ 1\ 1\ 1)$

$c_2(x) = (1 + x^3 + x^4)(x + x^2 + x^3) = x + x^2 + x^3 + x^4 + x^7$

hay: $c_2 = (c_{1,2}, c_{2,2}, c_{3,2}, c_{4,2}, c_{5,2}, c_{6,2}, c_{7,2}) = (1\ 1\ 1\ 1\ 0\ 0\ 1)$

Cuối cùng được chuỗi đầu ra là ghép xen của hai chuỗi trên như sau:

c = (11 01 11 11 1010 11)

Như vậy chuỗi bản tin vào 5 bit cộng thêm hai bit duôi bằng "0" để khởi đầu hai tầng sau của thanh ghi dịch vào trạng thái "0" để chúng không ảnh hưởng lên chuỗi bit tiếp theo sẽ được 7 bit. Do vậy tỷ lệ mã trong trường hợp này là r = 1/2.

Trong thực tế chuỗi hay đa thức tạo mã g_j xác định độ trễ của bit vào i khi đưa lên nhánh cộng j, nên có thể dễ dàng xác định chuỗi đầu ra của bộ lập mã bằng cách lập bảng như sau (bảng 3.9).

Bảng 3.9. Thí dụ về tính toán mã xoắn cho bộ lập mã 3.10d.

Chuỗi bản tin m			1	0	0	1	1				
Bổ xung hai bit đuôi m'	0	0	1	0	0	1	1	0	0		
1/ Trễ một tầng m'.x		0	0	1	0	0	1	1	0	0	
2/ Trễ hai tầng m'.x2			0	0	1	0	0	1	1	0	0
3/ Trễ ba tầng m'.x3				0	0	1	0	0	1	1	0 0
C1 = 1/ + 3/				1	0	1	1	1	1	1	
C2 = 1/ + 2/ + 3/				1	1	1	1	0	0	1	
C				11	01	11	11	10	10	11	

Trong bảng 3.9 sau khi bổ xung hai bit đuôi vào chuỗi bit vào của bộ lập mã, thực hiện trễ một bit (dịch toàn bộ chuỗi sang phải một bit), trễ hai bit (dịch chuỗi sang phải hai bit) và trễ ba bit (dịch toàn bộ chuỗi sang phải ba bit). Sau đó cộng 1/ với 3/ (tương ứng với $g_{1,1}$ =1, $g_{2,1}$ = 0 và $g_{3,1}$ = 1) để được $\mathbf{c_1}$ và 1/ với 2/ và với 3/ (tương ứng với $g_{1,2}$ = 1, $g_{2,2}$ = 1, $g_{3,2}$ =1) để được $\mathbf{c_2}$.

3.4.5. Biểu đồ trạng thái

❖ Mục đích

Xây dựng biểu đồ trạng thái và khảo sát hoạt động bộ lập mã bằng cách sử dụng biểu đồ trạng thái đặc trưng cho sơ đồ bộ lập mã được xét.

❖ Cấu tạo

Biểu đồ trạng thái gồm các nut trạng thái được nối với nhau bởi các nhánh cùng với các tổ hợp bit ra khi chuyển từ nút trạng thái này sang nút trạng thái khác. ***Để*** sử dụng biểu đồ trạng thái vào việc trình bầy hoạt động bộ lập mã mã xoắn, nghĩa là mỗi khi đưa đồng thời k bit tin vào bộ lập mã, ta xác định được n bit ra tương ứng thông qua biểu đồ trạng thái đặc trưng cho sơ đồ lập mã đó. ***Thì***, biểu đồ trạng thái phải: ***(1)*** chứa tất cả các trạng thái có thể có của bộ lập mã ***(2)*** thể hiện được sự chuyển dịch giữa các trạng thái mỗi khi đưa đồng thời k bit tin vào bộ lập mã, ***(3)*** quan hệ các bit vào/ra của bộ lập mã.

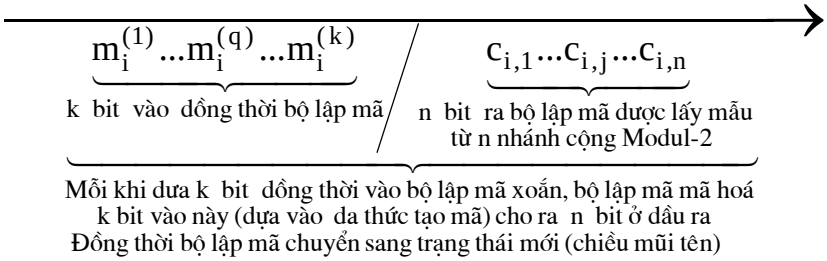
Từ các thông số đặc trưng của sơ đồ lập mã ta xác định được

Trang thái của môt bô lập mã xoắn: là tổ hợp (K-1) các bit vào đồng thời trước đó (số bit trạng thái là (K-1)k) mã vẫn ảnh hưởng lên sự hình thành các bit ra của bộ lập mã ở thời điểm xét.

Số nut trang thái có thể có: ***Từ*** các thông số đặc trưng của bộ lập mã như: số bit dịch vào bộ lập mã tại cùng một thời điểm k; độ dài hạn chế thể hiện số lần dịch cực đại của một nhóm k bit bản tin vào mà nhóm k bit này vẫn còn gây ảnh hưởng tới đầu ra bộ lập mã K; số bộ nhớ (mỗi bộ nhớ có k phần tử nhớ) của bộ lập mã M. Ta xác định được số trạng thái có thể có của bộ lập mã là

$$2^{(K-1)k} = \begin{cases} 2^{(M-1)k}, \text{nếu dầu vào bộ lập mã không nối với bộ cộng Modul} - 2 \\ 2^{M.k}, \quad \text{nếu dầu vào bộ lập mã nối với bộ cộng Modul} - 2 \end{cases}$$

Các nhánh nối giữa các trang thái: Thể hiện ***(1)*** chiều chuyển dịch giữa các trạng thái trong biểu đồ trạng thái bằng mũi tên; ***(2)*** điều kiện để chuyển dịch trạng thái thái (giá trị cụ thể của các bit trong k bit đưa vào đồng thời bộ lập mã & cấu trúc cụ thể cho bộ lập mã ⇔ đa thức tạo mã); ***(3)*** các giá trị cụ thể của các bit trong n bit đầu ra. Theo đó, mỗi nhánh được đánh nhãn bởi các bit vào/ra như sau



Việc đánh nhãn này cho thấy n bit đầu ra được xác định bởi k bit vào đồng thời và trạng thái trước đó của bộ lập mã và sự chuyển dịch từ trạng thái cũ sang trạng thái mới của bộ lập mã.

❖ Hoạt động

Do tính chất nhớ của bộ lập mã xoắn, nên quá trình tạo ra n bit đầu ra *không chỉ* phụ thuộc vào k bit tin đầu vào đồng thời *mà còn* phụ thuộc vào (K-1) tập hợp k bit đầu vào trước đó. Hay nói cách khác n bit đầu ra (được lấy mẫu từ các đầu ra của n bộ cộng Modul-2 tương ứng) ***không chỉ*** phụ thuộc vào k bit vào đồng thời ***mà còn*** phụ thuộc vào trạng thái trước đó của bộ lập mã (là một trạng thái thuộc một trong số $2^{(K-1)k}$ trạng thái có thể có của bộ lập mã).

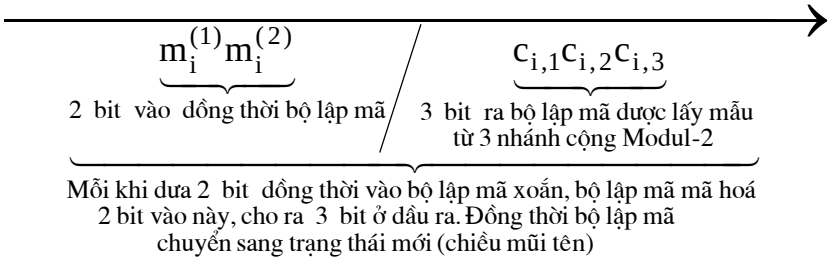
Biết được trạng thái hiện tại cùng với k bit vào đồng thời tiếp theo, dễ dàng xác định được n bit ra tiếp theo của bộ lập mã. Hay nói cách khác n bit ra ($c_{i,1}c_{i,2}\ldots c_{i,j}\ldots c_{i,n}$) tại thời điểm xét được xác định bởi: k bit vào đồng thời tại thời điểm xét và trạng thái trước đó của bộ lập mã.

Ví dụ

Biểu đồ trạng thái cho bộ lập mã ở hình 3.10 a và 3.10d được vẽ ở hình 3.12. Các nút trạng thái ở sơ đồ hình 3.12 a được xác định bởi hai bit đầu vào của Flip-Flop $m_i^{(1)}m_i^{(2)}$ còn ở hình 3.12 b được xác định bằng hai bit ở đầu ra của Flip-flop thứ nhất

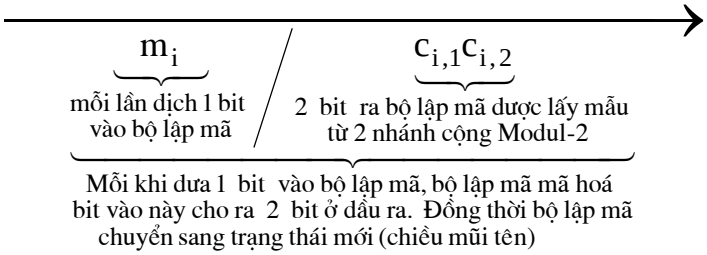
và thứ hai D_0D_1, vì thế chúng đều có bốn trạng thái: a = (00), b= (10), c=(01) và d=(11).

Đối với sơ đồ hình 3.12a: Khi 2 bit mới được đưa vào đồng thời ($m_i^{(1)}m_i^{(2)}$) kết hợp với hai bit vào đồng thời trước đó tạo ra **ba** bit ở đầu ra và trạng thái được chuyển đổi



Việc đánh nhãn này cho thấy 3 bit đầu ra được xác định bởi 2 bit vào đồng thời và trạng thái trước đó của bộ lập mã và sự chuyển dịch từ trạng thái cũ sang trạng thái mới củac bộ lập mã.

Đối với sơ đồ hình 3.12b: Khi một bit mới được đưa vào, hai bit trạng thái (D_0D_1) sẽ kết hợp với bit vào để tạo ra hai bit ra ($c_{i,1}c_{i,2}$) và trạng thái được chuyển đổi. Chuyển đổi trạng thái được thể hiện bằng một nhánh nối giữa trạng thái cũ và mới.



Thông thường, khi k =1 thì nhánh có bit vào là "1" được biểu thị bằng đường không liên tục còn nhánh có bit vào là "0" được biểu thị bằng đường liên tục, vì thế ở mỗi nhánh chỉ có các nhãn biểu thị hai bit ra.

3.4.6. Biểu đồ hình cây

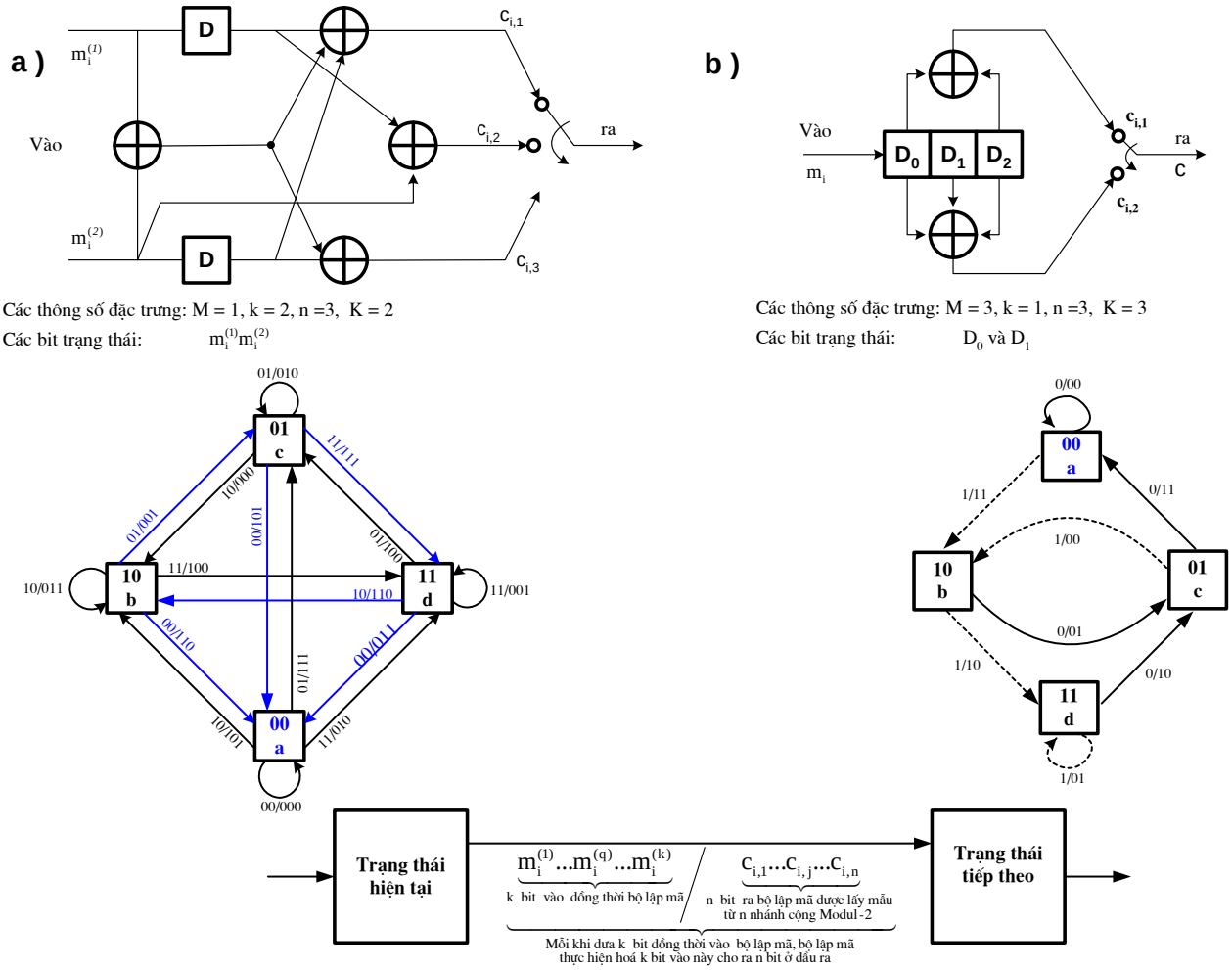
❖ Mục đích

Xây dựng biểu đồ hình cây và trình bầy hoạt động bộ lập mã theo thời gian bằng cách dùng biểu đồ hình cây đặc trưng cho sơ đồ bộ lập mã này.

❖ Cấu tạo

Biểu đồ hình cây gồm các nút trạng thái được nối với nhau bởi các nhánh cùng với các tổ hợp bit ra khi chuyển từ nút trạng thái này sang nút trạng thái khác có dạng hình cây. ***Để*** sử dụng biểu đồ hình cây vào việc trình bầy hoạt động bộ lập mã (nghĩa là mỗi khi đưa đồng thời k bit tin vào bộ lập mã xoắn có thể xác định được n bit ra tương ứng thông qua biểu đồ hình cây đặc trưng cho sơ đồ lập mã đó). ***Thì*****,** biểu đồ hình cây phải: ***(1)*** chứa tất cả các trạng thái có thể có của bộ lập mã; ***(2)*** thể

hiện được sự chuyển dịch giữa các trạng thái mỗi khi đồng thời đưa k bit tin vào bộ lập mã theo thời gian; ***(3)*** thể hiện quan hệ các bit vào/ra của bộ lập mã.



Hình 3.12. Biểu đồ trạng thái cho sơ đồ lập mã ở sơ đồ 3.10a và 3.10d

Đối với sơ đồ bộ lập mã hình 3.10 a: Từ mỗi nut ta được bốn nhánh tương ứng với các cặp bit đầu vào: 00; 01; 10; 11.

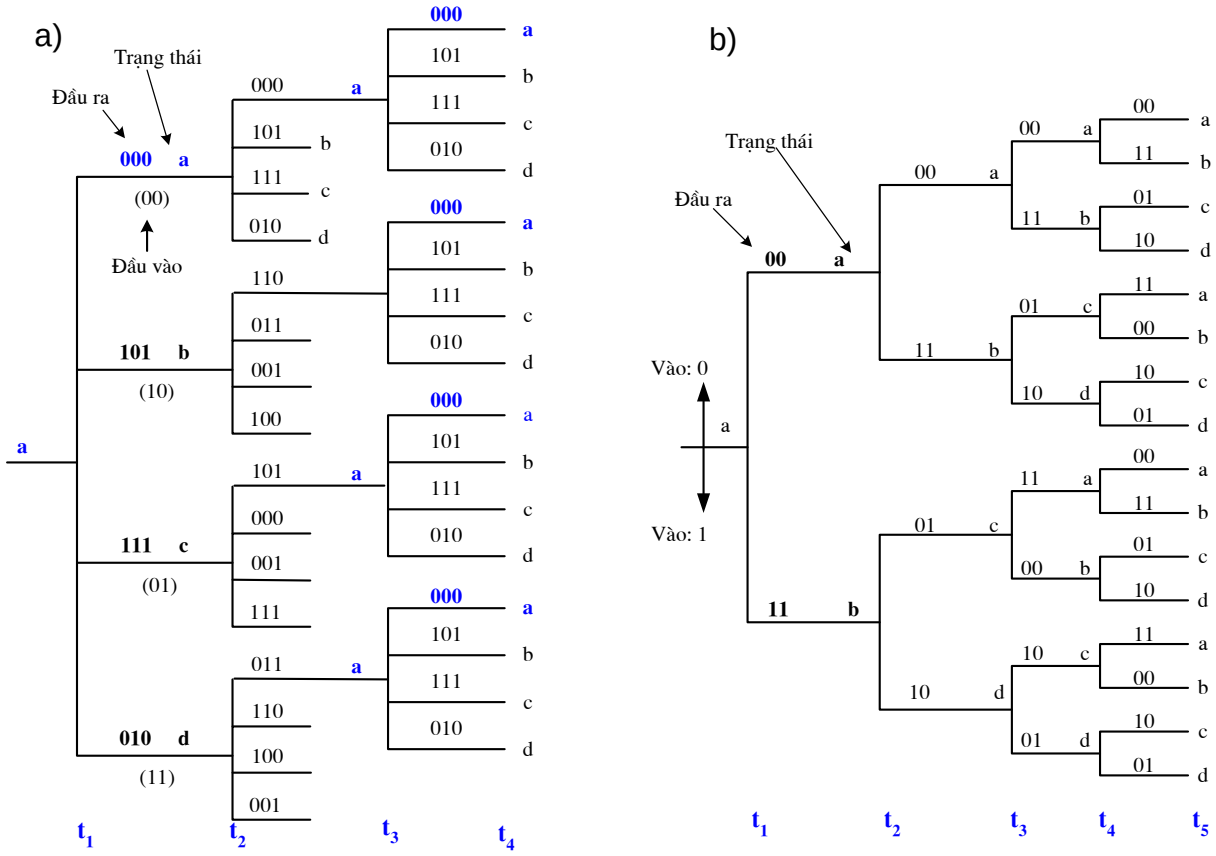
Đối với sơ đồ bộ lập mã hình 3.10 d: Từ mỗi nut ta được hai nhánh: nếu bit đầu vào bằng "0" ta được nhánh trên, ngược lại nếu bit đầu vào bằng "1" ta được nhánh dưới.

Mỗi nhánh cây cũng được đánh nhãn bằng các bit đầu ra tương ứng. Thí dụ về dạng biểu đồ hình cây cho bộ lập mã ở hình 3.10a và 3.10d được cho ở hình 3.13 a và b.

Nghiên cứu kỹ các biểu đồ hình cây nói trên thấy rõ sau nhánh thứ K (K: độ dài hạn chế) các nhánh lặp lại quy luật như trước và nửa trên giống như nửa dưới.

❖ Hoạt động

Do các nhánh cây biểu thị các bit đầu ra của bộ lập mã tương ứng với các bit đầu vào. Vì vậy, nếu cho chuỗi bit vào hoàn toàn xác định được chuỗi bit ra dựa vào quan hệ giữa các bit vào/ra tương ứng trên các nhánh của cây khi đi từ gốc (t_1) lên ngọn (t_i) của biểu đồ cây đặc trưng cho sơ đồ lập mã được xét.



Sau nhánh thứ K =2 các nhánh lại lặp lại quy luật trước và nửa trên của cây giống như nửa dưới.

Sau nhánh thứ K =3 các nhánh lại lặp lại quy luật trước và nửa trên của cây giống như nửa dưới.

Các bit trong (.....) và ở đầu mũi tên biểu thị cho các bit vào bộ lập mã

Hình 3.13. Biểu đồ hình cây cho các bộ lập mã

❖ Nhận xét

Mặc dù biểu đồ cây khắc phục nhược điểm của biểu đồ trạng thái (biểu đồ trạng thái không cho phép theo dõi các chuyển đổi trạng thái của bộ lập mã theo thời gian) bằng cách bổ sung kích thước thời gian cho biểu đồ trạng thái, (cho phép theo dõi hoạt động bộ lập mã theo thời gian) ***nhưng*** khi chuỗi bit vào quá dài thì cây cũng quá **"lớn"**. Để biểu diễn hoạt động của bộ lập mã theo thời gian thuận tiện hơn người ta dùng biểu đồ lưới.

3.4.7. Biểu đồ lưới

❖ Mục đích

Xây dựng biểu đồ lưới và trình bầy hoạt động bộ lập mã dựa vào biểu đồ lưới đặc trưng cho sơ đồ bộ lập mã được xét theo thời gian.

❖ Cấu tạo

Biểu đồ lưới gồm các nút trạng thái được nối với nhau bởi các nhánh cùng với các tổ hợp bit ra khi chuyển từ nút trạng thái này sang nút trạng thái khác tạo thành hình lưới theo thời gian. Để có thể trình bầy hoạt động bộ lập mã xoắn theo biểu đồ lưới này, nghĩa là mỗi khi đưa đồng thời k bit tin vào bộ lập mã ta có thể xác định được n bit ra tương ứng thông qua biểu đồ lưới đặc trưng cho sơ đồ lập mã đó. Muốn vậy, biểu đồ lưới phải: ***(1)*** chứa tất cả các trạng thái có thể có của bộ lập mã; ***(2)*** thể hiện được sự chuyển dịch giữa các trạng thái mỗi khi đồng thời đưa k bit tin vào bộ lập mã; ***(3)*** thể hiện được quan hệ các bit vào/ra của bộ lập mã.

Nut trang thái: Tại mỗi đơn vị thời gian lưới có $2^{(K-1)k} = 2^k.2^{(K-1)}$ nút để biểu thị $2^{(K-1)k}$ trạng thái.

Đánh nhãn:

Đối với sơ đồ lập mã hình 3.10a:

Giả sử trạng thái trước của bộ lập mã là a=00, khi đưa đồng thời hai bit tin $m_i^{(1)}m_i^{(2)}$ vào bộ lập mã, phụ thuộc vào giá trị cụ thể của $m_i^{(1)}m_i^{(2)}$ mà bộ lập mã sẽ chuyển sang một trong 4 trạng thái mới đồng thời cho ra cặp các giá trị cụ thể của $c_{i,1}c_{i,2}c_{i,3}$ như sau:

Nếu là $m_i^{(1)}m_i^{(2)}$ *= 00* $\Rightarrow$ *bộ lập mã chuyển sang trạng thái mới là a=00, đồng thời các bit đầu ra của bộ lập mã khi này là* $c_{i,1}c_{i,2}c_{i,3}$ *=000*

Nếu là $m_i^{(1)}m_i^{(2)}$ *= 10* $\Rightarrow$ *bộ lập mã chuyển sang trạng thái mới là b=10, đồng thời các bit đầu ra của bộ lập mã khi này là* $c_{i,1}c_{i,2}c_{i,3}$ *=101*

Nếu là $m_i^{(1)}m_i^{(2)}$ *= 01* $\Rightarrow$ *bộ lập mã chuyển sang trạng thái mới là c=01; đồng thời các bit đầu ra của bộ lập mã khi này là* $c_{i,1}c_{i,2}c_{i,3}$ *=111*

Nếu là $m_i^{(1)}m_i^{(2)}$ *= 11* $\Rightarrow$ *bộ lập mã chuyển sang trạng thái mới là d=11; đồng thời các bit đầu ra của bộ lập mã khi này là* $c_{i,1}c_{i,2}c_{i,3}$ *=111*

$$\overbrace{a=00}^{\text{Trạng thái trước}} : \underbrace{\overbrace{000}^{\text{Ra: } c_{i,1}c_{i,2}c_{i,3}}}_{\substack{\text{Vào: } m_i^{(1)}m_i^{(2)}=00 \\ \text{Chuyển sang trạng thái tiếp (a=00)}}} \mid \underbrace{\overbrace{101}^{\text{Ra: } c_{i,1}c_{i,2}c_{i,3}}}_{\substack{\text{Vào: } m_i^{(1)}m_i^{(2)}=10 \\ \text{Chuyển sang trạng thái tiếp (b=10)}}} \mid \underbrace{\overbrace{111}^{\text{Ra: } c_{i,1}c_{i,2}c_{i,3}}}_{\substack{\text{Vào: } m_i^{(1)}m_i^{(2)}=01 \\ \text{Chuyển sang trạng thái tiếp (c=01)}}} \mid \underbrace{\overbrace{010}^{\text{Ra: } c_{i,1}c_{i,2}c_{i,3}}}_{\substack{\text{Vào: } m_i^{(1)}m_i^{(2)}=11 \\ \text{Chuyển sang trạng thái tiếp (d=11)}}}$$

Đối với sơ đồ lập mã hình 3.10 d: Tương tự ta có

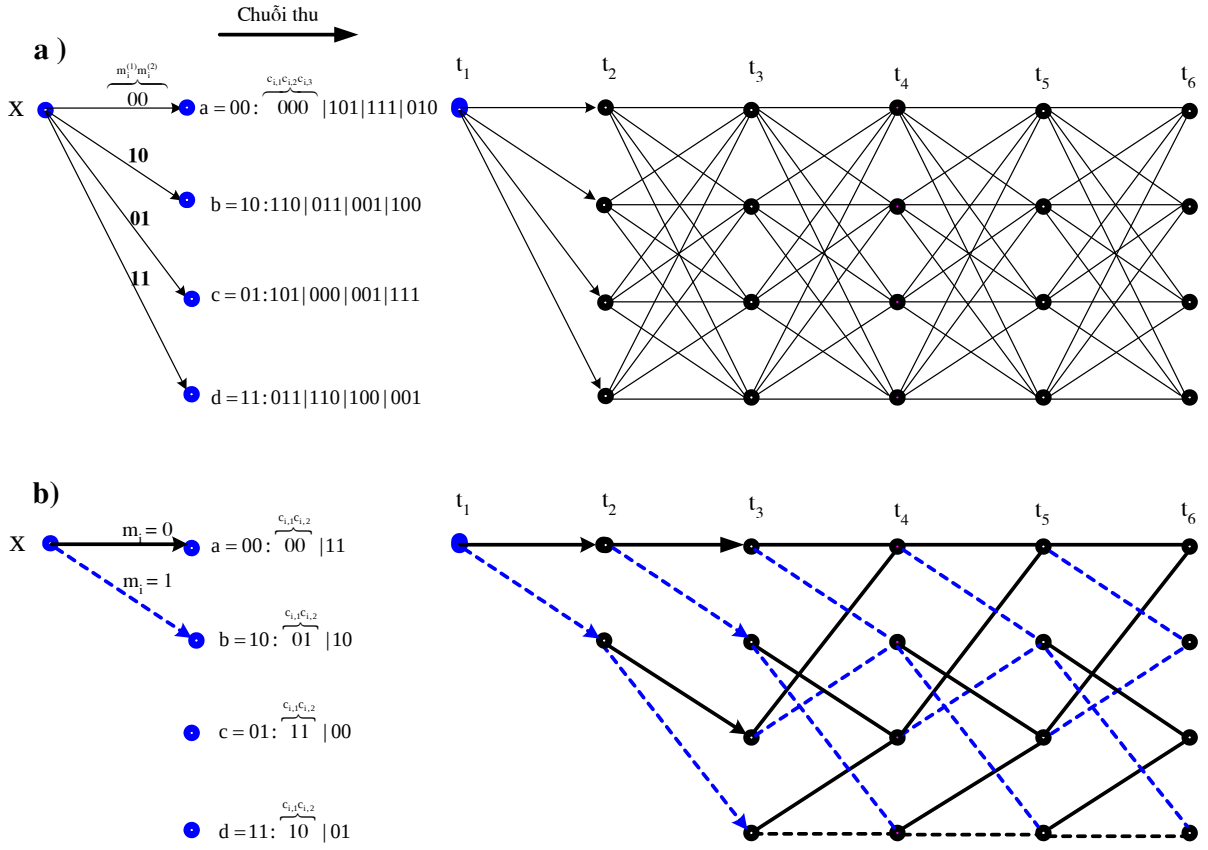
$$\overbrace{a=00}^{\text{Trạng thái trước}} : \underbrace{\overbrace{00}^{\text{Ra: } c_{i,1}c_{i,2}}}_{\substack{\text{Vào: } m_i=0 \\ \text{Chuyển sang trạng thái tiếp (a=00)}}} \mid \underbrace{\overbrace{11}^{\text{Ra: } c_{i,1}c_{i,2}}}_{\substack{\text{Vào: } m_i=1 \\ \text{Chuyển sang trạng thái tiếp (b=10)}}}$$

Cấu trúc biểu đồ lưới tuần hoàn cố định sau khi đạt được độ sâu bằng K (K tầng), từ sau thời điểm này

+ Vào một trạng thái có thể là từ một trong 2^k trạng thái trước đó

+ Ra mỗi trạng thái có thể chuyển vào một trong 2^k trạng thái tiếp theo

Ví dụ: Quá trình xây dựng biểu đồ lưới cho các sơ đồ lập mã hình 3.10a và 3,10b được cho ở hình 3.14 a và b.



Hình 3.14. Các dạng biểu đồ lưới cho các bộ lập mã ở hình 3.10a (hình 3.14 a) và 3.10d (hình 3.14b)

3.4.8. Giải mã xoắn

Tồn tại nhiều thuật toán để giải mã xoắn. Thuật toán Viterbi là thuật toán giải mã khả năng giống nhất và được dùng nhiều nhất, tuỳ vào chuỗi bit thu thuật toán duyệt tìm kiếm (***Search***) thông qua lưới tìm ra đường dẫn giống nhất để tạo ra chuỗi thu đó. Thực chất của thuật toán này là so sánh giữa chuỗi bit thu với tất cả các đường dẫn lưới đến cùng trạng thái tại $t = t_i$. Nếu hai đường dẫn cùng đi vào cùng một trạng thái tại thời điểm $t=t_i$, thì một đường dẫn có tương quan lớn nhất (số đo tốt nhất) hay *khoảng cách* nhỏ nhất được chọn và đường dẫn được chọn này được gọi là đường dẫn sống sót. Quá trình giải mã tiếp tục đi sâu vào trong lưới.

- ✓ Nếu sử dụng giải mã Viterbi quyết định *cứng*, thì thuật toán này tìm đường dẫn trong biểu đồ lưới *sao cho* đường dẫn này có khoảng cách ***Hamming*** cực tiểu so với chuỗi bit thu.

✓ Nếu dùng giải mã Viterbi quyết định *mềm*, thì thuật toán này tìm ra một đường dẫn trong biểu đồ lưới *sao cho* đường dẫn này có khoảng cách ***Euclic*** cực tiểu so với chuỗi bit thu.

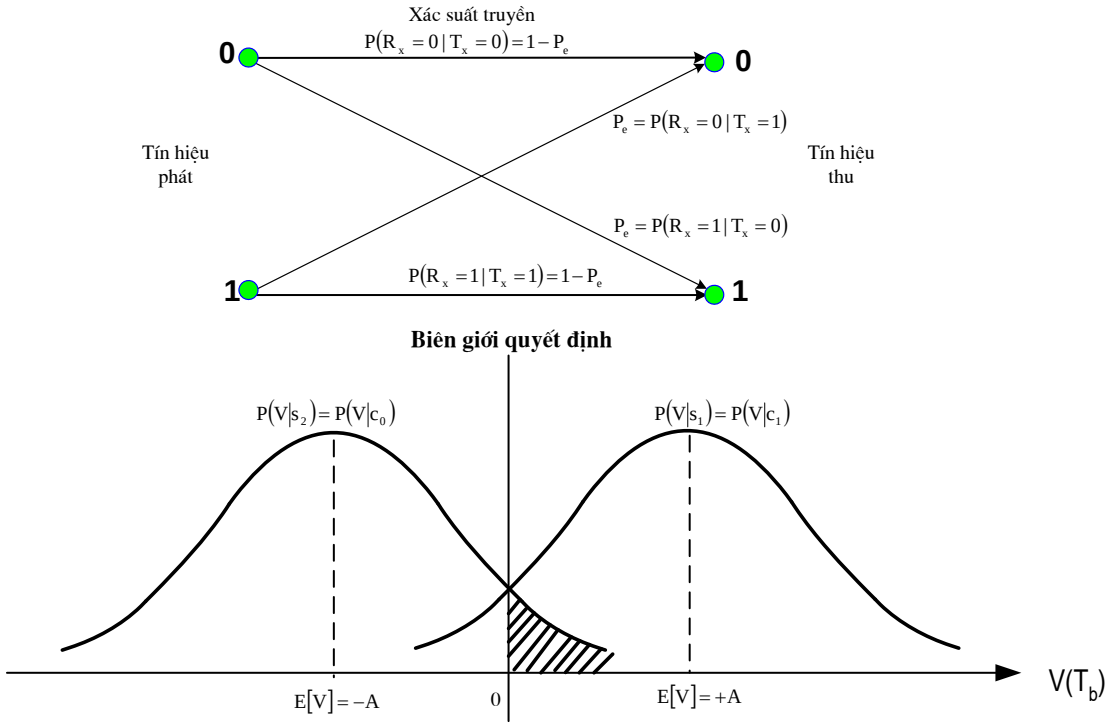
⇒*Vì vậy phương pháp giải mã xoắn Viterbi được coi là phương pháp tối ưu nhất.*

3.8.1. Khái niệm về quyết định cứng và mềm

❖ Giải mã quyết định cứng

➢ *Kênh quyết định cứng*: Kênh BSC là trường hợp đặc biệt của kênh kênh rời rạc không nhớ DMC khi các ký hiệu đầu vào/ra là cơ hai 0 và 1 và xác xuất truyền đối xứng được cho ở hình 3.15 A. Kênh BSC được xây dựng trên kênh AWGN hoàn toàn được đặc trưng bởi xác xuất truyền p.

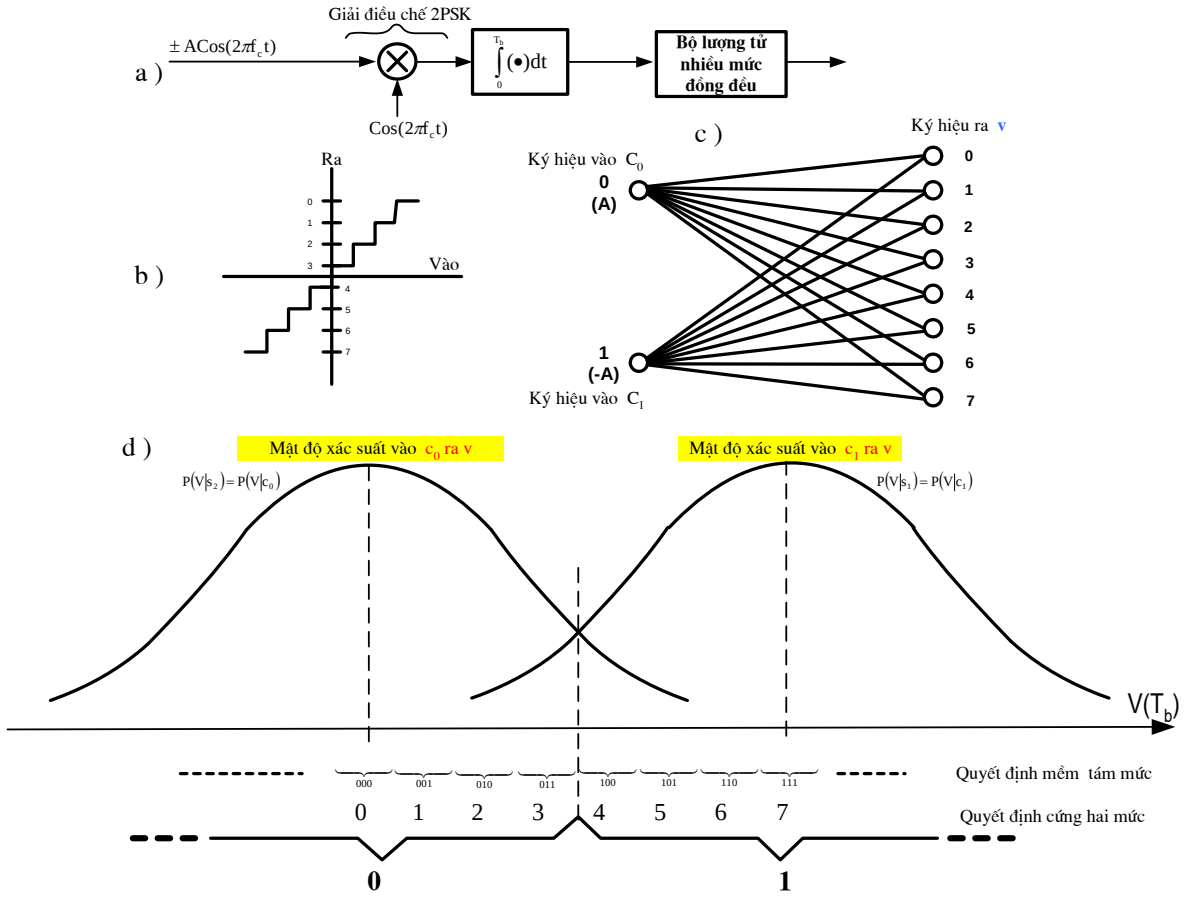
➢ *Giải mã quyết định cứng:* Khi bộ giải mã chỉ nhận hai mức tín hiệu từ bộ giải điều chế được gọi là giải mã quyết định cứng.



Hình 3.15 A. Kênh đối xứng cơ hai BSC (kênh quyết định cứng)

❖ Giải mã quyết định mềm

Khi đầu ra bộ giải điều chế $V(T_b)$ đưa vào bộ giải mã được lượng tử hoá thành nhiều mức (lớn hơn hai mức), thì giả mã được gọi là giải mã quyết định mềm. Hình 3.15B minh hoạ giải mã quyết định mềm với đầu ra bộ giải điều chế $V(T_b)$ được lượng tử hoá hành 8 mức (3 bit).



Hình 3.15B. Kênh rời rạc không nhớ hai mức vào Q mức ra và quyết định cứng, mềm. a) Máy thu; b) Đặc tính truyền đạt của bộ lượng tử nhiều mức; c), d) Phân bố xác suất chuyển đổi cho kênh nhiều mức

3.4.8.2. Giải mã theo khả năng giống nhất ML

Giả sử **m** là vector thể hiện chuỗi bit bản tin, **C** là vector thể hiện chuỗi từ mã đầu ở đầu ra bộ mã hóa đưa vào kênh AWGN và **V** là vector thể hiện chuỗi từ mã ra của kênh này, do tác động của tạp âm nên **V** có thể khác **C**. Bộ giải mã có nhiệm vụ đưa ra ước tính vector bản tin thu **m'**. Nếu tất cả các chuỗi vào đều có xác suất như nhau thì bộ giải mã đạt được xác suất lỗi thấp nhất sẽ là bộ giải mã thực hiện so sánh các xác suất có điều kiện được gọi là các hàm khả năng giống (Likehood Function), $P\left(V\middle|C^{(m)}\right)$, trong đó **V** là vector thu còn $\mathbf{C^{(m)}}$ là vector phát tương ứng với bản tin **m**.

Quy tắc quyết định theo khả năng giống nhất như sau:

Quyết định m' nếu:

$$\ln \underbrace{P\left(V\middle|C^{(m')}\right)}_{Likelihood\ Function} = \underbrace{\max_{\forall\ C^{(m)}} \left(\underbrace{\ln P\left(\underbrace{R_x = V}_{\text{Vector quan trắc}} \middle| \underbrace{T_x = C^{(m)}}_{\text{Chuỗi phát tương ứng bản tin m}} \right)}_{PM(V,C)} \right)}_{\text{Tìm đường dẫn khả năng giống vector thu nhất trong lưới}} \quad \textbf{(3.68)}$$

Vì kênh ***không nhớ*** nên hàm khả năng giống $P\left(V\middle|C^{(m)}\right)$ được xác định như sau:

$$\begin{aligned} PM(V,C) &= \ln P\left(V\middle|C^{(m)}\right) \\ &= \ln\left[\prod_{\forall i} P(V_i C_i)\right] \\ &= \ln\left[\prod_{\forall i}\prod_{j=1}^{n} P\left(V_{ij}\middle|C_{ij}\right)\right] \end{aligned} \quad (3.69)$$

trong đó i là số thứ tự của nhánh trong đường dẫn của chuỗi ở biểu đồ lưới ⇔ tầng thứ i của biểu đồ lưới $1 \le i \le m$ ((m=L+K-1) tầng trong biểu đồ lưới) ⇔ $1 \le t_i \le t_m$.

j là số thứ tự của kí hiệu ở mỗi nhánh ⇔ mỗi nhánh chứa n ký hiệu⇒ $1 \le j \le n$.

n là số nhánh cộng modul-2 ở sơ đồ mã hóa (số kí hiệu/ mỗi nhánh).

Viết lại phương trình (3.69) dưới dạng:

$$\begin{aligned} \underbrace{PM(V,C)}_{\substack{\text{Số đo khả năng giống} \\ \text{của đường dẫn}}} &= \sum_{\forall i} \underbrace{\sum_{j=1}^{n} \ln P(v_{ij} \mid c_{ij})}_{\substack{\mu_i\text{:Số đo khả năng giống} \\ \text{của nhánh thứ i}}} \\ &= \sum_{\forall i} \mu_i \end{aligned} \quad (3.70\ \&3.71)$$

Trong đó PM(V,C) được gọi là số đo khả năng giống của ***đường dẫn***

$$\mu_i = \sum_{j=1}^{n} \ln P(v_{ij} \mid c_{ij}) \quad (3.72)$$

được gọi là số đo khả năng giống của nhánh (⇔tương quan). Rõ ràng để được PM(v,c) max phải được tổng các ***số đo nhánh μ_i cực đại⇔ tương quan lớn nhất ⇔ khoảng cách Hamming cực tiểu***.

❖ Giải mã quyết định cứng (kênh BSC)

Nếu xét cho kênh BSC và các kí hiệu ở đầu ra của bộ giải điều chế được xác định bằng "0" hay "1" ta được:

$$P(V_{ij}|C_{ij}) = \begin{cases} p, & v_{ij} \neq c_{ij} \\ 1-p, & v_{ij} = c_{ij} \end{cases} \quad (3.73)$$

Giả sử các ký hiệu của nhánh thứ i khác so với *chuỗi thu con* ở d_i vị trí nhận được:

$$\begin{aligned} \mu_i &= d_i \ln p + (n - d_i)\ln(1-p) \\ &= \underbrace{d_i \underbrace{\ln\left(\frac{p}{1-p}\right)}_{-A} + \underbrace{n\ln(1-p)}_{-B}}_{\text{vì coi rằng } p<\frac{1}{2}} \\ &= -Ad_i - B \end{aligned} \quad (3.74\ \&3.75\&3.76)$$

vì A và B là các hằng số dương nên có thể định nghĩa lại số đo *khoảng cách Hamming* cho mỗi nhánh như sau

$$\mu_i = -d_i \quad (3.77)$$

Từ phương trình (3.71) ta có

$$PM(V,C) = -\sum_{\forall i} d_i \quad (3.78)$$

Từ phương trình (3.78) có thể phát biểu ***quy tắc quyết định cứng là***: Chọn $C^{(m')}$ sao cho tổng khoảng cách Hamming của các nhánh ***giữa*** vector thu V và vector phát phát $C^{(m)}$ là nhỏ nhất. Vậy có thể định nghĩa số đo ở phương trình (3.77) và (3.78)

$$\mu_i = d_i \quad \text{và} \quad PM(V,C) = \sum_{\forall i}\mu_i = \sum_{\forall i} d_i \quad (3.79)$$

⇒ <u>Như vậy</u>: Nhiệm vụ của giải mã quyết định cứng là tìm một đường dẫn trong biểu đồ lưới (biểu đồ lưới thể hiện cho tập tất cả các từ mã $C^{(m)}$ ⇔ tập các đường dẫn có thể có của mã) một đường dẫn $C^{(m')}$ có số đo khoảng cách Hamming so với chuỗi thu V là nhỏ nhất.

$C^{(m')} = C^{(m)}$ nếu và chỉ nếu $\ln P(V|C^{(m)}) = \max_i \{-Ad_i - B\}$

hay $C^{(m')} = C^{(m)}$ nếu và chỉ nếu $d_k = \min_i \{d_i\}$

❖ Giải mã quyết định mềm

Khi này, các ký hiệu ở đầu vào bộ giải mã gồm nhiều mức (mỗi ký hiệu được lượng tử hoá thành nhiều mức). Đầu vào bộ giải mã của ký hiệu thứ j của nhánh i từ đầu ra nhiều mức của giải điều chế 2-PSK (i chạy từ 1 đến Q mức) đưa vào bộ giải mã như sau

$$V_{ij} = \sqrt{E_b}(1 - 2C_{ij}) + n_{ij} \quad (3.80)$$

trong đó *(j thể hiện cho ký hiệu thứ j ở đầu vào bộ lượng tử hoá và i thể hiện cho mức thứ i của ký hiệu thứ j được lượng tử hoá thành Q mức)* n_{ij} là tạp âm trắng cộng Gausơ có trung bình không và v_{ij} là ***biến ngẫu nhiên*** có trung bình là $\sqrt{E_b}\,(1-2c_{ij})$. Phân bố xác suất có điều kiện ở đầu ra bộ giải điều chế được xác định bởi

$$P\left(V_{ij}\middle|C_{ij}\right)=f_{V_{ij}}\left(V_{ij}\middle|C_{ij}\right)=\frac{1}{\sqrt{2\pi\sigma_{V_{ij}}^2}}\exp\left\{-\frac{\left[V_{ij}-E\left(V_{ij}\right)\right]^2}{2\sigma_{V_{ij}}^2}\right\}$$
$$=\frac{1}{\sqrt{\pi N_0}}\exp\left\{-\frac{\left[V_{ij}-\sqrt{E_b}\left(1-2c_{ij}\right)\right]^2}{N_0}\right\} \qquad (3.81)$$

Đặt phương trình (3.81) vào phương trình (3.72), ta được:

$$\mu_i=\frac{1}{N_0}\left\{-\sum_{j=1}^{n}v_{ij}^2-nE_b+2\sqrt{E_b}\sum_{j=1}^{n}v_{ij}\left(1-2c_{ij}\right)\right\}-\frac{n}{2}\ln\left(\pi N_0\right) \qquad (2.75)$$

Loại bỏ các thành phần chung cho tất cả các nhánh ta có thể viết lại số đo khả năng giống của các nhánh ở dạng:

$$\mu_i=\sum_{j=1}^{n}v_{ij}\left(1-2c_{ij}\right) \qquad (2.76)$$

Từ phương trình (3.70) ta nhận được số đo khả năng giống của đường dẫn như sau:

$$PM\left(V,C\right)=\sum_{\forall i}\sum_{j=1}^{n}v_{ij}\left(1-2c_{ij}\right) \qquad (2.77)$$

Như vậy nhiệm vụ của giải mã quyết định mềm là phải chọn ra một đường dẫn có số đo khả năng giống lớn nhất

3.4.8.3. Giải mã Viterbi quyết định cứng

Nhiệm vụ của giải mã trong trường hợp này là phải tìm được một đường dẫn trong lưới có tổng số đo lớn nhất (có khoảng cách Hamming tích luỹ từ tất cả các nhánh thuộc đường dẫn đó so với chuỗi bit thu là nhỏ nhất ***hay*** tương quan với chuỗi bit thu là lớn nhất). Nếu có N bit vào bộ giải mã thì sẽ có N nút, mà từ mỗi nút có 2^k nhánh ra, tương ứng sẽ có 2^{Nk} đường dẫn phải tìm. Đây là nhiệm vụ không thể giải quyết nổi thậm chí khi N không lớn. Thuật toán Viterbi cho phép giảm số đường dẫn cần tìm tỷ lệ tuyến tính với Nk. Thuật toán này như sau.

Lưu ý: Do phía phát độn (K-1)k bit "0" nên luôn bắt đầu từ một trạng thái toàn không (*all-0 state*), di chuyển thông qua lưới theo các nhánh tương ứng với chuỗi bit thu và trở về trạng thái toàn không (*all-0 state*).

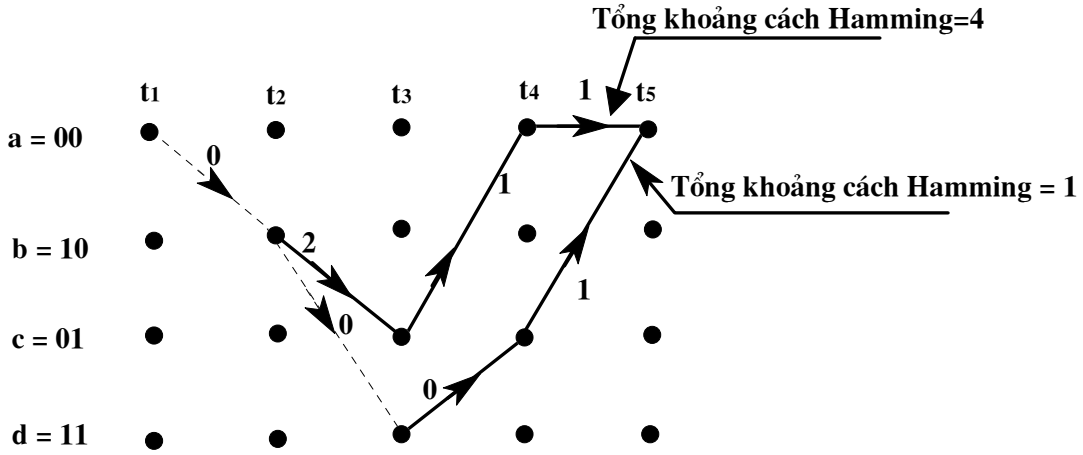
Đường dẫn: Đường dẫn là tập hợp các ***nhánh*** được nối với nhau bởi các ***nut*** trạng thái.

Số đo đường dẫn: là tổng khoảng cách Hamming tích luỹ (so với chuỗi bit thu) của tất cả các nhánh thuộc đường dẫn đó tại thời điểm xét t_i.

Số đo của nhánh (khoảng cách Hamming của nhánh): là số vị trí bit khác nhau giữa n bit trên nhánh đó với n bit của từ mã thu ở đoạn từ t_i đến t_{i+1}.

Đường dẫn sống sót: Nếu có hai hay nhiều đường dẫn cùng hội nhập vào một nút trạng thái thì giữ lại đường dẫn có số đo lớn nhất (khoảng cách Hamming tích luỹ nhỏ nhất) và loại bỏ các đường dẫn còn lại. Đường dẫn được giữ lại được gọi là đường dẫn sống sót.

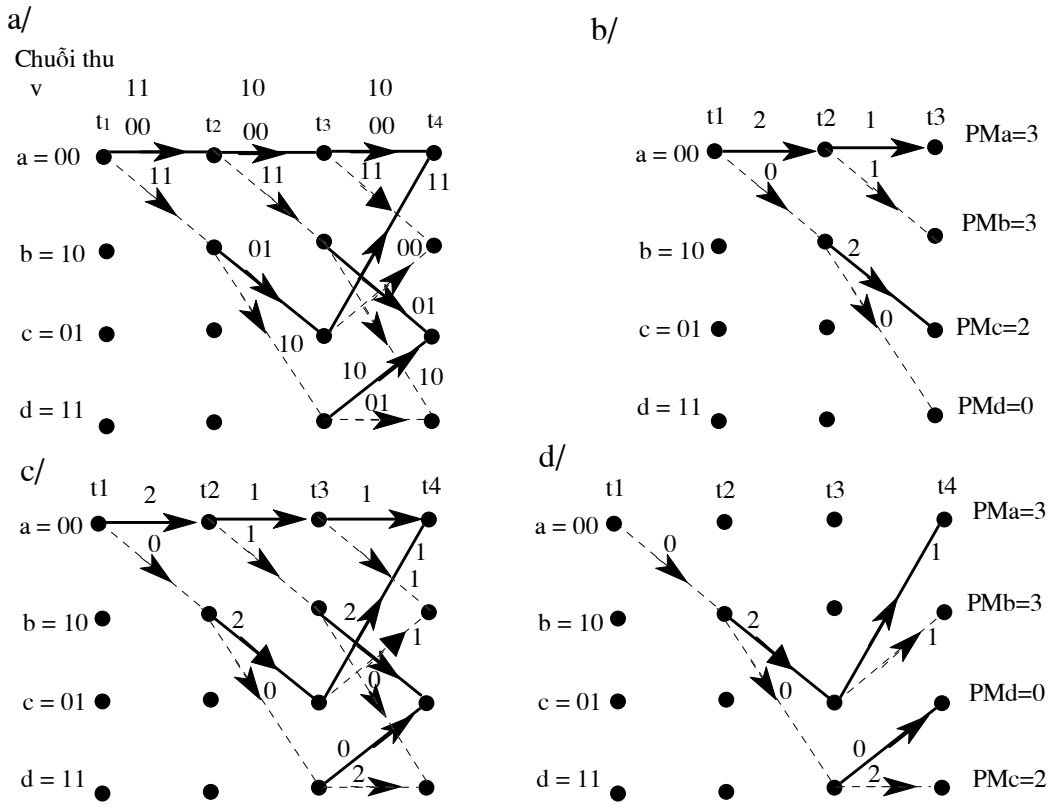
Ví dụ: Xét giải mã cho sơ đồ bộ lập mã hình 3.10d, khi này ta được hai nhánh hội nhập như ở hình 3.16 với nhãn trên mỗi nhánh là số đo nhánh và trong hai đường dẫn cùng hội nhập vào trạng thái **a=00** ở thời điểm t_5 thì đường dẫn có số đo 1 là đường dẫn sống sót.



Hình 3.16. Các số đo đường dẫn cho hai đường hội nhập

Ở thí dụ này, tại mỗi thời điểm t_i trên lưới có $2^{(K-1)k}$ =4 trạng thái, mỗi một trạng thái có thể hội nhập hai đường dẫn. Giải mã Viterbi bao gồm việc tính toán các số đo cho hai đường dẫn hội nhập vào mỗi trạng thái và loại bỏ một trong hai đường này. Việc tính toán này được thực hiện cho từng nút trong số 2^{K-1} nút ở thời điểm t_i; sau đó bộ giải mã dịch đến thời điểm t_i+1 và lặp lại quá trình này. Giả thiết chuỗi số liệu vào là **m**, từ mã phát là **c** và từ mã thu là **v**. Giả thiết rằng bộ giải mã biết chính xác trạng thái đầu của lưới (trong thực tế không cần thiết giả thiết này, nhưng nó sẽ đơn giản việc giải thích). ở thời điểm t_1 từ thu được là 11. Từ trạng thái 00 chỉ có thể có hai chuyển đổi đến trạng thái 00 hay 10 (hình 3.7a). Chuyển đổi trạng thái 00→00 có số đo nhánh là 2; chuyển đổi trạng thái 00→10 có số đo nhánh là 0. ở thời điểm t_2 có thể có hai nhánh rời bỏ một trạng thái (3.17b). Các số đo đường dẫn tích luỹ của các nhánh này được ký hiệu là PMa, PMb, PMc và PMd tương ứng với các trạng thái kết cuối. Tại thời điểm t_3 trên hình 3.17 c lại có hai nhánh đi ra từ một trạng thái. Kết quả là sẽ có hai đường dẫn hội nhập vào một trạng thái ở thời điểm t_4. Như đã nói ở trên ta có thể loại bỏ một nhánh có số đo đường dẫn lớn hơn. Nếu như số đo của hai đường nhánh hội nhập có cùng giá trị thì việc chọn nhánh bị loại bỏ dựa trên một quy tắc tuỳ chọn. Các đường dẫn sống sót đi vào từng trạng thái được cho ở hình 3.17 d. ở thời điểm này của quá trình giải mã chỉ còn một đường dẫn duy nhất giữa t1 và t_2. Vì thế bây giờ bộ giải mã có thể quyết định rằng chuyển đổi giữa t_1 vào t_2 là 00→10. Vì chuyển đổi này được tạo ra bởi bit vào "1" nên bộ giải mã sẽ đưa ra "1" là bit được giải mã đầu tiên. Đến đây ta hiểu rằng giải mã nhánh sống sót sẽ trở nên

dễ dàng như thế nào nếu ta vẽ các nhánh lưới bằng đường liên tục cho bit vào "0" và đường ngắt quãng cho bit vào "1". Lưu ý rằng bit thứ nhất chỉ được giải mã chừng nào việc tính toán số đo đường dẫn chưa vào sâu trong lưới. Đối với thực hiện thông thường của bộ giải mã, điều này thể hiện sự trễ giải mã, nó có thể gấp năm lần độ dài hạn chế. ở thí dụ này tương ứng với N=4 bit vào chỉ cần xét 4^N=16 đường dẫn.



Hình 3.17. Chọn các đường sống sót:
a/ Biểu đồ lưới và chuỗi thu,
b) Các đường sống sót ở t_2.,
c) So sánh số đo ở t_4,
d) Các đường sống sót ở t_4.

ở mỗi bước tiếp theo trong quá trình giải mã luôn luôn có thể có hai đường dẫn đi vào một trạng thái; một trong hai đường này sẽ bị loại bỏ bằng cách so sánh các số đo đường dẫn. Bộ giải mã tiếp tục như vậy để đi sâu hơn vào lưới và đưa ra các quyết định về các bit số liệu đầu vào bằng cách loại bỏ tất cả các đường dẫn trừ một đường.

❖ Tóm tắt thuật toán Viterbi

Chọn ra được một đường dẫn thông qua lưới, từ mã của chúng (ký hiệu **C**) có khoảng cách *Hamming* cực tiểu so với chuỗi thu **V**. ***Vì*** đường dẫn được xét bắt đầu từ trạng thái toàn không và trở lại trạng thái toàn không, ***nên*** giả định rằng đường dẫn này trải dài toàn bộ ***(L+K-1) nhánh ((L+K-1) tầng)*** **và** ***vì*** mỗi nhánh tương ứng

với n bit ra bộ lập mã ***nên*** C và V đều có sô bit là (**L+K-1)×n**. ***Nếu*** ký hiệu chuỗi bit tương ứng nhánh thứ ***i*** là c_i và v_i, trong đó $1 \le i \le (L+K-1)$ lưu ý rằng (độ dài c_i và v_i trên mỗi nhánh đều là n). ***thì*** khoảng cách Hamming giữa C và V được xác định là.

$$\underbrace{d(V,C)}_{PM(V,C)} = \sum_{i=1}^{L+K-1} d\ \underbrace{(c_i, v_i)}_{\text{cặp n bit/nhánh}} \qquad \text{(ext 1)}$$

1. Phân tích chuỗi thu vào (L+K-1) các *chuỗi con* mỗi chuỗi con có độ dài n.
2. Vẽ một lưới có độ sâu (L+K-1). Đối với ***K-1 tầng*** cuối cùng của lưới, chỉ chỉ cần vẽ các ***đường dẫn*** tương ứng với các chuỗi đầu vào toàn bit không. [Vì biết rằng chuỗi đầu vào đó đã được độn (K-1) k số 0 ở phía phát].
3. Đặt i=1 và đặt số đo của trạng thái toàn không ban đầu bằng 0.
4. Tìm khoảng cách Hamming chuỗi thu con thứ i (n bit trên nhánh thứ i) với tất cả các ***nhánh*** đang nối từ các trạng thái tầng thứ i đến các trạng thái tầng thứ (i+1) của lưới (*số đo trên mỗi nhánh*).
5. Cộng các khoảng cách này với các số đo của các trạng thái tầng thứ i để đạt được các ứng cử viên số đo đối với các trạng thái tầng thứ (i+1). ***Vì*** mỗi trạng thái của tầng thứ (i+1), có 2^k số đo ứng cử viên, ***nên*** mỗi số đo tương ứng với một nhánh kết thúc tại trạng thái đó.
6. Với mỗi trạng thái tại tầng thứ (i+1), chọn ra một số đo ứng cử viên cực tiểu và đánh nhãn cho nhánh tương ứng với giá trị cực tiểu này như là một đường sống sót và ấn định giá trị cực tiểu của các ứng cử viên số đo **là** các số đo của các trạng thái tầng thứ (i+1).
7. Nếu i= (L+K-1) thì chuyển bước tiếp theo; ngược lại thì chuyển về bước 4.
8. Bắt đầu bằng trạng thái toàn không tại tầng thứ ((L+K-1)+1=L+K), quay trở lại thông qua lưới dọc theo các đường sống sót để tiến đến trạng thái khởi đầu toàn không. Đường dẫn này là đường dẫn tối ưu và chuỗi bit thông tin đầu vào tương ứng với đường dẫn đó là một chuỗi thông tin được giải mã khả năng giống nhất. Để đạt được sự ước đoán về chuỗi bit vào tốt nhất phải loại bỏ (K-1)k các bit 0 cuối cùng từ chuỗi này (do phía phát độn (K-1)k số "0").

3.4.8.4. Giải mã Viterbi theo quyết định mềm

Có ba sự khác nhau

1. Thay vì xét chuỗi V, ta khảo sát trực tiếp với vector r, vector đầu ra của bộ giải điều chế số tối ưu (loại bộ lọc thích hợp hoặc bộ tương quan).
2. Thay vì xét chuối nhị phân c (có giá trị 0,1) ta khảo sát chuỗi c' tương ứng với theo.

$$c'_{ij}=\begin{cases}+\sqrt{E}.....\text{nếu}.....c_{ij}=1\\ -\sqrt{E}.....\text{nếu}.....c_{ij}=0\end{cases} \qquad \text{với } 1\le i\le L+K-1 \text{ và } 1\le j\le n \qquad \text{(ext 2)}$$

3. Thay vì xét khoảng cách Hamming, ta dùng khoảng cách Euclic.

Từ trước, ta có.

$$d^2{}_E(c',r)=\sum_{i=1}^{m} d^2{}_E(c_i',r_i) \qquad \text{(ext 3)}$$

Từ phương trình (ext 1) và (ext 3) thấy rõ vấn đề mà cần phải giải quyết là: Khi cho một vector a, tìm một đường dẫn thông qua lưới bắt đầu từ trạng thái toàn không và kết thúc tại trạng thái toàn không ***sao cho*** một vài số đo khoảng cách giữa a và một chuỗi b tương ứng với đường dẫn đó được cực tiểu. Đểm mấu chốt để dễ dàng thực hiện bài toán là trong cả hai trường hợp xét, khoảng cách giữa a và b đều được viết là tổng các khoảng cách tương ứng với các nhánh riêng lẻ của đường dẫn. Điều này dễ được nhận thấy từ (ext 1) và (ext 3).

Giả sử xét mã xoắn có k=1 ⇔ nghĩa là chỉ có hai nhánh đi vào mỗi trạng thái trong lưới. Nếu đường dẫn tối ưu tại một điểm nào đó đi qua trạng thái S, thì có hai ***đường dẫn*** nối trạng thái S đến các trạng thái trước S_1 và S_2 (xem hình: S_1 và S_2 là hai trạng thái cùng được nối đến trạng thái S).

Nếu muốn biết nhánh nào trong hai nhánh này là ứng cử tối để giảm thiểu toàn bộ khoảng cách, ***thì*** phải cộng toàn bộ số đo (minimum) tại các trạng thái S_1 và S_2 với số đo của các nhánh nối hai trạng thái này tới trạng thái S. Sau đó quan sát nhánh mà có số đo tổng ***cực tiểu*** tích luỹ đến trạng thái S là một ứng cử sẽ được quan tâm cho trạng thái sau trạng thái S. Nhánh này được gọi là nhánh ***sống sót*** tại trạng thái S, các nhánh khác hoàn toàn không phải là ứng cử viên thích hợp và được xoá. Đến đây, sau khi được xác định là sống sót tại trạng thái S. Ta lưu lại số đo cực tiểu tích lỹ đến trạng thái này và có thể chuyển tới trạng thái tiếp theo. Thủ tục này được thực hiện liên tục tới khi ta đạt được trạng thái toàn không ở tại điểm cuối của lưới.

Trường hợp k >1, chỉ khác ở chỗ tại mỗi ***tầng*** ta phải chọn một ***đường dẫn sống sót*** từ 2^k ***nhánh*** đi vào trạng thái S.

❖ Thuộc tính khoảng cách và hàm truyền đạt của mã xoắn

✓ *Thuộc tính khoảng cách*

Từ lý thuyết mã khối, biết rằng mã có khoảng cách cực tiểu d_{min} có thể sửa t lỗi trong đó

$$t \le \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor$$

Tuy nhiên với mã xoắn d_{min} được thay bằng d_f được gọi là khoảng cách tự do (cực tiểu) vì thế khả năng sửa lỗi của mã xoắn là

$$t_{error-correct\ of\ COV} \le \left\lfloor \frac{d_f - 1}{2} \right\rfloor$$

biểu thức trên là kí hiệu chỉ lấy phần nguyên lớn nhất.

Đặc tính hoá mã xoắn cho trước đưới dạng khả năng sửa lỗi dựa vào việc xác định khoảng cách tự do cực tiểu từ việc khảo sát biểu đồ lưới.

- **Lưu ý rằng:**

Khoảng cách tối thiểu d_f (đơn giản gọi khoảng cách tự do) của tập hợp tất cả các đường dẫn có độ dài bất kỳ tính từ điểm chúng rời ra sau đó lại hội nhập lại đường dẫn toàn không được gọi là khoảng cách tự do tối thiểu $\Rightarrow$ Tổng quát định nghĩa: Khoảng cách tự do d_f là khoảng cách Hamming tối thiểu giữa hai từ mã bất kỳ trong mã đó (mã được xét).

✓ *Hàm truyền đạt của mã xoắn*

Mặc dù ta có thể dùng biểu đồ lưới để tính toán khoảng cách tự do nhưng ta cũng có thể nhận được biểu thức để tính toán khoảng cách này. Biểu thức này được gọi là hàm truyền đạt của mã xoắn.

Biểu thức tổng quát:

$$T(D) = \sum_{d=d_f}^{\infty} a_d D^d \qquad (3.89)$$

Trong đó: $\mathbf{a_d}$ là số đường dẫn khác không có khoảng cách Hamming là d tương ứng với đường dẫn toàn không mà hội nhập với đường dẫn toàn không tại node j và tách (rẽ) ra từ đường dẫn toàn không tại một số node trước đó. (hay d là khoảng cách giữa đường dẫn này với đường dẫn toàn không)

PHỤ LỤC 1.1

Chương trình mô phỏng

❖ Để xét nguyên lý hoạt động của mã hoá khối tuyến tính (n,k) cho trước

- ✓ Ma trận tạo mã
- ✓ Ma trận bản tin

❖ Yêu cầu sau khi thực hiện chương trình sinh viên phải hiểu

- ✓ Quá trình (thuật toán) tạo ra bản tin m xắp xếp các bit thông tin.
- ✓ Thuật toán nhân hai ma trận tạo ra Codeword đầu ra bộ mã hoá
- ✓ Quá trình tìm trọng lượng cực tiểu của mã.
- ✓ Các xác định số liệu vào ra của bộ lập mã.

❖ Chương trình mô phỏng

```
function y = CS88(k)
echo on;
k=input('Nhap do dai k cho ma C(n,k) (k=4) =');
% Tạo ma trận bản tin đầu vào m
for i=1:2^k,
  for j=k:-1:1,
    if rem(i-1,2^(-j+k+1))>=2^(-j+k)
      u(i,j)=1;
    else
      u(i,j)=0;
    end
    echo off;
  end
end
echo on;
%g=input('Definde G & enter the Generator matrix=');
  g=[1 0 0 1 1 1 0 1 1 1
       1 1 1 0 0 0 1 1 1 0
       0 1 1 0 1 1 0 1 0 1
    1 1 0 1 1 1 1 0 0 1];
% Tạo ma trận số liệu ra
  c=rem(u*g,2);
% Tìm trong lượng mã cực tiểu
  w_min=min(sum((c(2:2^k,:))'));
% Hiển thị kết quả
  disp('          Ma tran ban tin m ');
  disp(u);
  disp('          Ma tran tu ma ra ');
  disp(c);
  disp('          Trong luong ma cuc tieu Wm')
  disp(w_min);
```

Kết quả ma trận m được tạo ra

```
 0   0   0   0
 0   0   0   1
 0   0   1   0
 0   0   1   1
 0   1   0   0
```

0	1	0	1
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

Ma trận Codeword đầu ra bộ mã hoá

0	0	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	0	0	1
0	1	1	0	1	1	0	1	0	1
1	0	1	1	0	0	1	1	0	0
1	1	1	0	0	0	1	1	1	0
0	0	1	1	1	1	0	1	1	1
1	0	0	0	1	1	1	0	1	1
0	1	0	1	0	0	0	0	1	0
1	0	0	1	1	1	0	1	1	1
0	1	0	0	0	0	1	1	1	0
1	1	1	1	0	0	0	0	1	0
0	0	1	0	1	1	1	0	1	1
0	1	1	1	1	1	1	0	0	1
1	0	1	0	0	0	0	0	0	0
0	0	0	1	0	0	1	1	0	0
1	1	0	0	1	1	0	1	0	1

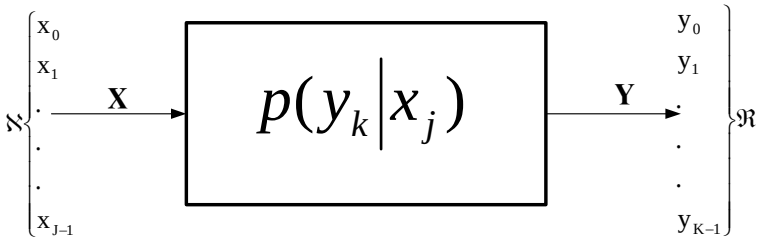
Trọng lượng mã cực tiểu Wm = 2

PHỤ LỤC 2.1

KÊNH KHÔNG NHỚ RỜI RẠC DMC

Descrete Memoryless Channel

Kênh không nhớ rời rạc được mô hình hoá bởi hình PL3.1.



Hình 3.1. Kênh không nhớ rời rạc DMC

Kênh không nhớ rời rạc là mô hình thống kê với: Đầu vào X. Đầu ra Y là phiên bản tạp âm của X. Trong đó X và Y đều là biến ngẫu nhiên.

Mỗi đơn vị thời gian, kênh nhận ký hiệu đầu vào X được chọn từ $\aleph$ và đáp ứng ra bằng cách nó phát ký hiệu Y từ $\Re$.

Kênh được gọi là "rời rạc- Descrete" khi cả $\aleph$ và $\Re$ đều có kích thước hữu hạn và được gọi là "không nhớ Memoryless" khi ký hiệu đầu ra hiện thời chỉ phụ thuộc ký hiệu vào hiện thời và không phụ thuộc vào bất kỳ một ký hiệu vào trước đó.

Hình PL3.1. minh hoạ kênh rời rạc không nhớ. Kênh được mô tả dưới dạng các đầu vào và đầu ra như sau.

+ Các đầu vào: $\aleph = \{x_0, x_1, ..., x_{J-1}\}$ (PL3.0a)

+ Các đầu ra: $\Re = \{y_0, y_1, ..., y_{J-1}\}$ (PL3.0b)

+ Tập các xác suất truyền:

$$P(y_k|x_j) = P(Y = y_k|X = x_j) \quad \text{Với } \forall\, j \,\&\, k \qquad \text{(PL3.1)}$$

Hiển nhiên ta có.

$$0 \le P(y_k|x_j) \le 1 \qquad \text{với mọi j và k} \qquad \text{(PL3.2)}$$

Bảng mẫu tự đầu vào $\aleph$ và đầu ra $\Re$ không nhất thiết phải có cùng kích thước. Chẳng hạn, trong mã hoá kênh kích thước đầu ra K của $\Re$ có thể lớn hơn kích thước J của đầu vào $\aleph$ vì thế mà K≥J. Mặt khác, ta gặp phải tình huống trong đó kênh phát ra cùng một ký hiệu khi một trong hai ký hiệu đầu vào được gửi đi khi đó K≤J.

Xác suất truyền $p(y_k|x_j)$ là xác suất có điều kiện mà đầy ra $Y=y_k$, đã biết đầu vào $X=x_j$. Do hạn chế về vật lý làm ảnh hưởng đến độ tin cậy khi truyền tin qua kênh gây ra lỗi. Vì vậy, ***khi k=j, thì xác suất truyền $p(y_k|x_j)$ thể hiện xác suất thu có điều kiện đúng, và khi k≠j, thể hiện xác suất lỗi có điều kiện***.

Phương pháp phổ biến để mô tả kênh không nhớ rời rạc là xắp xếp các xác suất truyền của kênh dưới dạng ma trận như sau.

$$P = \begin{bmatrix} p(y_0|x_0) & p(y_1|x_0) & ..... & p(y_{K-1}|x_0) \\ p(y_0|x_1) & p(y_1|x_1) & ..... & p(y_{K-1}|x_1) \\ . & . & & . \\ & . & ..... & . \\ p(y_0|x_{J-1}) & p(y_1|x_{J-1}) & ..... & p(y_{K-1}|x_{J-1}) \end{bmatrix}_{J\times K} \qquad \text{(PL3.3)}$$

Ma trận P kích thước J×K được gọi là ma trận kênh (Channel Matrix). Lưu ý mỗi hàng của ma trận P tương ứng với đầu vào kênh cố định (Fixed Channel Input), còn mỗi cột của ma trận P tương ứng với đầu ra kênh cố định (Fixed Channel Output). Cũng cần lưu ý rằng, thuộc tính cơ bản của ma trận kênh P là tổng các phần tử dọc theo một hàng nào đó của ma trận luôn bằng 1, nghĩa là.

$$\sum_{k=0}^{K-1} p(y_k|x_j) = 1 \qquad \text{Với mọi j} \qquad \text{(PL3.4).}$$

Giả sử đầu vào kênh rời rạc không nhớ được chọn tương ứng với phân phối xác suất $\{p(x_j), j=0,1,...J-1\}$. Nói cách khác, sự kiện đầu vào $X=x_j$ xuất hiện với xác suất

$$P(x_j) = P(X=x_j) \qquad \text{với } j = 0,1,...,J-1 \qquad \text{(PL3.5)}$$

Xác định biến ngẫu nhiên X biểu thị cho đầu vào kênh, xác định biến ngẫu nhiên Y biểu thị cho đầu ra kênh. Phân phối xác suất hợp của các biến ngẫu nhiên X và Y được cho bởi.

$$\begin{aligned} p(x_j, y_k) &= P(X = x_j, Y = y_k) \\ &= P(Y = y_k \mid X = x_j)P(X = x_j) \\ &= p(y_k \mid x_j)p(x_j) \end{aligned} \qquad \text{(PL3.6)}$$

Phân phối xác suất mép (*Marginal Probability Distribution*) của biến nhẫu nhiên ra Y đạt được bằng cách lấy trung bình phụ thuộc của $p(x_j,y_k)$ trên x_j như sau.

$$\begin{aligned} p(y_k) &= P(Y = y_k) \\ &= \sum_{j=0}^{J-1} P(Y = y_k \mid X = x_j)P(X = x_j) \\ &= \sum_{j=0}^{J-1} P(y_k \mid x_j)P(x_j) \qquad \text{Với } k = 0,1,\ldots, K-1 \end{aligned} \qquad \text{(PL3.7)}$$

Với J=K, thì xác suất lỗi ký hiệu trung bình Pe được xác định là xác suất mà biến ngẫu nhiên đầu ra Y_k khác với biến ngẫu nhiên đầu vào X_j, được lấy trung bình trên toàn bộ k≠j. Vì vậy ta viết.

$$\begin{aligned} P_e &= \sum_{\substack{k=0 \\ k\neq j}}^{K-1} P(Y = y_k) \\ &= \sum_{\substack{k=0 \\ k\neq j}}^{K-1} \sum_{j=0}^{J-1} P\left(\underbrace{y_k}_{R_x} \middle| \underbrace{x_j}_{T_x} \right) P\left(\underbrace{x_j}_{T_x} \right) \end{aligned} \qquad \text{(PL3.8)}$$

$\Rightarrow$ Hiệu (1- Pe) là xác suất thu đúng trung bình.

Các xác suất $p(x_j)$ với j = 0,1,...,J-1, được coi là xác suất tiên nghiệm (*Priori-probability*) của các ký hiệu vào.

Phương trình PL3.7 cho biết: nếu biết đầu vào có xác suất tiên nghiệm $p(x_j)$ và biết ma trận kênh, ma trận xác suất truyền $p(y_k|x_j) \Rightarrow$ thì tính được xác suất ký hiệu ra $p(y_k)$. Phần sau ta tổng kết lại, ta cho $p(x_j)$, $p(y_k|x_j)$ trong trường hợp này ta có thể tính $p(y_k)$ bằng phương trình PL3.7.

<u>Ví dụ</u>: Kênh nhị phân đối xứng BSC Binary Symmetric Channel.

Kênh BSC là trường hợp cụ thể của kênh không nhớ rời rạc DMC với J=K=2.

$$P_e = \sum_{\substack{k=0 \\ k\neq j}}^{1}\sum_{j=0}^{1} P(y_k|x_j)P(x_j)$$

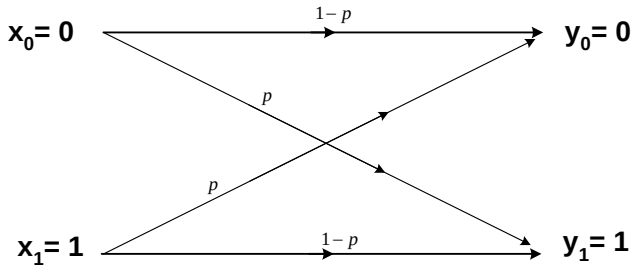
$$= \sum_{\substack{k=0 \\ k\neq j}}^{1} P(y_k|x_0)P(x_0) + P(y_k|x_1)P(x_1)$$

$$= P(y_1|x_0)P(x_0) + P(y_0|x_1)P(x_1)$$

Kênh có hai ký hiệu đầu vào ($x_0 = 0$, $x_1 = 1$) và hai ký hiệu đầu ra ($y_0 = 0$, $y_1 = 1$). Kênh được gọi là đối xứng vì xác suất thu được 1 nếu 0 được gửi đi bằng xác xuất thu được bit 0 nếu bit 1 được gửi đi tức là $p(1|0) = p(0|1) \Rightarrow$ ma trận kênh là.

$$P = \begin{bmatrix} p(0|0) & p(1|0) \\ p(0|1) & p(1|1) \end{bmatrix}$$

Xác suất truyền hoặc xác suất lỗi có điều kiện được ký hiệu là p. Sơ đồ xác suất truyền của BSC được cho ở hình PL3.2.



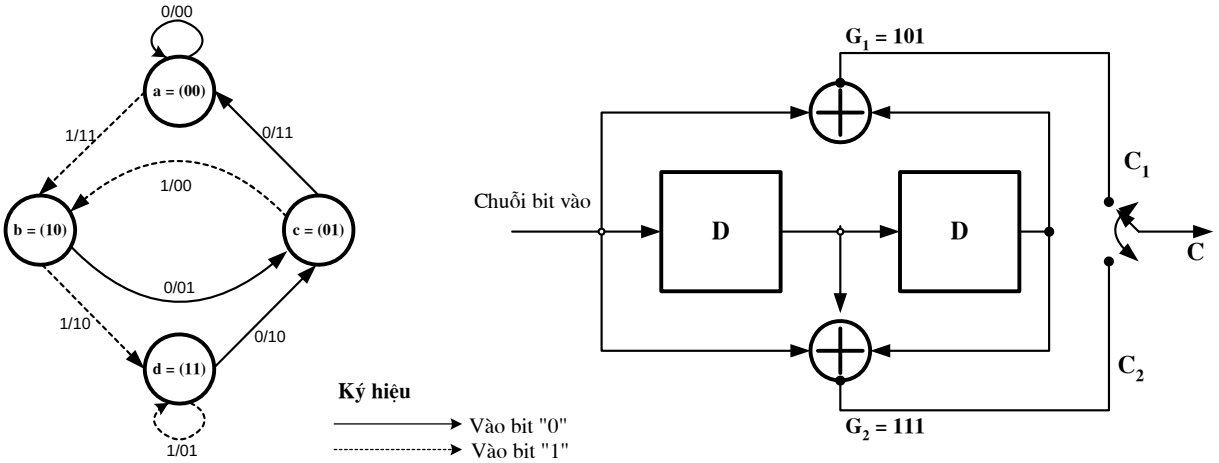
Hình PL3.2: Kênh đối xứng cơ hai BSC

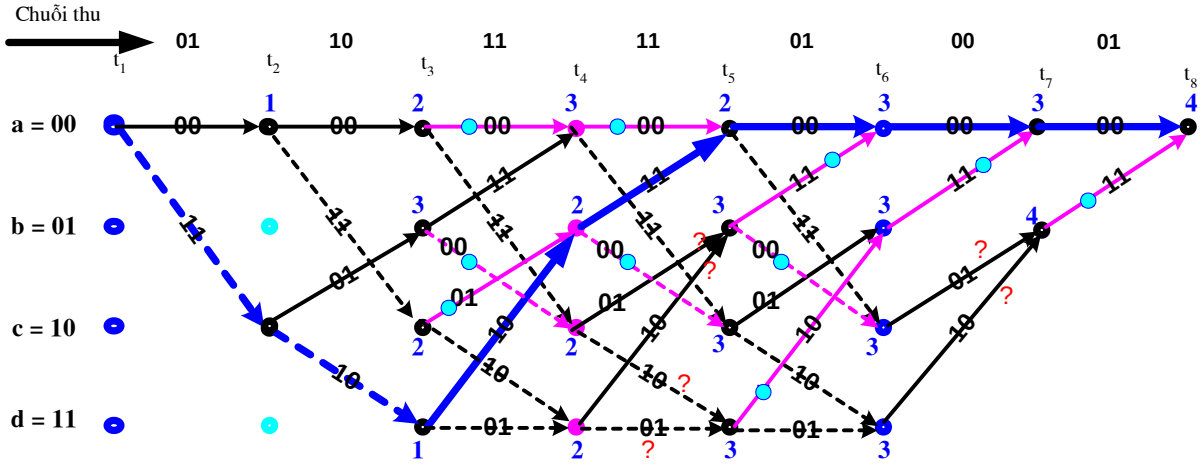
PHỤ LỤC 2.2

LẬP MÃ XOẮN

Convolution Encoder

Bài tập 2.1

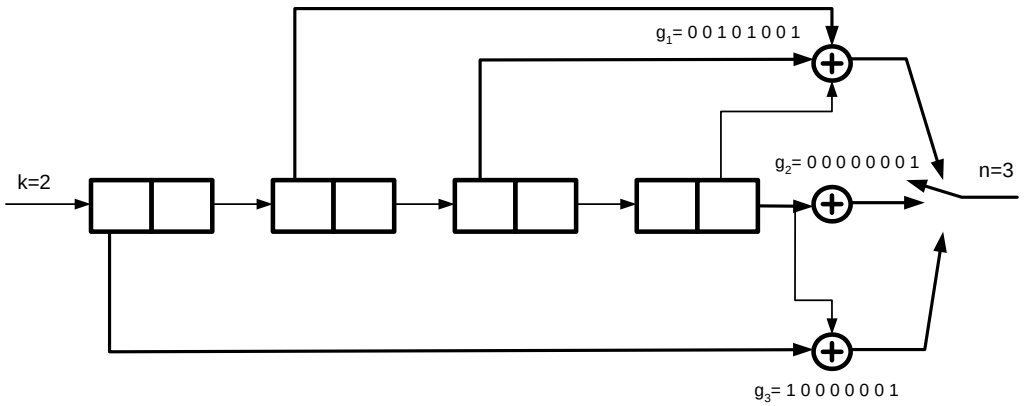




Hình PL3.3 Sơ đồ, sơ đồ trạng thái và sơ đồ lưới của bộ lập mã xoắn

Bài tập 2.2

Sơ đồ khối bộ lập mã xoắn có k=2 n=3 và K=4 được cho ở hình PL3.4



Hình PL3.4: Sơ đồ bộ lập mã xoắn với k=2,n=3,M=4,K=4

Xác định đầu ra bộ mã hoã xoắn hình PL3.4 khi chuỗi tin vào là

m=[1 0 0 1 1 1 0 0 1 1 0 0 0 0 1 1 1]

LỜI GIẢI

❖ Phân tích bài toán

Tỷ lệ mã r =2/3. Lưu ý rằng 3 bit ra bộ lập mã *không những* phụ thuộc vào 2 bit tin được nạp vào bộ lập mã *mà còn* phụ thuộc vào nội dung của ba tầng đầu tiên (6 bit) của bộ lập mã. Nội dung của tầng cuối cùng không ảnh hưởng đến đầu ra vì chúng bị rời khỏi bộ lập mã mỗi khi 2 bit tin được nạp vào bộ lập mã.

Ma trận tạo mã đặc trưng cho bộ lập mã được xác định bởi

$$G = \begin{bmatrix} g_1 \\ g_2 \\ : \\ g_{n-1} \\ g_n \end{bmatrix}_{n\times(Kk)}$$

$$\text{Thành phần thứ i của } g_j = \begin{cases} 1, \text{ nếu thành phần thứ i của bộ ghi dịch dược nối} \\ \quad \text{với bộ kết hợp tương ứng với ký hiệu thứ j của dầu ra} \\ 0, \text{ nếu khác} \end{cases}$$

$$\text{với} \begin{cases} 1 \le i \le Kk \\ 1 \le j \le n \end{cases}$$

Theo đó cụ thể đối với hình PL3.4 nhận được

$$g_1 = [0 \quad 0 \quad 1 \quad 0 \quad 1 \quad 0 \quad 0 \quad 1]$$

$$g_2 = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1]$$

$$g_3 = [1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1]$$

Nói chung nó là ma trận **n×Kk** $\Rightarrow$ với hình PL3.4 nhận được.

$$G = \begin{bmatrix} 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{3\times 8}$$

❖ Xác định độn các bit không

Đôn loai 1: Do độ dài bản tin ban đầu bài cho là 17 không phải là bội số của k=2, nên độn thêm một bit 0 là đủ, kết quả độ dài chuỗi tin là L_m = 18.

$$m = \Big[\ \underbrace{\overleftarrow{\text{Direction of masegae}}}\ \underbrace{\underbrace{1001110011000111}_{17\text{ bit; k=2}}\ \underbrace{0}_{\text{Để độ dài bản tin là số nguyên lần k}}}_{\text{Độ dài là số nguyên lần k}}\ \Big]$$

Đôn loai 2: Để đưa bộ lập mã về trạng thái khởi đầu toàn không, phải độn (K-1)k số không. Vì vậy chuỗi bit vào bộ lập mã khi này là.

$$m = \Big[\ \underbrace{\underbrace{000000}_{\text{Loại 2}}\ \underbrace{\overleftarrow{\text{hướng truyền}}\ 1\,0\,0\,1\,1\,1\,0\,0\,1\,1\,0\,0\,0\,1\,1\,1}_{\text{Độ dài bản tin ban dầu}}\ \underbrace{0}_{\text{Loại 1}}\ \underbrace{000000}_{\text{Loại 2}}}_{\text{Hai loại dộn số 0: Loại 1 để độ dài là số nguyên lần k (k=2)}\ \text{Loại 2 dộn (K-1)k số không = 6 để dưa bộ lập mã về trạng thái toàn không}}\ \Big]$$

❖ Xác định ma trận tạo mã G

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

n =3, K=4, L_m =18 độ dài chuỗi bit ra sẽ là.

$$\left(\frac{L_m}{k} + K - 1\right) \times n = \left(\frac{18}{2} + 4 - 1\right) \times 3 = 36$$

❖ Kết quả chạy chương trình Matlab

output=cnv_encd([0 0 1 0 1 0 0 1;0 0 0 0 0 0 0 1;1 0 0 0 0 0 0 1],2,[1 0 0 1 1 1 0 0 1 1 0 0 0 0 1 1 1])

output = 0 0 0 0 0110111110101110011010010011111 1

❖ Chương trình Matlab

```
function output = cnv_encd(g,k0,iput)
%kiem tra de biet neu don cac bit 0 them la can thiet.
k0=input(' Nhap k cho bo ma hoa= ');
g=input('   Nhap ma tran G cho bo ma hoa = ');
iput=input('Nhap chuoi bit thong tin vao bo ma hoa= ');
```

```
if rem(length(iput),k0)>0
  iput=[iput,zeros(size(1:k0-rem(length(iput),k0)))];
end
n=length(iput)/k0;
% kiem tra kich thuoc Matran G.
if rem(size(g,2),k0)>0
  error('loi, kich thuoc G la sai ');
end
% xac dinh l va n0,
l=size(g,2)/k0;
n0=size(g,1);
% cong cac so 0 phu
u=[zeros(size(1:(l-1)*k0)),iput,zeros(size(1:(l-1)*k0))];
% tao uu, ma tran ma cac cot cua no la noi dung cua conv
% bo ma hoa tai cac Cky clock khac nhau.
u1=u(l*k0:-1:1);
for i=1:n+l-2
  u1=[u1,u((i+l)*k0:-1:i*k0+1)];
end
uu=reshape(u1,l*k0,n+l-1);
% xac dinh dau ra
output=reshape(rem(g*uu,2),1,n0*(l+n-1));
%output=cnv_encd(g,k0,iput);
```

Bài tập 2.3 (Thí du 3.7):

Tìm các bit ra của bộ lập mã được cho ở hình 3.10a khi chuỗi bit vào có L=2 và một bit đuôi (độn) như sau:

$$m^{(1)} = (110)$$
$$m^{(2)} = (010)$$

❖ Chuỗi tạo mã

Để sử dụng (3.61) cần viết lại các dữ liệu vào ở dạng sau

$$m^{(1)} = (110)$$
$$m^{(2)} = m^{(1)} + m^{(2)} = (100)$$
$$m^{(3)} = (010)$$

Vậy có thể biểu diễn các vector bản tin tại các thời điểm i-p=0,1,2 như sau

$$m_0 = (110)$$
$$m_1 = (101)$$
$$m_2 = (000)$$

Trong trường hợp này có ba đầu vào nên q=1,2,3; một bộ nhớ và đầu vào bộ lập mã được nối đến các nhánh cộng nên p=0,1; ba đầu ra nên j=1,2,3.

Đối với đầu vào thứ nhất q=1 có được đáp ứng của ba nhánh ra như sau:

$$g_1^{(1)} = (0,1)$$
$$g_2^{(1)} = (0,1)$$
$$g_3^{(1)} = (0,0)$$

Đối với đầu vào q=2

$$g_1^{(2)} = (1,0)$$
$$g_2^{(2)} = (0,0)$$
$$g_3^{(2)} = (0,1)$$

Tương tự đối với đầu vào q=3

$$g_1^{(3)} = (0,1)$$
$$g_2^{(2)} = (1,0)$$
$$g_3^{(3)} = (0,1)$$

Từ các đáp ứng trên viết các ma trận kết nối như sau:

$$G_0 = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \qquad G_1 = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

Sử dụng ptr(3.61) ta được

$$c_0 = m_0.G_0 = (101)$$
$$c_1 = m_1.G_0 + m_0.G_1 = (100)$$
$$c_2 = m_2.G_0 + m_1.G_1 = (011)$$

Ghép chung các bit ở ba thời điểm trên ta được các bit đầu ra:

$$c = (101 \quad 100 \quad 011)$$

❖ Đa thức tạo mã

Để tính toán đầu ra bằng đa thức tạo mã trước hết cần phải biểu diễn các bit ở đầu vào thứ q=1,2,3 ở dạng các đa thức bản tin như sau:

$$m^{(1)}(x) = 1 + x$$
$$m^{(2)}(x) = m^{(1)}(x) + m^{(2)}(x) = 1$$
$$m^{(3)}(x) = m^{(2)}(x) = x$$

Đối với đầu ra j =1 ta được các đa thức tạo mã như sau:

$$g_1^{(1)}(x) = x$$
$$g_1^{(2)}(x) = 1$$
$$g_1^{(3)}(x) = x$$

Đối với đầu ra j =2 ta được các đa thức tạo mã như sau:

$$g_2^{(1)}(x) = x$$
$$g_2^{(2)}(x) = 0$$
$$g_2^{(3)}(x) = 1$$

Đối với đầu ra j =3 ta được các đa thức tạo mã như sau:

$$g_3^{(1)}(x) = 0$$
$$g_3^{(2)}(x) = 1$$
$$g_3^{(3)}(x) = x$$

Sử dụng (3.67) để tính chuỗi bit ra của nhánh j

Với j=1 ta được

$$\underbrace{\mathbf{c}_j(x) = \sum_{q=1}^{k} \mathbf{m}^{(q)}(x).\mathbf{g}_j^{(q)}(x)}_{\substack{j=1;k=3 \\ \Downarrow}}$$

$$c_1(x) = \sum_{q=1}^{3} m^{(q)}(x).g_1^{(q)}(x)$$
$$= m^{(1)}(x).g_1^{(1)}(x) + m^{(2)}(x).g_1^{(2)}(x) + m^{(3)}(x).g_1^{(3)}(x)$$
$$= (1+x)x + 1.1 + x.x = 1 + x$$

$\Rightarrow$ Vậy $\mathbf{c_1=(110)}$

Với j=2 ta được

$$\underbrace{\mathbf{c}_j(x) = \sum_{q=1}^{k} \mathbf{m}^{(q)}(x).\mathbf{g}_j^{(q)}(x)}_{\substack{j=2;k=3 \\ \Downarrow}}$$

$$c_2(x) = \sum_{q=1}^{3} m^{(q)}(x).g_2^{(q)}(x)$$
$$= m^{(1)}(x).g_2^{(1)}(x) + m^{(2)}(x).g_2^{(2)}(x) + m^{(3)}(x).g_2^{(3)}(x)$$
$$= (1+x)x + 1.0 + x.1 = x^2$$

$\Rightarrow$ Vậy $\mathbf{c_2=(001)}$

Với j =3 ta được:

$$\underbrace{\mathbf{c}_j(x) = \sum_{q=1}^{k} \mathbf{m}^{(q)}(x).\mathbf{g}_j^{(q)}(x)}_{\substack{j=3;k=3 \\ \Downarrow}}$$

$$c_3(x) = \sum_{q=1}^{3} m^{(q)}(x).g_3^{(q)}(x)$$
$$= m^{(1)}(x).g_3^{(1)}(x) + m^{(2)}(x).g_3^{(2)}(x) + m^{(3)}(x).g_3^{(3)}(x)$$
$$= (1+x).0 + 1.1 + x.x = 1 + x^2$$

$\Rightarrow$ Vậy $\mathbf{c_3=(101)}$

Ghép các bit ở đầu ra nói trên ta được chuỗi bit đầu ra bộ lập mã

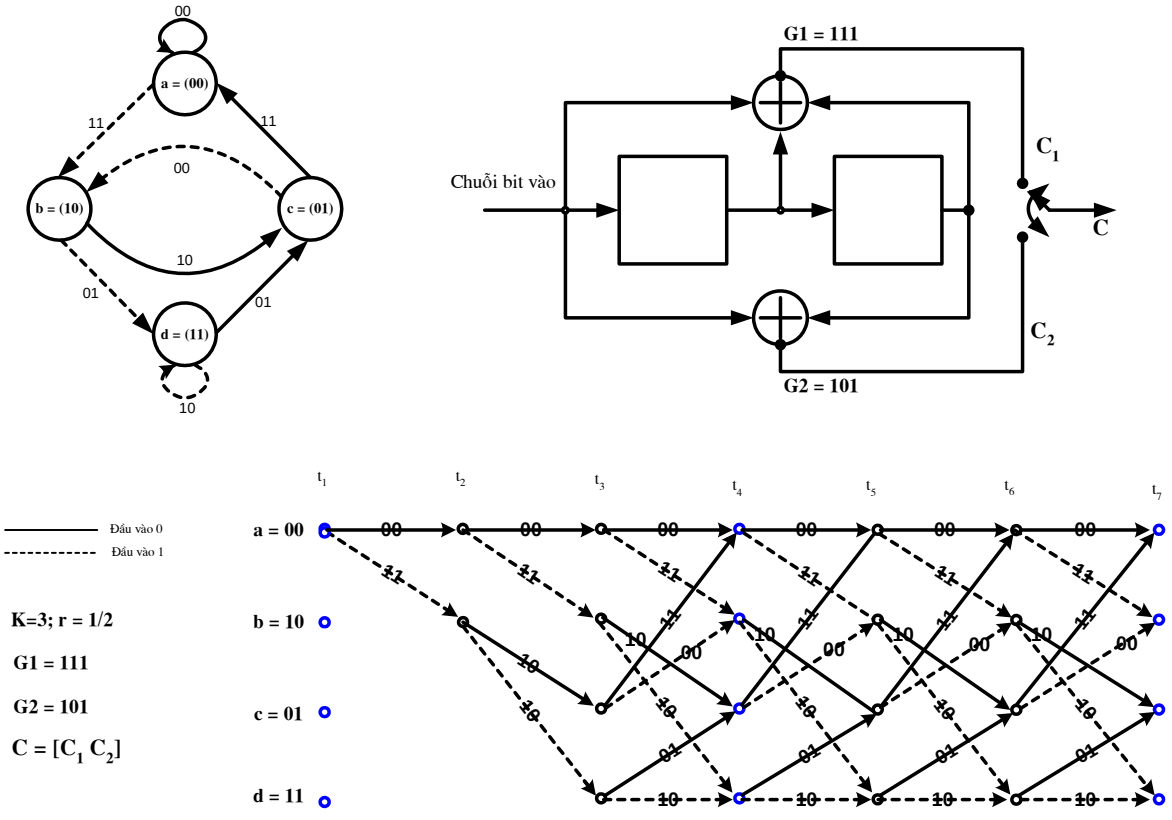
c=(101 101 011)

PHỤ LỤC 2.3

GIẢI MÃ XOẮN

Bài tập 2.4 (Thí du 3.7):

Xét giải mã cho mã xoắn được cho bởi hình 2.2.5. Theo đó có k_0=1, n_0 = 2 và L=3 theo đó có 4 trạng thái có thể có được thể hiện bởi biểu đồ trạng thái và biểu đồ lưới tương ứng.



Hình PL3.5. Sơ đồ và biểu đồ chuyển đổi trạng thái, biểu đồ lưới của bộ mã hóa r=1/2, K=3

Chuỗi tin đầu vào bộ lập mã là

$$m = [1 \quad 0 \quad 1 \quad 0 \quad 1 \quad 1]$$

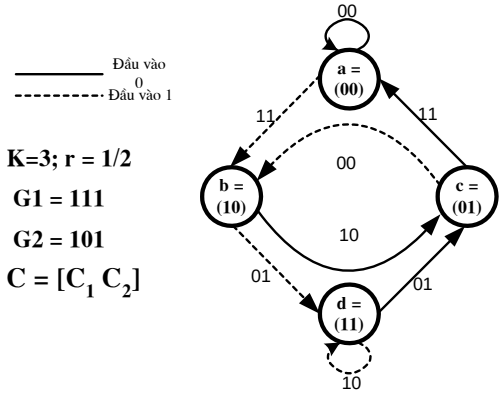
Chuỗi vector phát (từ mã phát) được cho bởi.

$$C = [1 \quad 1 \quad 1 \quad 0 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1]$$

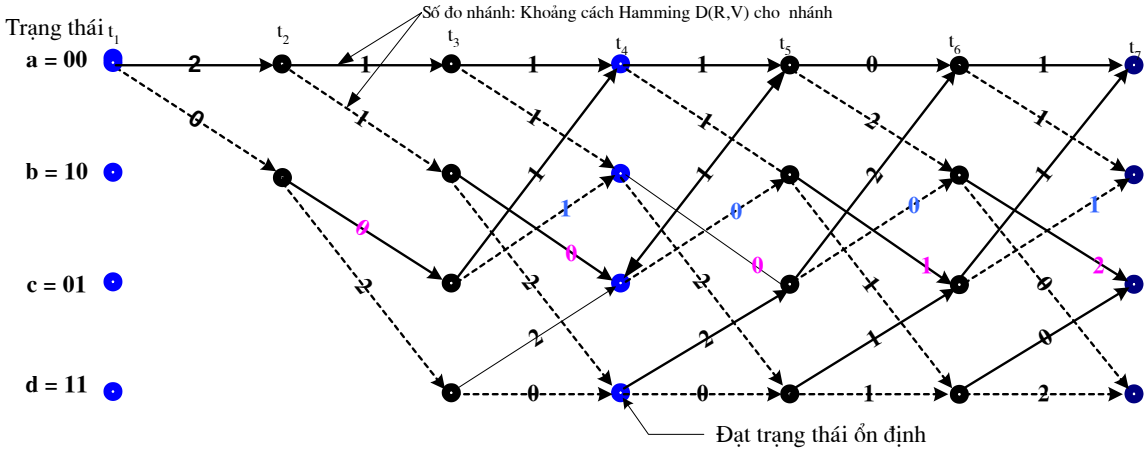
Giả sử chuỗi thu là

$$V = \left[1 \;\; 1 \;\; 1 \;\; 0 \;\; \underbrace{1}_{\text{Error}} \;\; 0 \;\; 1 \;\; 0 \;\; 0 \;\; 0 \;\; 0 \;\; 1 \right]$$

Theo đó ta có biểu đồ lưới được cho ở hình PL3.6.



Bản tin vào:	**m**	**1**	**0**	**1**	**0**	**1**	**1**
Chuỗi phát:	**C**	**11**	**10**	**00**	**10**	**00**	**01**
Chuỗi thu:	**V**	**11**	**10**	**10**	**10**	**00**	**01**



Hình PL3.6. Biểu đồ lưới với các số đo nhánh được cho bởi khoảng cách Hamming dựa vào Vector thu V=(11010100001) cho bộ mã hoá hình PL3.5 có các thông số r=1/2,K=3)

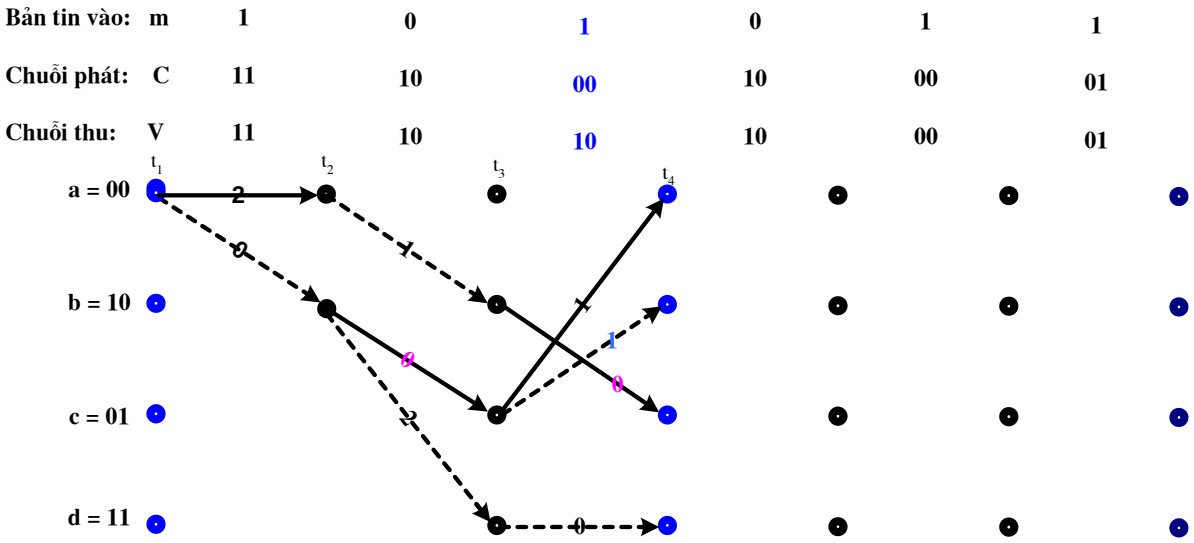
Với mỗi nhánh biểu đồ lưới ở hình PL3.6 có một số đo nhánh được xác định bởi khoảng cách Hamming giữa n bit trên mỗi nhánh và n bit của chuỗi bit thu V. Ví dụ: hai bit thu đầu tiên là 11 tương ứng có khoảng cách Hamming là 2 và 0 so với 2 bit tương ứnh với sự chuyển dịch trạng thái từ a⇒a và a⇒b. Tại thời điểm t = t_4 đạt được trạng thái ổn định của lưới và trạng thái khởi đầu của bộ lập mã là a=(00). Nếu hai đường dẫn nào đó cùng hội nhập vào một nut trạng thái, thì đường dẫn có khoảng cách Hamming cực tiểu so với chuỗi bit thu tại thời điểm đó được giữ lại còn đường dẫn kia bị loại bỏ. Tại mỗi thời điểm t = t_i có 2^{K-1} nut trạng thái, trong đó K là độ dài hạn chế của bộ lập mã. Quá trình giải mã được thực hiện như sau:

- **Tại thời điểm t = t_4**

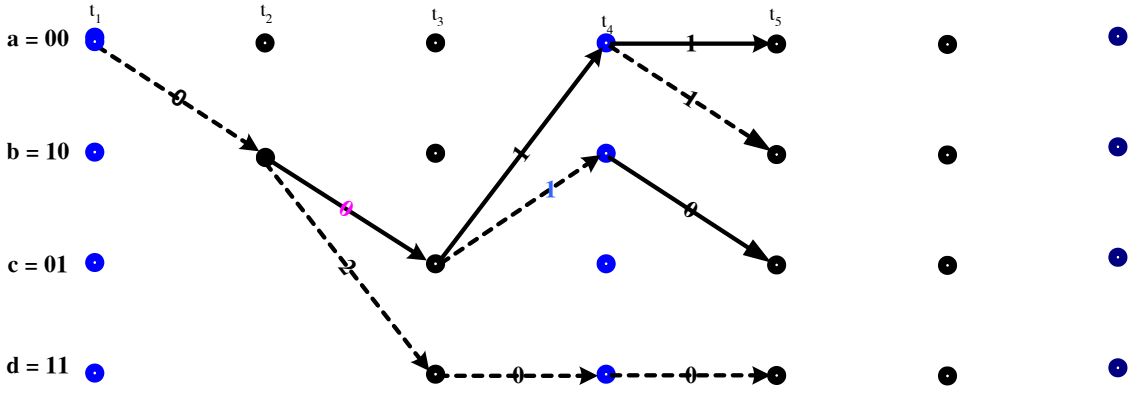
Các đường dẫn sống sót ở mỗi trạng thái tại thời điểm t = t_4 được cho ở *hình PL3.7 (a)*. Đường dẫn sống sót ở trạng thái a tại thời điểm t=t_4 là đường dẫn đi qua các trạng thái a⇒b⇒c⇒a với tổng khoảng cách Hamming 0+0+1=1, được chọn so với đường dẫn đi qua các trạng thái a⇒a⇒a⇒a có tổng khoảng cách Hamming là 2+1+1=4. Tại trạng thái b (nut thứ hai) tại thời điểm t=t_4, đường dẫn sống sót a⇒b⇒c⇒b có tổng khoảng cách Hamming bằng 1 và loại bỏ đường dẫn a⇒a⇒a⇒b có tổng khoảng cách Hamming là 4. Lưu ý rằng, khi này vẫn chưa quyết định được bít vào đầu tiên.

➢ **Tại thời điểm t=t_5**

Các đường dẫn sống sót tại mỗi trạng thái được cho ở *hình PL3.7(b)* được xác định theo các khoảng cách Hamming mỗi khi chuyển dịch trạng thái. Do tất cả các đường dẫn sống sốt đều có sự chuyển dịch trạng thài từ a⇒b (từ t_1 đến t_2) nên quyết định bit đầu vào là 1 ⇒ vì vậy khoảng thời gian trễ giải mã từ t_1 đến t_5 đặc trưng cho cặp sơ đồ lập mã và giải mã này.



a) Các đường dẫn sống sót tại thời điểm t = t_4



b) Các đường dẫn sống sót tại thời điểm t = t_5

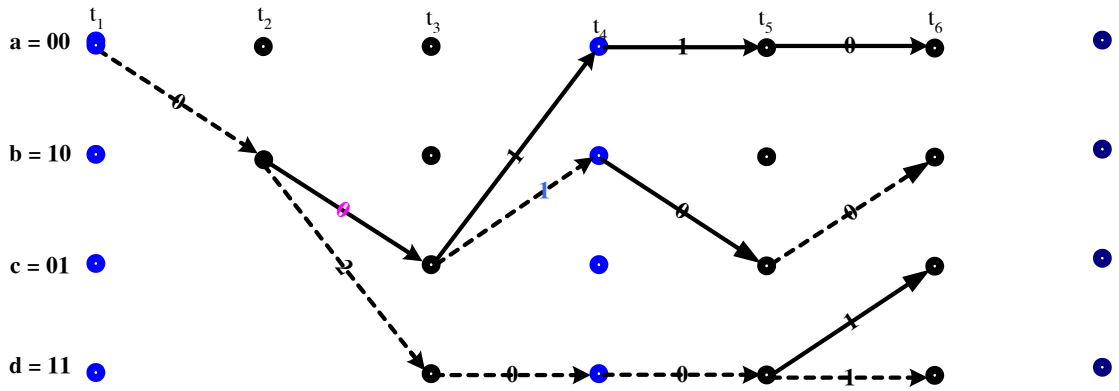
Hình PL3.7 Các đường dẫn sống sót

➢**Tại thời điểm t=t_6**

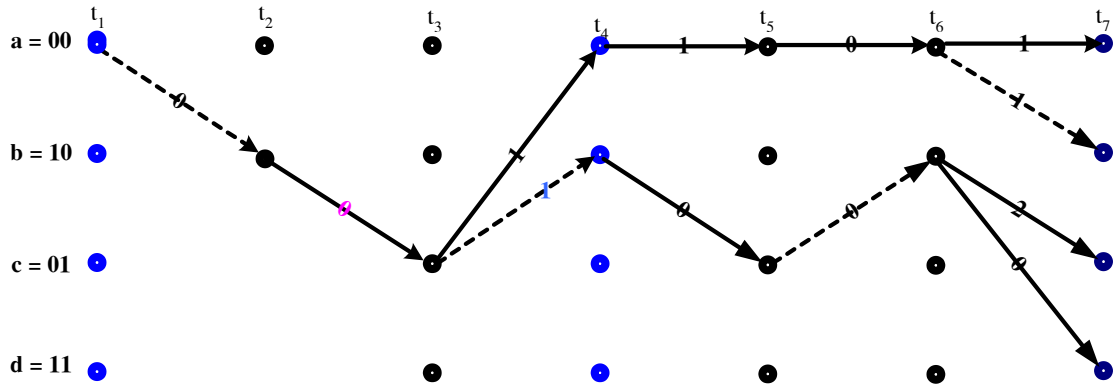
Các đường dẫn sống sót được cho ở *hình PL3.7 (c)* tại nut trạng thái c và nut d đồng xác suất (P=0,5) vì các khoảng cách Hamming đều bằng nhau.

➢**Tại thời điểm t=t_7**

Thực hiện tương tự nhận được các đường dẫn sống sót như được cho ở *hình 2.2.7(d)*



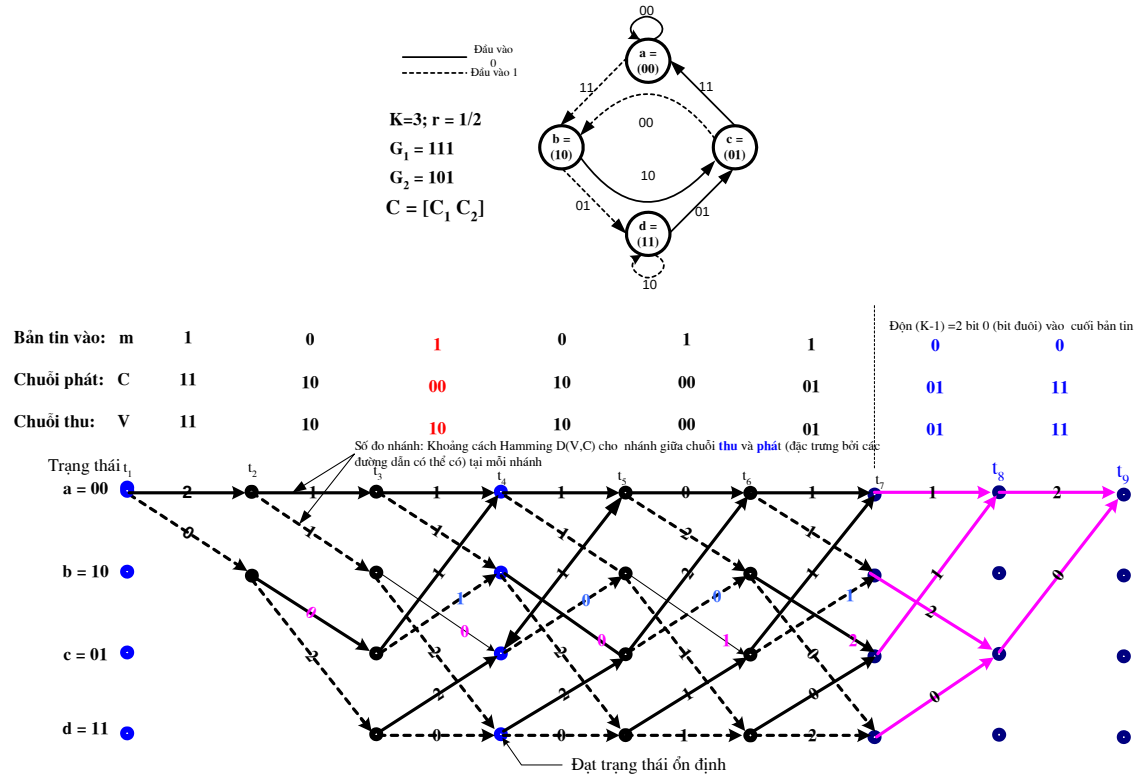
c) Các đường dẫn sống sót tại thời điểm t = t_6



d) Các đường dẫn sống sót tại thời điểm t = t_7

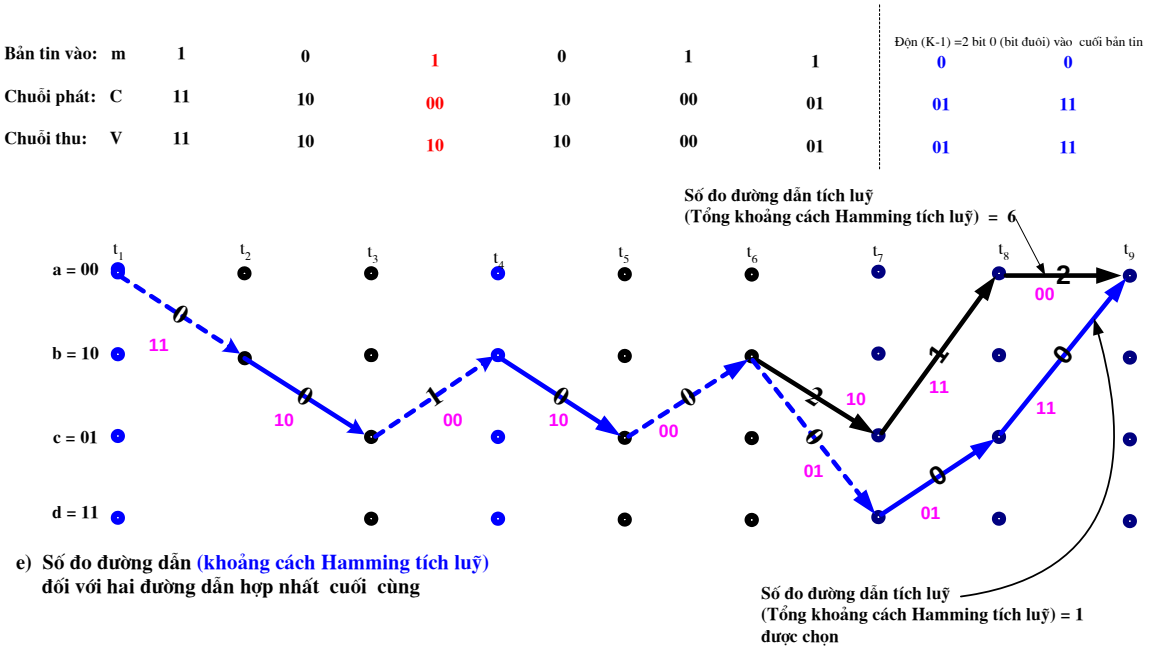
Hình PL3.7 Các đường dẫn sống sót

Thấy rõ, thuật toán Viterbi yêu cầu phải lưu trữ đường dẫn khoảng cách cực tiểu tại mỗi nut trạng thái cũng như số đo đường dẫn của nó, khoảng cách cực tiểu so với chuỗi thu. Hơn nữa, tại mỗi bước cần phải so sánh giữa hai đường dẫn cùng hội nhập vào một nut trạng thái, theo đó cần phải bốn lần so sánh (đối với ví dụ này). Để tránh dùng quá nhiều bộ nhớ ở phía phát thực hiện phân đoạn chuỗi bit và đưa thêm (K-1)k =2 số 0 (các bit đuôi) vào các phân đoạn để đưa trạng thái cuối cùng về trạng thái a = (00) ⇒ đường dẫn sống sót cuối cùng (kết quả) là đường dẫn kết thúc tại node trạng thái toàn không a=(00). Như vậy việc độn thêm (K-1)k số 0 vào cuối chuỗi bản tin dẫn đến đã ép các nut trạng thái có thể có về trạng thái toàn không. Cụ thể trong ví dụ này các trạng thái a, b, c, d, chuyển về trạng thái a như sau: a⇒a⇒a; b⇒c⇒a; c⇒a⇒a; d⇒c⇒a;



Hình PL3.8. Biểu đồ lưới cho thấy sự hội tụ các trạng thái về trạng thái a = (00).

Như vậy đến thời điểm t_9, mới nhận được đường dẫn tối ưu. Kết quả ta được đường dẫn tối ưu la đường dẫn mà có tổng khoảng cách Hamming tại t_9 trạng thái a nhỏ nhất.



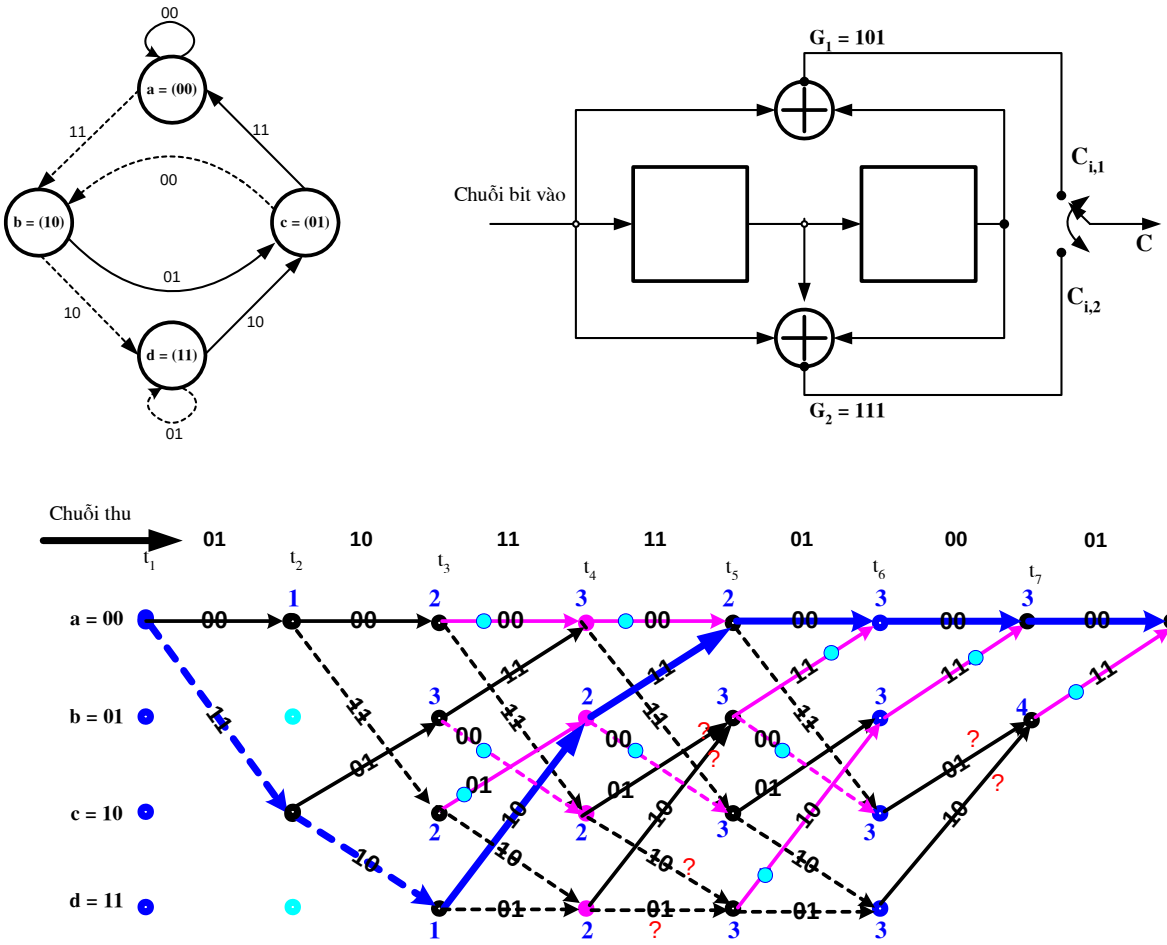
Kết quả: $C^{(m')} = \left[1\ 0\ 1\ 0\ 1\ 1\ \underbrace{0\ 0}_{\text{Tail bit}}\right]$

Hình PL3.9. Đường dẫn tối ưu có khoảng cách Hamming nhỏ nhất được chọn.

Bài tập 2.5

Sơ đồ lập mã được cho ở hình PL3.10. Hãy dùng phương pháp giải mã Viterbi theo quyết định cứng để giải mã chuỗi thu sau

V = 0 1 1 0 1 1 1 1 0 1 0 0 0 1



Hình PL3.10. Sơ đồ khối, biểu đồ trạng thái và biểu đồ lưới

LỜI GIẢI

❖ Các thông số đặc trưng

- ✓ M=3; K=3; n=2; k=1
- ✓ Độ dài chuỗi thu V = 14 ⇒ (L+K-1) =7 ⇒ phải vẽ lưới có độ sâu là 7 được cho ở hình PL3.10 ⇒ chuỗi bit tin phát la L=5

Cũng cần lưu ý rằng do chuỗi bit tin đầu vào đã được độn (K-1)k bit "0" = 2 bit 0, vì thế hai tầng cuối cùng của lưới chỉ cần vẽ các nhánh tương ứng với các đầu vào toàn

bit không, nghĩa là độ dài thực tế của chuỗi tin vào là 5, sau khi độn 2 bit 0 được tăng lên thành 7. Biểu đồ lưới cho trường hợp này được cho ở *hình PL3.10*.

❖ Chuỗi thu V cũng được cho ở hình PL3.10

➢ ***Giải mã quyết định cứng***

- ✓ Lưu ý rằng trong quá trình vẽ lưới ở hai tầng (*stage)* cuối cùng, chỉ xét các đầu vào *zeros* tới bộ lập mã. (hai tầng cuối cùng, không có các đường nét đứt tương ứng với các đầu vào bit 1).
- ✓ Số đo trạng thái toàn không ban đầu được đặt là 0, số đo tầng kế tiếp được tính. Trong bước này thì chỉ có một nhánh (Branch) đi vào mỗi trạng thái (State), vì vậy sẽ không có sự so sánh và các số đo (metrics số đo là khoảng cách Hamming giữa một phần chuỗi thu và các nhánh của lưới) được cộng với số đo của trạng thái state trước đó.
- ✓ Tầng tiếp theo, không có sự so sánh.
- ✓ Trong tầng thứ 4, vì lần đầu tiên có hai nhánh đều đi vào mỗi trạng thái *state*, nghĩa phải thực hiện so sánh và chọn đường sống sót. Từ hai nhánh đi vào mỗi trạng thái *state*, nhánh tương ứng với số đo tích luỹ tổng cuối cùng nhỏ nhất được giữ lại làm đường dẫn sóng sót, đường còn lại bị xoá (được đánh dấu bởi một dấu x trên lưới).
- ✓ Nếu ở một tầng nào đó, hai đường dẫn cùng cho kết quả số đo bằng nhau, thì chúng đều có thể là đường đường dẫn sống sót, những trường hợp như vậy phải được đánh dấu bởi dấu hỏi '?' trong biểu đồ lưới.
- ✓ Thủ tục trên được diễn ra cho đến trạng thái toàn không cuối cùng của lưới; sau đó khởi đầu từ trạng thái đó (trạng thái toàn không), *di chuyển dịch dọc theo các đường dẫn sống sót tới trạng thái toàn không khởi đầu (Initial)*. Đường dẫn này được ký hiệu bằng một đường dẫn đậm thông qua lưới, là ***đường dẫn tối ưu là đường dẫn có khoảng cách Hamming cực tiểu so với chuỗi thu V***.
- ✓ Chuỗi bít đầu vào tương ứng đường dẫn này là 1100000, trong đó hai bit cuối cùng là hai bit không mang thông tin nhưng được cộng thêm vào để đưa bộ lập mã trở về trạng thái toàn không. Vì vậy chuối bit thông tin là 11000.
- ✓ Từ mã tương ứng với đường dẫn được chọn là 11101011000000, là từ mã có khoảng cách Hamming = 4 so với chuỗi thu. Tất cả các đường dẫn khác thông qua lưới có khoảng cách Hamming so với chuỗi bit thu lớn hơn 4 đều không phải là đường dẫn tối ưu.

➢ ***Giải mã quyết định mềm***

Quy trình tương tự với giải mã quyết định cứng song khoảng cách Euclic bình phương thay thế cho khoảng cách Hamming.