LỜI CẢM ƠN

Sau quá trình học tập và nghiên cứu. em đã hoàn thành khóa luận của mình về " Nghiên cứu mã Turbo" dưới sự hướng dẫn và chỉ bảo tận tình của Thạc sỹ Đoàn Hữu Chức.

Với tình cảm trân trọng. em xin chân thành cảm ơn Thạc sỹ Đoàn Hữu Chức đã hướng dẫn, chỉ bảo em hoàn thành khóa luận. Em xin gửi lời cảm ơn sâu sắc tới các thầy cô trong khoa Điện tử - Viễn thông cùng toàn thể các thầy cô trong trường Đại Học Dân Lập Hải Phòng đã dạy dỗ em trong bốn năm học vừa qua.

Sự tiến bộ trong học tập và nghiên cứu của tôi có sự giúp đỡ và động viên rất lớn của các bạn cùng lớp và người thân. Tôi xin cảm ơn những tình cảm quý báu đó.

Hải Phòng, ngày 09 tháng 07 năm 2009

Hoàng Hữu Hiệp

MỞ ĐẦU

Bộ mã hóa và giải mã Turbo cho chất lượng rất cao và được ứng dụng rộng rãi trong thông tin di động. Nó cho phép tiến gần giới hạn Shannon.

Để đi đến khái niệm về mã Turbo, ta nghiên cứu tới những khái niệm có liên quan là nền tảng để xây dựng nên cấu trúc bộ mã hóa và giải mã. Đó là những khái niệm về mã chập, mã kề,và các khái niệm toán học về xác suất, các quá trình ngẫu nhiên của một thống kê kiểm tra: Xác suất hậu nghiệm, xác suất tiền nghiệm. hàm mật độ xác suất.Và đặc biệt là những khái niệm : Đại số log-hợp lệ(log-likelihood), thông tin ngoại lai,…Thông qua ví dụ về mã nhân chúng ta thấy tác dụng của bộ giải mã SISO.

Sau khi có được những khái niệm cơ bản đó. chúng ta tìm hiểu về cấu trúc bộ mã hóa và giải mã lặp dựa trên thuật toán MAP với bộ giải mã SISO (Soft Input - Soft Output).Tìm hiểu về thuật toán giải mã Turbo. Sau đó là các ứng dụng của mã hóa Turbo trong hệ thống thông tin di động.

Cuối cùng là chương trình mô phỏng việc mã hóa và giải mã Turbo trong hệ thống thông tin di động CDMA 2000 qua đó thấy được chất lượng của mã Turbo và các ứng dụng to lớn của mã Turbo trong đời sống khoa học kỹ thuật.

Nội dung đồ án gồm 5 chương :

- Chương 1 : Mã chập, mã kề.
- Chương 2 : Các khái niệm về mã Turbo.
- Chương 3 : Cấu trúc mã Turbo và bộ giải lặp. Thuật toán giải mã Turbo.
- Chương 4 : Ứng dụng mã Turbo trong thông tin di động.
- Chương 5 : Chương trình mô phỏng mã Turbo trông hệ thống thông tin di động CDMA 2000 và rút ra nhận xét.
- Phục lục mô phỏng bằng Matlap

MỤC LỤC

DANH MỤC CÁC CHỮ VIẾT TẮT

	Product Code	Mã nhân
	Extrinsic Likelihood	Hợp lệ ngoại lai
	Metric	Số đo
	A priori	Thông tin tiền nghiệm
	Extrinsic	Thông tin ngoại lai
	Survivor	Đường tồn tại
3G	Third Generation technology	Công nghệ truyền thông thế hệ thứ 3
4G	Fourth Generation Technology	Công nghệ truyền thông thế hệ thứ 4
APP	A posteriori probability	Xác suất hậu nghiệm
ATM	Asynchronous Transfer Mode	Chế độ truyền không đồng bộ
AWGN	Additive white Gaussian noise	Nhiễu cộng trắng chuẩn
BER	Bit error rate	Tỷ số lỗi bít
Bps	bits per second	Bít trên giây
BPSK	Binary phase shift keying	Khóa dịch pha nhị phân
BSC	Binary symmetric channel	Kênh đối xứng nhị phân
CDMA	Code Division Multiple Access	Đa truy cập phân chia theo mã
CRC	Cyclic Redundancy Code	
DS non – OTD	Direct Spreading – non Orthogonal Transmit Diversity	Đơn sóng mang không sử dụng phân tập phát trực giao
DS OTD	Direct Spreading Orthogonal Transmit Diversity	Đơn sóng mang với phân tập phát trực giao
FEC	Forward Error Correction	Sửa lỗi hướng tới trước

FER	Frame error rate	Tỷ số lỗi khung
GIS	Geographic Information System	Hệ thống thông tin địa lý
GSM	Global System for Mobile Communications	Hệ thống thông tin di động toàn cầu
HCCC	Hybrid Concatenated Convolutional Code	Kết nối hổn hợp các bộ mã tích chập
ISI	Inter-symbol interference	Xuyên nhiễu giữa các ký hiệu
LLR	Log-likelihood ratios	Tỷ số log-hợp lệ
LSB	Least Significant Bit	Bít trọng số thấp nhất.
MAP	Maximum a posteriori	Thuật toán cực đại hậu nghiệm
MC	Multicarrier	Đa sóng mang
MCC	Multimedia Communication	Truyền thông đa phương tiện
ML	Max Log MAP	Khả năng xảy ra lớn nhất
MLSE	Maximum likelihood squence estimation	Chuỗi hợp lệ tối đa
Mp	Multiplexer	Bộ ghép
MPSK	M-ary phase shift keying	Khóa dich pha đa mức
MSB	Most Significant Bit	Bit có giá trị cao nhất
PCCC	Parallel Concatenated Convolutional Code	Kết nối song song các mã tích chập
pdf	probability density function	Hàm mật độ xác suất
QAM	Quadrature Amplitude Modulation	Bộ điều biến biên độ vuông góc
QPSK	Quaternary phase shift Keying	Khóa dịch pha bốn mức
RS	Reed Solonon	Mã tuyến tính

RSC	Recursive systematic convolutional	Mã chập hệ thống hồi quy
SCCC	Serial Concatenated Convolutional Code	Kết nối nối tiếp các mã tích chập
SER	Symbol error rate	Tỷ lệ lỗi ký hiệu
SISO	Soft input, soft output	Lối vào mềm-Lối ra mềm
SNR	Signal-to-noise ratio	Tỷ số tín trên tạp
SOVA	Soft output Viterbi algorithm	Thuật toán Viterbi lối ra mềm
TC	Turbo Code	Mã Turbo
TCM	Trellis coded modulation	Điều chế mã lưới
VA	Viterbi algorithm	Thuật toán Viterbi
VOD	Video-On-Demand	Video theo yêu cầu
WC	Wireless Communication	Truyền thông không giây

Chương 1

MÃ CHẬP, MÃ KỀ

1.1 GIỚI THIỆU

Để đi đến khái niệm về mã Turbo, ta nghiên cứu tới những khái niệm có liên quan là nền tảng để xây dựng nên cấu trúc bộ mã hóa và giải mã. Đó là những khái niệm về mã chập, mã kề.

Với mã khối, chuỗi thông tin được chia đoạn trong từng khối và được mã hoá độc lập với dạng của chuỗi mã như là một dãy kế tiếp của chiều dài các từ mã độc lập cố định. Mã chập thì khác, n bít được bộ mã chập tạo ra tương ứng k bít thông tin phụ thuộc vào k bít dữ liệu và các khung dữ liệu trước đó. Và nó là bộ mã hoá có bộ nhớ.

Mã chập khác xa so với mã khối, trên phương diện về cấu trúc, công cụ phân tích và thiết kế. Đặc tính đại số là quan trọng trong cấu trúc của một bộ mã khối tốt và nâng cao hiệu suất thuật giải của bộ giải mã. Ngược lại, các bộ mã chập tốt hầu như đều được nhận ra qua việc nghiên cứu tính toán toàn diện, và hiệu suất các thuật giải của việc giải mã xuất phát trực tiếp từ bản chất trạng thái chuỗi của các bộ mã chập hơn là từ tính chất đại số của mã.

Trong phần này, ta sẽ bắt đầu tìm hiểu cấu trúc của mã chập,cách biểu diễn mã chập thông qua các giản đồ : hình cây, hình lưới, và trạng thái.

Trong phần tiếp theo của chương ta sẽ đề cập tới mã kề (concatenated codes),

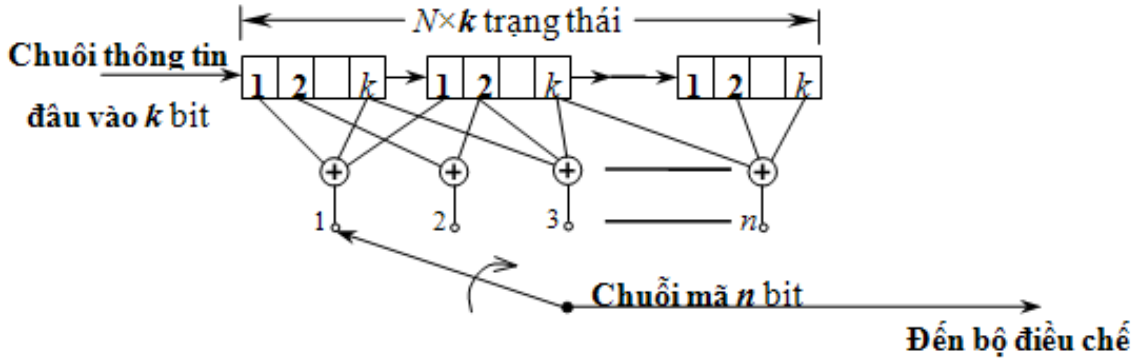
Khái niệm đã được giới thiệu lần đầu tiên bởi Forney (1966) từ đó mà tìm ra nhiều phạm vi rộng rãi trong các ứng dụng.

1.2 CẤU TRÚC MÃ CHẬP VÀ GIẢN ĐỒ BIỂU DIỄN

1.2.1 Cấu trúc mã chập

Mã chập được tạo ra bằng cách cho chuỗi thông tin truyền qua hệ thống các thanh ghi dịch tuyến tính có số trạng thái hữu hạn. Cho số lượng thanh ghi dịch là N, mỗi thanh ghi dịch có k ô nhớ và đầu ra bộ mã chập có n hàm đại số tuyến tính. Tốc độ mã là R = k/n, số ô nhớ của bộ ghi dịch là N×k và tham số N còn gọi là chiều dài ràng buộc(Contraint length) của mã chập (xem hình 1.1)

Giả thiết, bộ mã chập làm việc với các chữ số nhị phân, thì tại mỗi lần dịch sẽ có k bit thông tin đầu vào được dịch vào thanh ghi dịch thứ nhất và tương ứng có k bit thông tin trong thanh ghi dịch cuối cùng được đẩy ra ngoài mà không tham gia vào quá trình tạo chuỗi bit đầu ra. Đầu ra nhận được chuỗi n bit mã từ n bộ cộng môđun-2 (xem hình 1.1). Như vậy, giá trị chuỗi đầu ra kênh không chỉ phụ thuộc vào k bit thông tin đầu vào hiện tại mà còn phụ thuộc vào (N-1)k bit trước đó, cấu thành lên bộ nhớ $v \triangleq (N-1)k$ và được gọi là mã chập (n, k,N).



Hình 1.1 Sơ đồ tổng quát bộ mã chập

Giả sử u là véctơ đầu vào, x là véctơ tương ứng được mã hoá, bây giờ chúng ta mô tả cách tạo ra x từ u. Để mô tả bộ mã hoá chúng ta phải biết sự kết nối giữa thanh ghi đầu vào vào đầu ra hình 1.1. Cách tiếp cận này có thể giúp chúng ta chỉ ra sự tương tự và khác nhau cúng như là với mã khối. Điều này có thể dẫn tới những ký hiệu phức tạp và nhằm nhấn mạnh cấu trúc đại số của mã chập. Điều đó làm giảm đi tính quan tâm cho mục đích giải mã của chúng ta. Do vậy, chúng ta chỉ phác hoạ tiếp cận này một cách sơ lược. Sau đó, mô tả mã hoá sẽ được đưa ra với những quan điểm khác.

Để mô tả bộ mã hoá hình 1.1 chúng ta sử dụng N ma trận bổ sung G_1, G_2 …,G_N bao gồm k hàng và n cột. Ma trận G_i mô tả sự kết nối giữa đoạn thứ i của k ô nhớ trong thanh ghi lối vào với n ô của thanh ghi lối ra. n lối vào của hàng đầu tiên của G_i mô tả kết nối của ô đầu tiên của đoạn thanh ghi đầu vào thứ i với n ô của thanh ghi lối ra. Kết quả là "1" trong G_i nghĩa là có kết nối, là "0" nghĩa là không kết nối. Do đó chúng ta có thể định nghĩa ma trận sinh của mã chập :

$$G_1 = \begin{bmatrix} G_1 & G_2 & \cdots & G_N & & \\ & G_1 & G_2 & \cdots & G_N & \\ & & G_1 & G_2 & \cdots & G_N \\ & & & G_1 & G_2 & \cdots \\ & & & & \cdots & \cdots \; G_N \end{bmatrix} \quad (1.1)$$

Và tất cả các các lối vào khác trong ma trận bằng 0. Do đó nếu lối vào là véctơ u,tương ứng véctơ mã hoá là : $x = uG_\infty$ (1.2)

Bộ mã chập là hệ thống nếu, trong mỗi đoạn của *n* chữ số được tạo, *k* số đầu là mẫu của các chữ số đầu vào tương ứng. Nó có thể xác định rằng điều kiện nà tương đương có các ma trận *k* x *n* theo sau :

$$G_1 = \left[\begin{array}{ccccc|c} 1 & 0 & 0 & \cdots & 0 & \\ 0 & 1 & 0 & \cdots & 0 & \\ 0 & 0 & 1 & \cdots & 0 & P_1 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \\ 0 & 0 & 0 & \cdots & 1 & \end{array}\right] \quad (1.3)$$

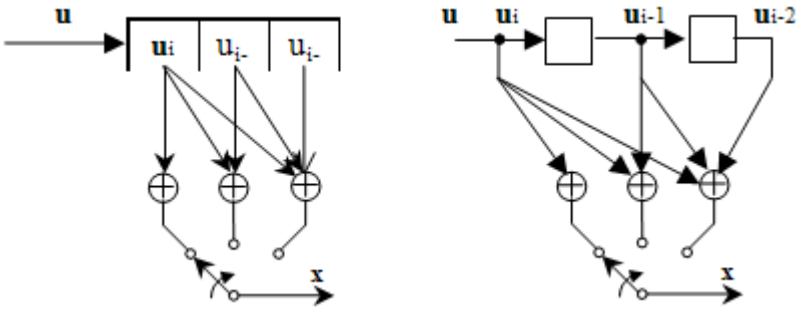
Và

$$G_i = \left[\begin{array}{ccccc|c} 1 & 0 & 0 & \cdots & 0 & \\ 0 & 1 & 0 & \cdots & 0 & \\ 0 & 0 & 1 & \cdots & 0 & P_i \\ \cdots & \cdots & \cdots & \cdots & \cdots & \\ 0 & 0 & 0 & \cdots & 1 & \end{array}\right] \quad (1.4)$$

$$i = 2,3 \ldots N$$

Chúng ta xét một vài ví dụ minh hoạ :

Ví dụ 1: Xét mã chập (3,1,3). Hai giản đồ tương đương cho bộ mã hoá được chỉ ở hình 1.2:



Hình 1.2 : Hai giản đồ tương đương cho bộ mã chập (3,1,3)

Bộ thứ nhất sử dụng thanh ghi với 3 ô nhớ, ngược lại, bộ thứ hai sử dụng 2 ô nhớ, mỗi ô coi như là bộ trễ đơn vị. Lối ra thanh ghi được thay thế bởi bộ tính toán đọc được chuỗi lối ra của 3 bộ cộng. Bộ mã hoá được quy định bởi 3 ma trận bổ sung (trong thực tế là 3 véctơ hàng do k=1)

$$G_1 = [1 \quad 1 \quad 1]$$
$$G_2 = [0 \quad 1 \quad 1]$$
$$G_3 = [0 \quad 0 \quad 1]$$

Do đó, ma trận sinh từ (1.1) là :

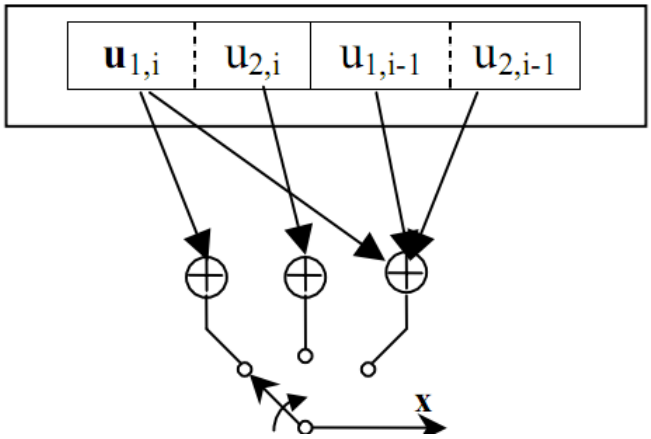
$$G_\infty = \begin{bmatrix} 111 & 011 & 001 & 000 & \cdots & \cdots & \cdots \\ 000 & 111 & 011 & 001 & 000 & \cdots & \cdots \\ 000 & 000 & 111 & 011 & 001 & 000 & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix}$$

Từ (1.2) ta có thể suy ra : Nếu chuỗi thông tin vào u = (11011…) được mã hoá thành chuỗi x=(111100010110100…). Bộ mã hoá là hệ thống. Chú ý rằng chuỗi mã hoá có thể được tạo bằng tổng modul-2 các hàng của G_∞ tương ứng với "1" trong chuỗi thông tin.

Ví dụ 2 : Xét mã (3,2,2). Bộ mã hoá được chỉ trong hình 1.3.Bây giờ mã được định nghĩa thông qua 2 ma trận:

$$G_\infty = \begin{bmatrix} 101 & 001 & 000 & \ldots & \ldots \\ 010 & 001 & 000 & \ldots & \ldots \\ 000 & 101 & 001 & 000 & \ldots \\ 000 & 010 & 001 & 000 & \ldots \\ 000 & 000 & 101 & 001 & 000 \\ 000 & 000 & 010 & 001 & 000 \\ \ldots & \ldots & \ldots & \ldots & \ldots \end{bmatrix}$$

Chuỗi thông tin u = (11011011…) được mã hóa thành chuỗi mã x = (111010100110…)



Hình 1.3 : Bộ mã chập (3,2,2).

Một cách tương tự ta cũng có thể biểu diễn ma trận sinh G = (G_1, G_2,…, G_N), Như vậy ý nghĩa của ma trận sinh là nó chỉ ra nó chỉ ra phải sử dụng các hàm tương ứng nào để tạo ra véc tơ dài *n* mỗi phần tử có một bộ cộng môđun-2, trên mỗi véc tơ có $N \times k$ tham số biểu diễn có hay không các kết nối từ các trạng thái của bộ ghi dịch tới bộ cộng môđun-2 đó. Xét véc tơ thứ *i* (gi, n ≥ i ≥ 1), nếu tham số thứ j của G_i $(L \times k \geq j \geq 1)$ có giá trị "1" thì đầu ra thứ *j* tương ứng trong bộ ghi dịch được kết nối tới bộ cộng môđun-2 thứ *i* và nếu có giá trị "0" thì đầu ra thứ *j* tương ứng trong bộ ghi dịch không được kết nối tới bộ cộng môđun-2 thứ *i*

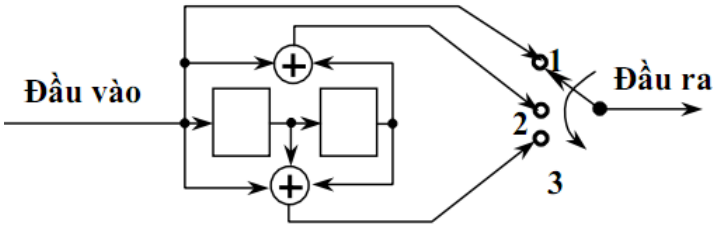
Ví dụ 3: Cho bộ mã chập có chiều dài ràng buộc N = 3, số ô nhớ trong mỗi thanh ghi dịch k = 1, chiều dài chuỗi đầu ra n = 3 tức là mã (3,1,3) và ma trận sinh của mã chập có dạng sau:

$$G = \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_n \end{bmatrix} \Leftrightarrow G \begin{bmatrix} 100 \\ 101 \\ 111 \end{bmatrix} = G(4,5,7) \qquad (1.5)$$

Có thể biểu diễn dưới dạng đa thức sinh là:

$$G(D) = [D^2 \quad 1 + D^2 \quad 1 + D + D^2] \qquad (1.6)$$

Do đó sơ đồ mã chập được biểu diễn như sau :



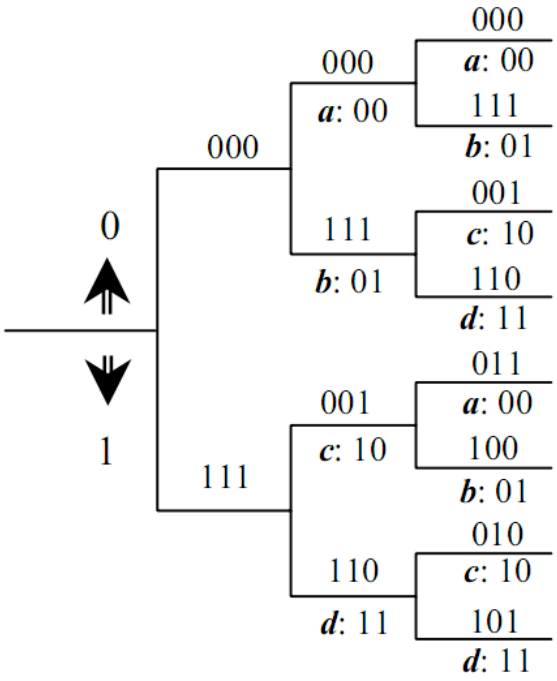
Hình 1.4 : Sơ đồ bộ mã chập với N=3, k=1, n=3 và đa thức sinh (1.6)

1.2.2 Biểu diễn mã chập

Có ba phương pháp để biểu diễn mã chập đó là : sơ đồ lưới, sơ đồ trạng thái và sơ đồ hình cây. Để làm rõ phương pháp này ta tập trung phân tích dựa trên ví dụ 3

** Sơ đồ hình cây :*

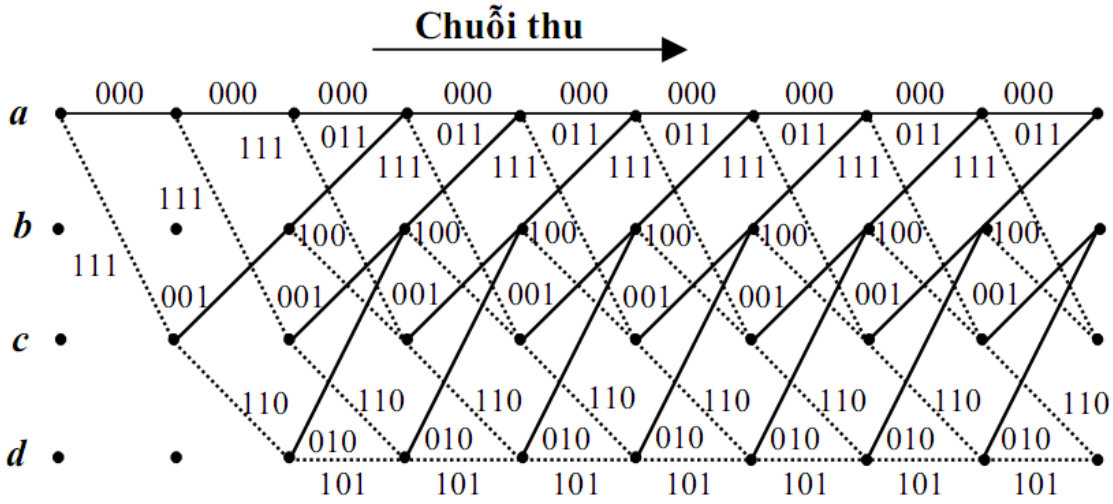
Từ ví dụ 3, giả thiết trạng thái ban đầu của các thanh ghi dịch trong bộ mã đều là trạng thái "toàn 0". Nếu bit vào đầu tiên là bit "0" (k = 1) thì đầu ra ta nhận được chuỗi "000" (n = 3), còn nếu bit vào đầu tiên là bit "1" thì đầu ra ta nhận được chuỗi "111". Nếu bit vào đầu tiên là bit "1" và bit vào tiếp theo là bit "0" thì chuỗi thứ nhất là "111" và chuỗi thứ hai là chuỗi "001". Với cách mã hoá như vậy, ta có thể biểu diễn mã chập theo sơ đồ có dạng hình cây (xem hình 1.5). Từ sơ đồ hình cây ta có thể thực hiện mã hoá bằng cách dựa vào các bit đầu vào và thực hiện lần theo các nhánh của cây, ta sẽ nhận được tuyến mã, từ đó ta nhận được dãy các chuỗi đầu ra.



Hình 1.5 : Sơ đồ hình cây với N=3, k=1,n=3 (ví dụ 3)

**Sơ đồ hình lưới :*

Do đặc tính của bộ mã chập, cấu trúc vòng lặp được thực hiện như sau: chuỗi *n* bit đầu ra phụ thuộc vào chuỗi *k* bit đầu vào hiện hành và *(N-1)* chuỗi đầu vào trước đó hay *(N-1)* × *k* bit đầu vào trước đó. Từ ví dụ 3 ta có chuỗi 3 bit đầu ra phụ thuộc vào 1 bit đầu vào là "1" hoặc "0" và 4 trạng thái có thể có của hai thanh ghi dịch, ký hiệu là a = "00"; b = "01"; c = "10"; d = "11". Nếu ta đặt tên cho mỗi nút trong sơ đồ hình cây (hình 1.5) tương ứng với 4 trạng thái của thanh ghi dịch, ta thấy rằng tại tầng thứ 3 có 2 nút mang nhãn a và 2 nút mang nhãn b, 2 nút mang nhãn c và 2 nút mang nhãn d. Bây giờ ta quan sát tất cả các nhánh bắt nguồn từ 2 nhánh có nhãn giống nhau (trạng thái giống nhau) thì tạo ra chuỗi đầu ra giống nhau, nghĩa là hai nút có nhãn giống nhau thì có thể coi như nhau. Với tính chất đó ta có thể biểu diễn mã chập bằng sơ đồ có dạng hình lưới gọn hơn, trong đó các đường liền nét được ký hiệu cho bit đầu vào là bit "0" và đường đứt nét được ký hiệu cho các bit đầu vào là bit "1" (xem hình 1.6). Ta thấy rằng từ sau tầng thứ hai hoạt động của lưới ổn định, tại mỗi nút có hai đường vào nút và hai đường ra khỏi nút. Trong hai đường đi ra thì một ứng với bit đầu vào là bit "0" và đường còn lại ứng với bit đầu vào là bit "1".



Hình 1.6: Sơ đồ hình lưới bộ mã chập ví dụ 3. Trạng thái ban đầu toàn bằng "0"

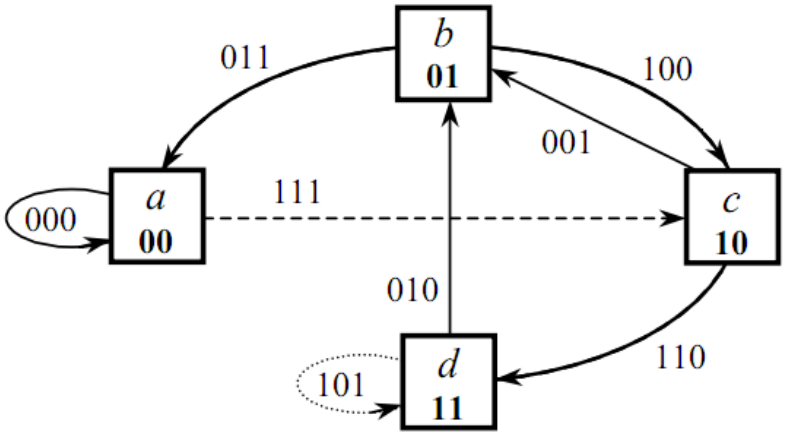
**Sơ đồ trạng thái :*

Sơ đồ trạng thái được thực hiện bằng cách đơn giản sơ đồ 4 trạng thái có thể có của bộ mã (a, b, c và d tương ứng với các trạng thái 00, 01, 10, và 11)và trạng thái chuyển tiếp có thể được tạo ra từ trạng thái này chuyển sang trạng thái khá quá trình chuyển tiếp có thể là:

$$a \xrightarrow{0} a, a \xrightarrow{1} c, b \xrightarrow{0} a, b \xrightarrow{1} c, c \xrightarrow{0} b, c \xrightarrow{1} d, d \xrightarrow{0} b, d \xrightarrow{1} d, \quad (1.7)$$

Ký hiệu $\alpha \rightarrow \beta$ là quá trình chuyển tiếp từ trạng thái α sang trạng thái β với bit đầu vào là bít "1".

Kết quả ta thu được sơ đồ trạng thái trong hình 1.7 như sau:



Hình 1.7: Sơ đồ trạng thái của bộ mã chập trong ví dụ 3.

Từ sơ đồ trạng thái hình 1.7, các đường liền nét được ký hiệu cho bit đầu vào là bit "0" và đường đứt nét được ký hiệu cho các bit đầu vào là bit "1".

So với sơ đồ hình lưới và sơ đồ hình cây thì sơ đồ trạng thái là sơ đồ đơn giản nhất.

1.2.3 Phân bố trong mã chập

Phân bố trọng số của mã chập là một tham số quan trọng để tính chất lượng của nó. Chúng ta định nghĩa Ai là số lượng các chuỗi có trọng số i trong lưới mà nó phân kỳ khỏi tuyến "toàn 0" tại một điểm nào đó và hồi qui lần đầu tiên tại điểm nút sau đó.

Tập hợp :$\{A_{\text{dfree}}, A_{\text{dfree+1}}, A_{\text{dfree+2}},\ldots A_{\text{dfree+i}}\ldots\}$ được gọi là phân bố trọng số của mã chập.

Phân bố trọng số có thể tính bằng cách cải tiến sơ đồ chuyển đổi trạng thái của mã. Sơ đồ trạng thái cải tiến có thể nhận được bằng cách triển khai từ trạng thái ban đầu "toàn 0" là S_0 hay còn gọi là S_{in} cho đến trạng thái kết thúc S_{out} cũng là trạng thái "toàn 0". Mỗi tuyến trong sơ đồ trạng thái được kết nối bắt đầu trạng thái S_{in} và kết thúc về trạng thái S_{out} biểu diễn một chuỗi mã phân kỳ và hồi qui về trạng thái "toàn 0" đúng một lần. Trọng số chuỗi mã A_i biểu diễn số lượng các chuỗi mã phân kỳ từ chuỗi "toàn 0" tại cùng một điểm nút và hồi qui lần đầu tiên tại các nút tiếp theo. Nói cách khác A_i bằng số lượng các tuyến có trọng số i trong sơ đồ chuyển đổi trạng thái mở rộng được nối từ điểm đầu đến điểm cuối.

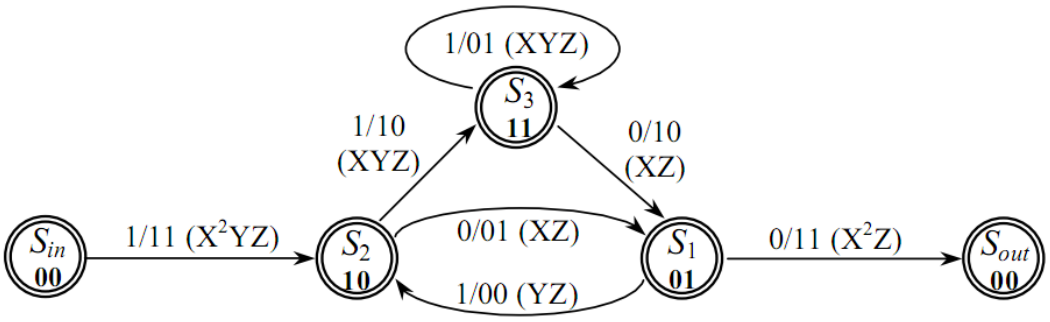
Gọi X là biến vô định liên quan đến trọng số Hamming của chuỗi mã hoá đầu ra i, Y là biến vô định liên quan đến trọng số Hamming của chuỗi thông tin j, và Z là biến vô định liên quan đến từng nhánh. Mỗi nhánh trong sơ đồ chuyển đổi trạng thái được đánh số X^iY^jZ. Sơ đồ chuyển đổi trạng thái được đánh số như trên được gọi là sơ đồ chuyển đổi trạng thái mở rộng. Sơ đồ chuyển đổi trạng thái được đánh số như trên được gọi là sơ đồ chuyển đổi trạng thái mở rộng

Phân bố trọng số có thể nhận được từ hàm truyền đạt của sơ đồ chuyển đổi trạng thái mở rộng. Sơ đồ chuyển đổi trạng thái mở rộng có thể xem là đồ thị đường đi của tín hiệu và hàm truyền đạt có thể nhận được theo qui luật

của Mason. Hàm truyền đạt có thể nhận được từ một tập các phương trình mô tả sự chuyển đổi trạng thái trong sơ đồ chuyển đổi trạng thái mở rộng

Ví dụ về sơ đồ chuyển đổi trạng thái mở rộng

Chúng ta xem xét bộ mã chập (2,1) và sơ đồ chuyển đổi trạng thái tương ứng của nó trong hình 1.8.



Hình 1.8: Sơ đồ trạng thái mở rộng

Ví dụ chuyển đổi từ trạng thái $S_{in} \rightarrow S_2$, ta có trọng số Hamming đầu ra của nhánh đó là 2 khi đầu vào là 1 (tương ứng với 1/11), nên nhánh này được gán là X^2 YZ. Nhánh từ $S_2 \rightarrow S_1$ ta có 0/01 nên nhãn của nó là XZ... Ta có hệ các phương trình mô tả sự chuyển giao trạng thái trong sơ đồ trạng thái mở rộng như sau:

$$S_2 = YZS_1 + X^2YZS_{in} \quad (1.8)$$

$$S_1 = XZS_2 + XZS_3 \quad (1.9)$$

$$S_3 = XYZS_2 + XYZS_3 \quad (1.10)$$

$$S_{out} = X^2ZS_1 \quad (1.11)$$

Giải S_3 từ (1.10), ta có:

$$S_3 = \frac{XYZS_2}{1 - XYZ} \quad (1.12)$$

Thay thế (1.12) vào (1.9), ta có:

$$S_1 = \frac{XZS_2}{1 - XYZ} \quad (1.13)$$

Thay thế (1.13) vào (1.8), ta có:

$$S_2 = \frac{(1 - XYZ)X^2YZS_2}{1 - XYZ - XYZ^2}$$

(1.14)

Ngoài ra, thay thế (1.13) vào (1.11), ta có:

$$S_{out} = \frac{X^3 Z^2 S_2}{1 - XYZ}$$

(1.15)

Cuối cùng, thay thế (1.14) vào (1.15), chúng ta hàm truyền đạt của T(X,Y,Z) là

$$T(X,Y,Z) = \frac{S_{out}}{S_{in}} = \frac{X^5 Y Z^3}{1 - XYZ(1+Z)}$$

(1.16)

Cuối cùng, thay thế (1.2.7) vào (1.2.8), chúng ta hàm truyền đạt của T(X,Y,Z) là :

$$T(X,Y,Z) = X^5YZ^3 + X^6Y^2Z^4 + (X^6Y^2 + X^7Y^3)Z^5 + (2X^7Y^3 + X^8Y^4)Z^6 + (X^7Y^3 + 3X^8Y^4 + X^9Y^5)Z^7 + (3X^8Y^4 + 4X^95 + X^{10}Y^6)Z^8 + \cdots$$

Chú ý rằng, khoảng cách tự do cực tiểu đối với mã này là 5 và số lượng của các từ mã tại khoảng cách này là A5 = 1. Chuỗi thông tin tạo ra từ mã này có trọng số Hamming bằng 1 và chiều dài đoạn khác "0" gồm 3 nhánh. Có một chuỗi thông tin khác có trọng số bằng 2, tạo ra từ mã có trọng số bằng 6, với 4 nhánh khác "0".

Nếu chúng ta bỏ qua chiều dài của nhánh bằng cách đặt Z = 1, hàm truyền đạt sẽ trở thành:

$$T(X,Y) = X^5Y + 2X^6Y^2 + 4X^7Y^3 + 8X^8Y^6 + \cdots \quad (1.17)$$

Ngoài ra, phân bố trọng số Ai có thể nhận được bằng cách đặt Y = 1.

$$T(X) = X^5 + 2X^6 + 4X^7 + 8X^8 + \cdots \quad (1.18)$$

Tổng số của các chuỗi thông tin khác 0 trên tất cả các tuyến có trọng số Hamming j có thể nhận được bằng cách lấy đạo hàm một phần của T(X,Y,Z) theo Y:

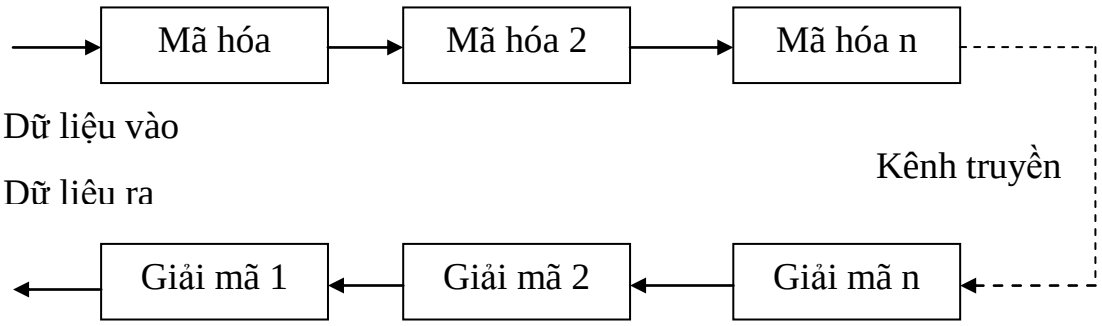
$$\left.\frac{\delta T(X,Y,Z)}{\delta Y}\right|_{Y=1,Z=1} = X^5 + 4X^6 + 12X^7 + \cdots$$

(1.19)

1.3 MÃ KỀ (Concatenated code)

1.3.1 Cấu trúc và nguyên lý

Mã kề được giới thiệu vào năm 1966, với mục đích là tìm ra lớp mã mà xác suất lỗi sẽ giảm theo hàm mũ ở tốc độ bé hơn dung lượng, trong khi việc giải mã phức tạp sẽ tăng về tính chất đại số. Nó đã thúc đẩy các nhà nghiên cứu lý thuết quan tâm, các mã kề đã tạo ra sự chuẩn hoá cho những ứng dụng mà các yểu tố như độ tăng ích mã hoá cao, vầ điều kiện phức tạp là cần thiết. Khái niệm mã kề được giải thích trong hình 1.9, hai hay nhiều bộ mã hóa được sắp xếp thành các tầng dựa trên nguyên tắc rất đơn giản: lối ra của bộ mã hoá đầu tiên (outermost) được đưa tới lối vào của bộ mã hoá thứ hai, và cứ như vậy.



Hình 1.9: Nguyên lý của mã hoá kề

Giả sử bộ mã hoá 1 là mã khối (n_0, k_0), và bộ mã hoá 2 là mã khối (n_i, k_i), Tham số k_i và n_0 phải là bội của nhau. Thông thường n0 lớn hơn ki, bới vậy :

$n_0 = mk_i$ m là số nguyên (1.3.1)

Do đó, từ mã của bộ mã hóa 1 là mốt số nguyên lần so với từ dữ liệu của bộ mã hoá 2. Ký hiệu R_c^0, R_i^0 là tốc độ má hoá của mã 1 và mã 2, khi đó tốc độ toàn bộ của mã kề là

$$R_c = \frac{k_0}{mn_i} = \frac{k_0}{n_0}\frac{n_0}{mn_i} = R_c^0 R_c^i \quad (1.3.2)$$

Như vậy tốc độ toàn bộ của mã kề bằng tích tốc độ của hai mã cấu thành.

Lợi ích chủ yếu của các mã kề là chúng bổ sung cách thức giải mã từng giai đoạn làm mất đi tính phức tạp của việc giải mã (mni, k0) trong tầng của hai mã đơn giản $(n_i.k_i)$ và $n_0.k_0$. Nói cách khác, bộ mã hoá 2 được giải mã đầu tiên, và sau đó là bộ mã 1. Có một vài cách thức thực hiện việc giải mã tầng, cái mà phụ thuộc vào bản chất của bộ giải điều chế và hoạt động của bộ giải mã 1. Lối ra bộ giải điều chế có thể là cả quyết định cứng lẫn quyết định mềm, và bộ mã hoá 2, lần lượt có thể cung cấp những đánh giá cứng hay mềm tới bộ mã hoá 1 của ký hiệu mã 1. Forney (1966) đã chỉ ra cách tốt nhất để hoạt động giải mã tầng yêu cầu đối với bộ mã hoá 2 là đánh giá xác suất hậu nghiệm của ký hiệu mã hoá 1 được cho bởi chuỗi kênh nhận. Thủ tục tối ưu này yêu cầu hoạt động mềm bộ giải điều chế và bộ giải mã 2.

Đơn giản, chiến thuật gần tối ưu mà bộ giải mã 2 chỉ tạo ra những quyết định ký hiệu cứng, bộ mã hoá 1 xem như là tương đương kênh truyền được thiết lập bởi tầng của bộ mã hoá 2, bộ điều chế, kênh truyền, giải điều chế, và bộ giải mã cứng 2. Kênh truyền tương đương này được đặc trưng bởi xác suất lỗi ký hiệu ps - phụ thuộc vảo tỷ số tín trên tạp qua kênh vật lý, giản đồ điều chế, và dung lượng sửa đúng lỗi của bộ mã hoá 2.

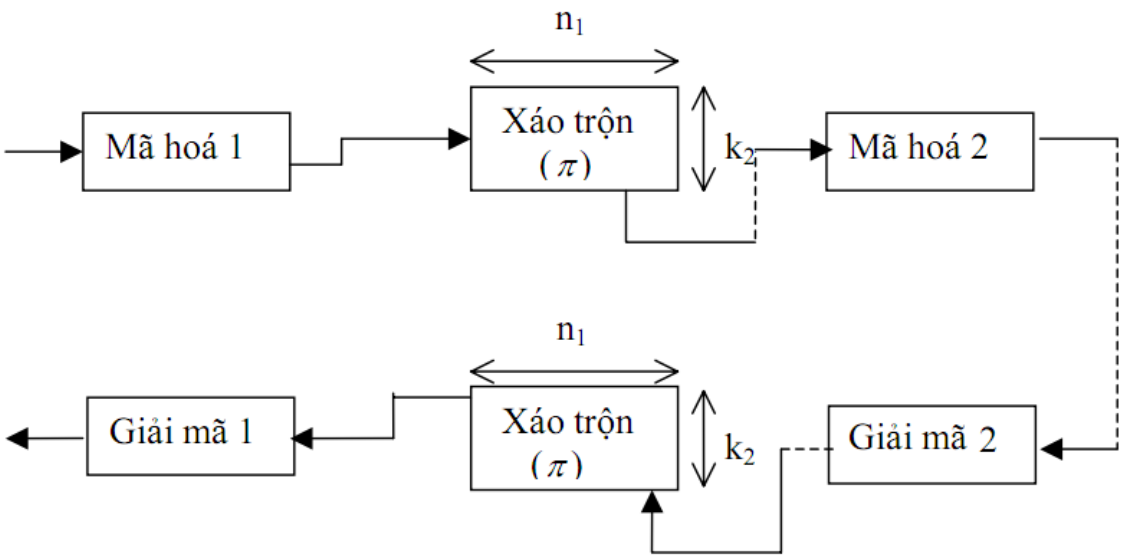
Đối với bộ má hoá (n_0, k_0, t) RS (Reed Solonon) sử dụng K- bít ký hiệu, và giả sử xác suất lỗi bit là p_s ở lối vào bộ giải mã RS, khi đó chúng ta có thể đánh giá xác suất lỗi bít :

$$P_b(e) \leq \frac{2^K - 1}{2^K - 1} \sum_{j-t+1}^{n_0} \frac{j+1}{n_0} \left[\frac{n_0}{j}\right] p_s^j (1-p_s)^{n_0 - 1} \tag{1.3.3}$$

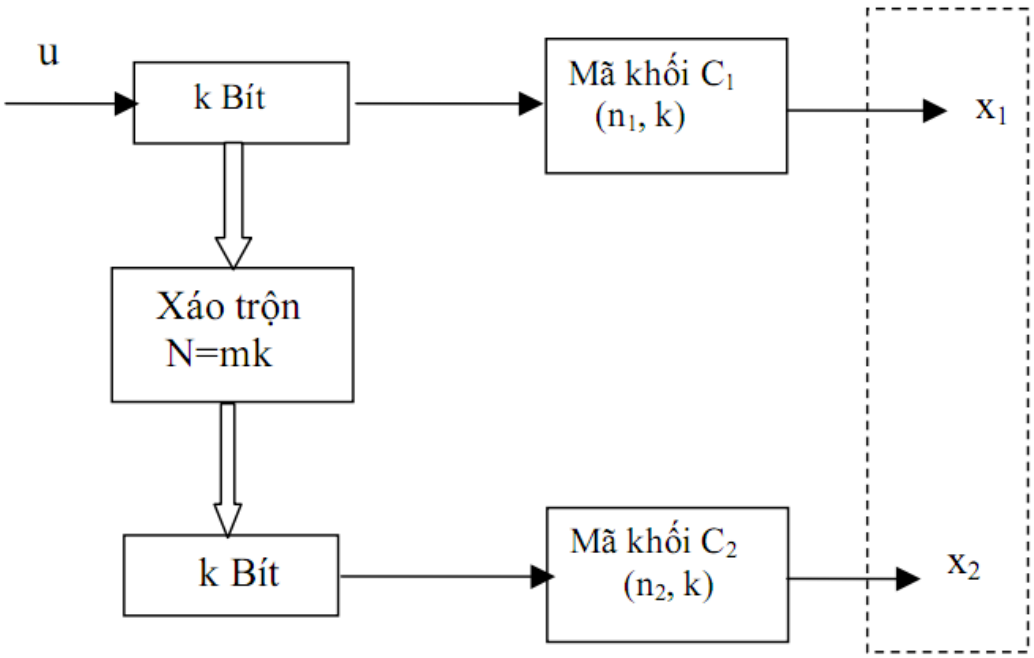
Như vậy, qua khái niệm này ta thấy nó hai điểm bất lợi : Thứ nhất là, lối vào của các bộ giải mã là quyết định cứng, chúng không thể thực hiện giải mã quyết định mềm. Do đó, việc giải mã toàn bộ không thể có hợp lệ tối đa (maximum likelihood). Thứ hai, lỗi giải mã toàn bộ mã hoá trong co xu hướng tăng sự xuất hiện lỗi, mà bộ mã ngoài không thể khắc phục.

Để tránh ít nhất là hai vấn đề trên, người ta đưa ra sơ đồ mã hoá và giải mã như sơ đồ sau :

1.3.2 Sơ đồ mã hóa



Hình 1.10. Mã kề với bộ xáo trôn nối tiếp



Hìn 1.11 Sơ đồ mã kề song song

Ở sơ đồ hình 1.10- mã kề nối tiếp thì bộ mã hoá 1 là mã RS (Reed - Solonon) còn bộ mã hoá 2 là mã chập. TacCũng có thể dùng các bộ mã khối để thay thế các bộ mã hoá trên. Còn ở sơ đồ hình 1.11 - mã kề song song thì thông thường cả hai bộ mã hoá thường là bộ mã khối.

Khi ta thay thế hai bộ mã khối này băng hai mã chập hệ thống đệ quy (Recursive System Convolutional Code - RSC) và thuật toán giải mã lặp và cấu trúc đó gọi là mã Turbo sẽ được đề cập ở chương sau.

Bộ xáo trộn (ký hiệu là π) hay còn gọi là bộ ghép xen là một tiến trình thực hiện hoán vị trật tự sắp xếp của chuỗi gốc theo một quan hệ xác định một - một. Ngược lại, bộ giải xáo trộn thực hiện trả lại chuỗi thu được theo đúng trật tự sắp xếp của chuỗi gốc.

Bộ xáo trộn đóng vai trò rất lớn trong việc nâng cao khả năng sửa lỗi của mã, nó được sử dụng rộng rãi trong các sơ đồ mã kênh khi trên kênh truyền thường xảy ra lỗi cụm, ví dụ kênh pha đinh đa đường, kênh ghi từ …Kỹ thuật xáo trộn được thực hiện ngay giữa khối mã kênh và kênh truyền với mục đích làm thay đổi trật tự sắp xếp của chuỗi đầu vào để tạo ra một chuỗi mới có trật tự sắp xếp khác đi để truyền trên kênh. Tín hiệu đầu thu nhận được cùng với lỗi cụm xảy ra trên kênh được bộ giải xáo trộn sắp xếp lại về đúng trật tự ban đầu, quá trình này đã làm phân tán lỗi cụm ra thành các lỗi đơn hay nói cách khácla lỗi xuất h iện độc lập, ngẫu nhiên với mã, nhờ đó vấn đề sửa lỗi trở nên đơn giản hơn. Một lợi ích nữa là nhờ xáo trộn làm giảm được độ tương quan của các chuỗi đầu vào các bộ mã thành phần, do đó khi đưa qua quá trình giải mã nhiều trạng thái sẽ làm tăng chất lượng mã hoá lên rất nhiều so với quá trình giải mã duy nhất một trạng thái.

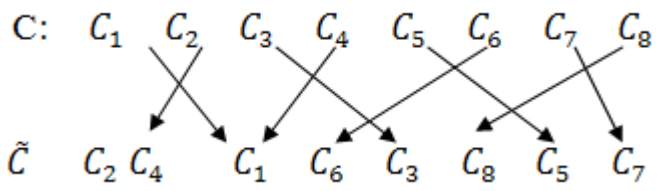
Ta giả sử có chuỗi bit gốc : và vị trí các bít lỗi là liền kề nhau như sau:

$$c = (c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8,),$$

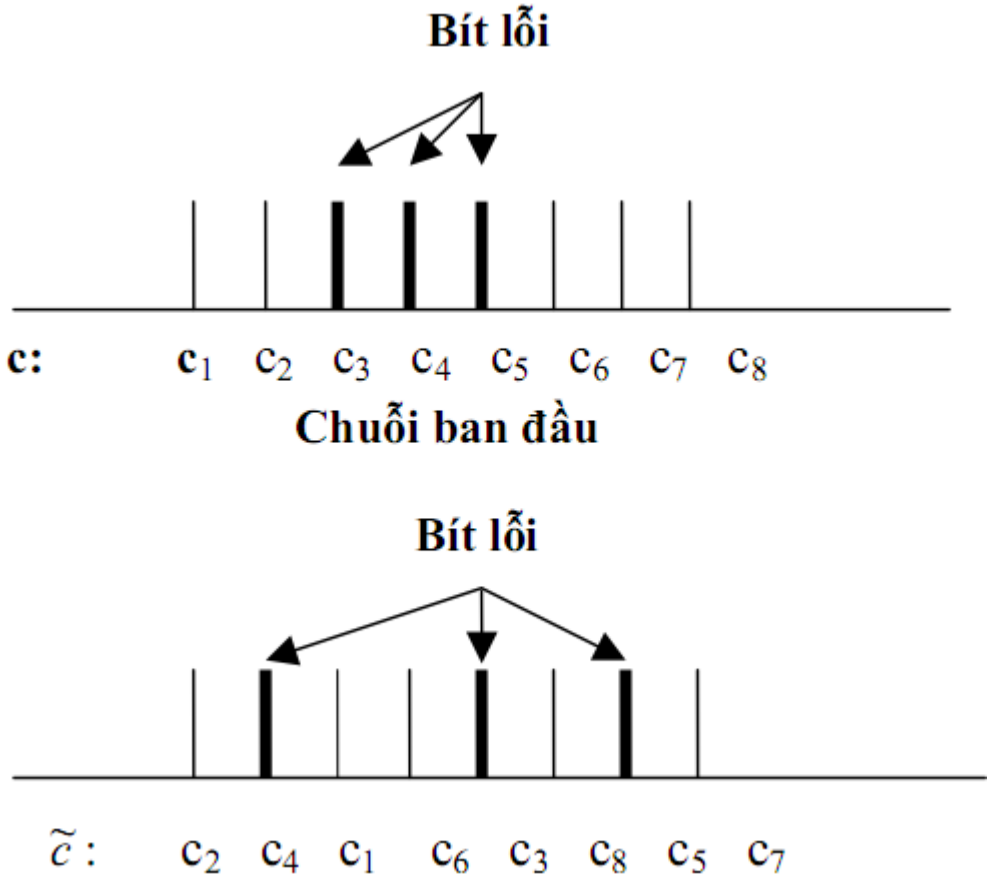
và chuỗi xáo trộn :

$$\tilde{c} = (\tilde{c}_1, \tilde{c}_2, \tilde{c}_3, \tilde{c}_4, \tilde{c}_5, \tilde{c}_6, \tilde{c}_7, \tilde{c}_8,) = (c_2, c_4, c_1, c_6, c_3, c_8, c_5, c_7)$$

Sắp xếp trong bộ xáo trộn :



Như thể giả sử bít C_3, C_4, C_5 bị lỗi khi đó sau khi qua bộ xáo trộn thì các bít lỗi phân bố ngẫu nhiên độc lập nhau (hình vẽ dưới)



Tóm lại, chương vừa rồi đã trình bày về vai trò mã kênh trong hệ thông tin số, giới hạn Shannon và phân tích về hai loại mã có chất lượng cao trong hệ thống viến thông : mã chập và mã kề là cơ sở để nghiên cứu tiếp về mã Turbo.

Chương 2
CÁC KHÁI NIỆM VỀ MÃ TURBO

Giản đồ mã kể lần đầu tiên được đề xuất bởi Forney như là phương pháp để nâng cao mã hoá bằng cách kết hợp 2 hay nhiều khối đơn giản liên hệ với nhau hay các mã thành phần (đôi khi còn được gọi là các mã cấu thành). Kết quả là các mã có khả năng sửa lỗi lớn hơn rất nhiều so với các mã sửa lỗi khác, và chúng được cung cấp với cấu trúc cho phép liên hệ dễ dàng cho việc giải mã phức tạp trở nên nhẹ đi. Một chuỗi mã kề hầu hết thường được sử dụng cho hệ thống giới hạn công suất như các bộ phát trên các đầu dò không gian (deep-space probes). Phổ biến nhất của các giản đồ này bao gồm mã Reed-Solomon ở ngoài (ứng dụng lúc đầu, di chuyển cuối) đi theo sau là mã chập trong (áp dụng cuối, di chuyền lúc đầu). Một mã Turbo có thể được xem như sự chọn lọc hoàn hảo của các cấu trúc mã kề thêm với thuật toán lặp cho việc giải mã kết hợp với dãy mã.

Các mã Turbo lần đầu tiên được giới thiệu vào năm 1993 (bởi Bernou, Glavieux, và Thitimajshima) đưa ra giản đồ về xác suất lỗi như là hàm của Eb/N0 và số lần lặp. Ở đây giản đồ đã được mô tả những thành tựu của chúng với xác suất lỗi bít là 10-5, sử dụng ở tốc độ mã 1/2 qua kênh nhiễu trắng (AWGN- là kênh có mật độ phổ công suất trải đều, tức không đổi) và điều chế BPSK có tỷ số E_b/N_0 dB. Các mã được tạo nên bằng cách sử dụng 2 hay nhiều mã thành phần theo cách lặp khác nhau của cùng một chuỗi thông tin. Trong khi đó, đối với các mã chập bước cuối cùng của bộ giải mã tạo ra quyết định cứng giải mã các bit (hay một cách tổng quát, các ký hiệu được mã hoá), đối với giản đồ mã kề, như mã Turbo, hoạt động thích hợp, thuật toán giải mã có thể không giới hạn bản thân nó vượt qua quyết định cứng trong các bộ giải mã. Thông tin cần dùng nhất được biết từ mỗi bộ giải mã, thuật toán giải mã có ảnh hưởng lẫn nhau đối với quyết định mềm hơn là các quyết định cứng. Đối với hệ thống có hai mã thành phần, khái niệm sau : giải mã Turbo là qua các quyết định cứng ở lối ra của bộ giải mã này lại là đầu vào của bộ giải mã khác, và quá trình này lặp một số lần cho đến khi tạo ra những quyết định đáng tin cậy.

Để đi đến tìm hiểu cấu trúc của mã Turbo và bộ giải lặp chúng ta xem xét các khái niệm toán học liên quan.

2.1 CÁC KHÁI NIỆM VỀ MÃ TURBO

2.1.1. Các hàm hợp lệ (LIKELIHOOD FUNTIONS)

Những thiết lập toán học về kiểm chứng các giả thuyết dựa trên định lý Bayes. Đối với kỹ nghệ thông tin liên lạc, các ứng dụng có liên quan tới kênh AWGN là điều đáng quan tâm hơn cả, tác dụng lớn nhất của định lý Bayes mô tả xác suất hậu nghiệm (APP- a posteriori probability) của một quyết định trong các số hạng của biến ngẫu nhiên liên tục *x* là :

$$P(d = i|x) = \frac{P(x|d = i)P(d = i)}{p(x)} \tag{2.1}$$

Và

$$P(x) = \sum_{i=1}^{M} p(x|d = i)P(d = i) \tag{2.2}$$

Trong đó $P(d = i|x)$ là APP,và $d=i$ biểu diễn dữ liệu d thuộc về lớp tín hiệu thứ i từ tập hợp M lớp. Hơn nữa, $P(d = i|x)$ biểu diễn hàm mật độ xác suất (pdf) của tín hiệu nhiễu cộng dữ liệu có giá trị liên tục được nhận x, với điều kiện trên lớp tín hiệu $d=i$. Cũng vây, $P(d = i)$ được gọi là xác suất tiền nghiệm (a priori probability), là xác suất xảy ra ở lớp tín hiệu thứ i. x điển hình là sự ngẫu nhiên quan sát hay là một thống kê kiểm tra được tạo ra ở lối ra của bộ giải điều chế hay ở bộ vi xử lý khác. Do đó, p(x)là hàm mật độ xác suất pdf của tín hiệu nhận x, tạo ra thống kê kiểm tra trên toàn bộ không gian của các lớp tín hiệu.

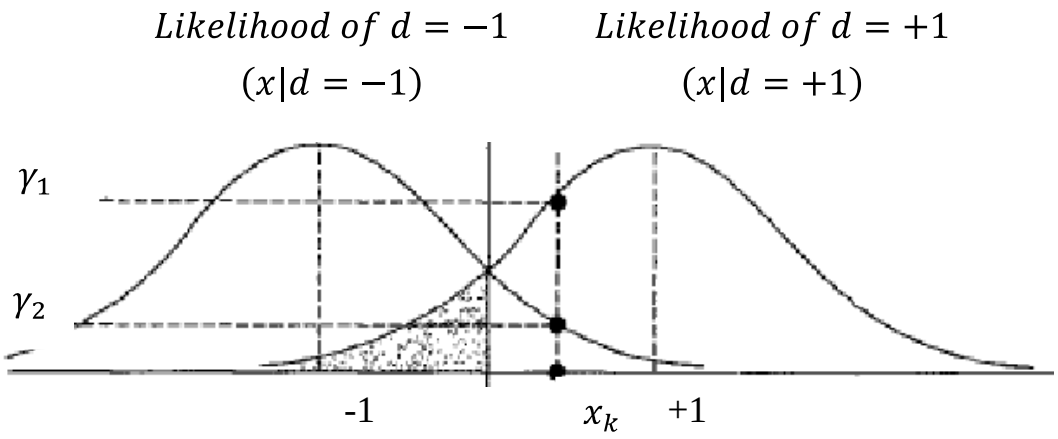
Phương trinh (2.1), cho quan sát đặc biệt, *p(x)* là hệ số chia kể từ khi nó đươc sinh ra bởi lấy trung bình qua tất cả các lớp của không gian. Chữ thường p được sử dụng để chỉ rõ hàm pdf của biến ngẫu nhiên liên tục, và chữ hoa P được sử dụng để chỉ xác suất (tiền nghiệm va APP). Xác định APP của tín

hiệu nhận được từ phương trình (2.1) có thể được xem như là kết quả của thí nghiệm. Trước khi thí nghiệm, nói chung có tồn tại (hoặc có thể ước lượng được) một xác suất tiền nghiệm $P(d=i)$. Thí nghiệm bao gồm sử dụng phương trình (2.1) cho tính toán APP, $P(d = i|x)$,có thể xem như là "lọc lựa tinh tế" tin tức trước đó về dữ liệu, xảy ra bởi quá trình kiểm tra tín hiệu nhận x.

2.1.2 Tr- ờng hợp hai lớp tín hiệu

Đối với các phần tử logic nhị phân 1 và 0 được biểu diễn bằng các mức điện thế điện tử tương ứng là +1 và -1. Biến d được sử dụng để biểu diễn bit dữ liệu được phát, mỗi khi xuất hiện, nó xem như là điện thế hoặc phần tử logic. Đôi khi theo một cách định dạng là tiện lợi hơn định dạng khác ; chúng ta có thể nhận ra sự khác nhau trong từng hoàn cảnh. Bít nhị phân 0 (hay mức điện thế -1) là phần tử không trong phép cộng. Đối với việc phát tín hiệu qua kênh AWGN, Hình (1.10) chỉ ra hàm pdf có điều kiện, ám chỉ như là hàm hợp lệ (likelihood funtions). Hàm bên phải $p(x|d = +1)$ chỉ rõ hàm pdf của biến ngẫu nhiên x với điều kiện $d=+1$ đang được phát. Hàm bên trái $p(x|d = -1)$ giảI thích tương tự,là hàm pdf của biến ngẫu nhiên x với điều kiện $d = -1$ đang được phát. Hoành độ biểu diễn toàn bộ miền giá trị có thể của thống kê kiểm tra x được tạo tại bộ nhận. Trên hình (1.10), với giá trị quy định x_k được biết, ở đây chỉ số chỉ ra khoảng thời gian quan sát thứ k. Một đường thẳng từ x_k cắt hai hàm hợp lệ sinh ra 2 giá trị hợp lệ $l_1 = p(x_k|d_k = +1)$ và $l_2 = p(x_k|d_k = -1)$. Vai trò của quyết định cứng được biết gọi là hợp lệ tối đa, lựa chọn dữ liệu $d_k = +1$ hay $d_k = -1$ kết hợp với giá trị lớn trong hai giá trị bị chặn hay tương ứng. Đối với mỗi bít dữ liệu ở thời điểm k, điều này tương đương việc giải mã $d_k = +1$ nếu x_k nằm ở bên phải của đường quyết định đặt tên là γ_0,ngược lại là giải mã $d_k = -1$

Sơ đồ hàm hợp lệ được biểu diễn như sau:



Hình 2.1: Hàm hợp lệ

Nguyên tắc quyết định tương tự, được gọi là tối đa hậu nghiệm (maximum a posteriori -MAP), có thể được xem như là nguyên tắc tối thiểu xác suất lỗi (minimum - probability - of- error rule), đánh giá tới xác suất tiền nghiệm của dữ liệu. Biểu thức tổng quát cho nguyên tắc MAP trong thuật ngữ APPs là :

$$P(d = +1)|x \underset{H_2}{\overset{H_1}{\gtrless}} P(d = -1|x) \qquad (2.3)$$

Phương trình (2.2) cho thấy chúng ta có thể lựa chọn các giả thiết H1, $(d = +1)$ nếu APP, $P(d = -1|x)$ lớn hơn APP, $P(d = -1|x)$ Ngược lại, lựa chọn giả thiết H2, $(d = -1)$. Sử dụng định lý Bayes vào phương trình (2.1), APPs trong phương trình (2.3) có thể được thay thế bởi các biểu thức tương đương, sinh ra:

$$P(x|d = +1)P(d = +1) \underset{H_2}{\overset{H_1}{\gtrless}} P(d = -1|x)P(d = -1) \qquad (2.4)$$

Ở đây, pdf p(x) xuất hiện ở cả 2 vế của bất phương trình (2.1) nên đã bị khử. Phương trình (2.4) được biểu diễn trong thuật ngữ về tỷ số, tạo ra gọi là kiểm tra tỷ số hợp lệ (loglikelihood ratio test) như sau:

$$\frac{p(x|d = +1)}{p(x|d = -1)} \underset{H_2}{\overset{H_1}{\gtrless}} \frac{P(d = -1)}{P(d = +1)}$$

Hay

$$\frac{p(x|d=+1)P(d=+1)}{p(x|d=-1)P(d=-1)} \underset{H_2}{\overset{H_1}{\gtrless}} 1$$

(2.5)

2.1.3. Tỷ số log- hợp lệ (LOG-LIKELIHOOD RATIO)

Bằng cách đưa ra thuật toán về tỷ số hợp lệ được phát triển trong phương trình (2.3) đến phương trình (2.5), chúng ta tạo ra được metric có ích gọi là tỷ số log-hợp lệ (Log-Likelihood Ratio – LLR). Nó là số thực biểu diễn lối ra quyết định cứng của bộ tách sóng (detector), được định nghĩa :

$$L(d|x) = \log\left[\frac{P(d=+1|x)}{P(d=+1|x)}\right] = \log\left[\frac{p(x|d=+1)P(d=+1|x)}{p(x|d=-1)P(d=+1|x)}\right]$$

(2.6)

Do vậy

$$L(d|x) = \log\left[\frac{p(x|d=+1)}{p(x|d=-1)}\right] + \log\left[\frac{P(d=+1|x)}{P(d=+1|x)}\right]$$

(2.7)

Hay :

$$L(d|x) = L(x|d) + L(d) \quad (2.8)$$

Ở đây *L(x | d)* là LLR của thống kê kiểm tra x tạo ra bởi phép đo của lối ra kênh x dưới điều kiện lần lượt là $d=+1$ hay $d=-1$ có thể đã được phát, và L(d) là LLR tiền nghiệm của bít dữ liệu d. Đơn giản ký hiệu, Phương trình (2.8) được viết :

$$L'(\hat{d}) = L_c(x) + L_d \quad (2.9)$$

Ở đây, ký hiệu $L_c(x)$ nhấn mạnh rằng toán hạng LLR này là kết quả của phép đo kênh truyền tạo ở bộ nhận. Phương trình (2.1) tới (2.9) được phát triển với chỉ một bộ tách sóng dữ liệu trong tư duy của chúng ta. Tiếp theo, sự giới thiệu về bộ giải mã sẽ sinh ra những lợi ích thông thường của việc tạo quyết định. Cho mã hệ thống, lối ra LLR(lối ra mềm) của bộ giải mã bằng :

$$L(\hat{d}) = L'(\hat{d}) + L_e(\hat{d}) \quad (2.10)$$

Ở đây, $L'(\hat{d})$ là LLR của bít dữ liệu ở lối ra của bộ giải điều chế (lối vào của bộ giải mã), và $L_e(\hat{d})$ được gọi là LLR ngoại lai, biểu diễn thông tin bổ sung được vay mượn từ quá trình giải mã. Chuỗi lối ra của bộ giải mã hệ thống được cấu thành các giá trị biểu diễn các bít dữ liệu và các bít chẵn lẻ. Từ phương trình (3.9) và (3.10) lối ra LLR của bộ giảI mã bây giờ được viết như sau :

$$L(\hat{d}) = L_c(x) + L(\hat{d}) + L_e(\hat{d}) \quad (2.11)$$

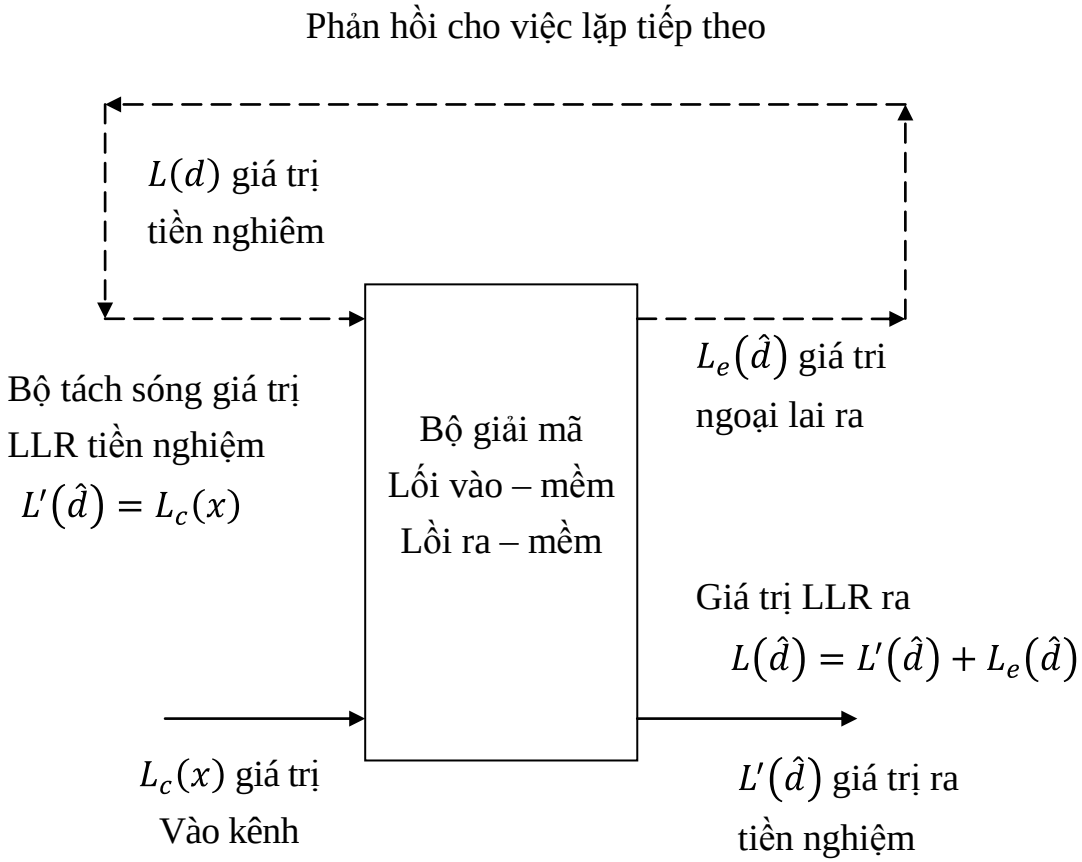
Phương trình (2.11) cho thấy lối ra LLR của bộ giải mã hệ thống có thể được biểu diễn như là có ba phần tử LLR - một phép đo kênh truyền, mộ thông tin tiền nghiệm về dữ liệu,và LLR ngoại lai xuất phát duy nhất từ bộ giải mã. Cuối cùng tạo ra $L(\hat{d})$ mỗi LLRs riêng có thể được cộng thêm vào phương trình (3.11), vì 3 số hạng đều là thống kê độc lập. Lối ra bộ giảI mã mềm này $L(\hat{d})$ là một số thực cung cấp một quyết định cứng - Giá trị dương của $L(\hat{d})$ quyết định rằng *d=+1*, và giá trị âm quyết định $d = +1$. Độ lớn của $L(\hat{d})$ thể thiện độ tin cậy của quyết định đó. Thông thường giá trị của $L_e(\hat{d})$ đối với việc giải mã có cùng ký hiệu như là $L_c(x) + L(d)$ và do đó hoạt động nhằm cải thiện độ tin cậy của $L(\hat{d})$.

3.1.4 Nguyên lý của giải mã lặp

Trong bộ nhận thông tin thông thường, bộ giải điều chế thường được thiết kế để tạo ra những quyết định mềm và rồi được truyền tới bộ giải mã. Việc cải thiện chất lượng (hiệu suất) –lỗi (error- performance) sử dụng hệ thống như quyết định mềm so sánh với quyết định cứng được đánh giá gần 2dB trong AWGN. Như bộ giải mã có thể được gọi là Bộ giải mã lối vào – mềm / lối ra- mềm (soft- input / soft – output), bởi vì quá trình giải mã cuối cùng ở lối ra của bộ giải mã phải kết thúc trong các bít (Các quyết định cứng). Với mã Turbo, ở đây, sử dụng 2 hay nhiều mã thành phần, và việc giải mã bao hàm từ một bộ giải mã là lối vào cho bộ cứng sẽ không được thích hợp. Đó là nguyên nhân cấc quyết định cứng trong bộ giải mã làm giảm bớt chất lượng hệ thống (so sánh với các quyết định mềm). Do đó những gì cần thiết cho việc giải mã của các mã Turbo là bộ giải mã lối vào - mềm / lối ra- mềm. Việc giải mã lặp đầu tiên

của bộ giải mã lối vào -mềm / lối ra- mềm được giải thích trong hình (2.2), Tổng quát ta giả sử rằng dữ liệu nhị phân có khả năng là như nhau, tạo ra giá trị LLR tiền nghiệm ban đầu *L(d)* = 0 ứng với số hạng thứ ba trong phương trình (2.7). Giá trị LLR kênh truyền $L_e(x)$ được đo bởi việc lập logarit của tỷ số giá trị l_1 và l_2 cho quan sát riêng biệt *x* (Hình 2.1), xuất hiện ở số hạng thứ hai của phương trình (2.7). Lối ra $L(\hat{d})$ của bộ giải mã trong hình 2.2 được tạo thành LLR từ bộ tách sóng $L'(\hat{d})$ và lối ra LLR ngoại lai $L_e(x)$, biểu diễn thông tin vay mượn từ quá trình giải mã. Như giải thích trong hình 2.2, việc giải mã lặp, hợp lệ ngoại lai, được phản hồi tới lối vào (của bộ giải mã thành phần khác) để tạo ra sự lọc lựa tinh tế xác suất tiền nghiệm của dữ liệu cho bộ lặp tiếp theo.

Ta sẽ minh họa bộ SISO cho mã hệ thống :



Hình 2.2 Bộ giải mã lối vào – mềm / lối ra – mềm

(cho mã hệ thống)

2.2 ĐẠI SỐ LOG - HỢP LỆ (LOG - LIKELIHOOD ALGEBRA)

Để giải thích sự phản hồi lặp tốt nhất của lối ra bộ giải mã mềm, chúng ta sẽ dùng khái niệm Đại số Log- hợp lệ. Đối với dữ liệu độc lập thống kê d, tổng của hai tỷ số log - hợp lệ (LLRs) được định nghĩa như sau :

$$L(d_1) \boxplus L(d_2) \triangleq L(d_1 \oplus d_2) = \log_e \frac{e^{L(d_1)} + e^{L(d_2)}}{1 + e^{L(d_1)} e^{L(d_2)}} \tag{2.12}$$

$$\approx (-1) \times sgn[L(d_1)] \times sgn[L(d_2)] \times min(|L(d_1)|, |L(d_2)|) \tag{2.13}$$

Ta sẽ chứng minh công thức (2.12). Thật vậy, xuất phát từ định nghĩa về $L(d_1)$ chúng ta có:

$$L(d) = \log_e \left[\frac{P(d=+1)}{P(d=-1)}\right] = \log_e \frac{P(d=+1)}{1 - P(d=+1)}$$

Do đó :

$$e^{L(d)} = \left[\frac{P(d=+1)}{1 - P(d=+1)}\right]$$

$$e^{L(d)} - e^{L(d)} P(d=+1) = P(d=+1)$$

$$e^{L(d)} = P(d=+1)\left(1 + e^{L(d)}\right)$$

$$P(d=+1) = \frac{e^{L(d)}}{(1 + e^{L(d)})}$$

Mặt khác :

$$P(d=-1) = 1 - P(d=+1) = \frac{1}{1 + e^{L(d)}}$$

Ở đây $d_1$và d_2 là các bít dữ liệu độc lập thống kê biểu diễn các thế +1 và -1 tương ứng với mức lôgíc 1 và 0. Theo cách này, thì $d_1 \oplus d_2$ sinh ra -1 khi d_1

và d_2 có các giá trị như nhau (cùng là +1 hay -1) và sinh ra +1 khi d_1 và d_2 có giá trị khác nhau

Do đó:

$$L(d_1 \oplus d_2) = \log_e \frac{P(d_1 \oplus d_2 = 1)}{P(d_1 \oplus d_2 = -1)}$$
$$= \log_e \frac{P(d_1 = +1)p(d_2 = -1) + P(d_1 = -1)p(d_2 = +1)}{P(d_1 = +1)p(d_2 = +1) + P(d_1 = -1)p(d_2 = -1)}$$
$$= \log \frac{e^{L(d_1)} + e^{L(d_2)}}{(1 + e^{L(d_1)}e^{L(d_2)})}$$

Ta sẽ tìm hiểu ý nghĩa các ký hiệu trong công thức : ở đây logarit tự nhiên đươc sử dụng, và hàm sgn(.) biểu diễn "hàm dấu". Có 3 phép cộng trong phương trình (2.12). Dấu + được sử dụng cho phép cộng thông thường. Dấu $\oplus$ được sử dụng để chỉ tổng modul-2 của dữ liệu được biểu diễn dưới dạng các số nhị phân. Dấu $\boxplus$ chỉ phép cộng log-hợp lệ, tương đương với phép toán được mô tả trong phương trình (2.12). Tổng của hai LLRs ký hiệu bởi toán hạng $\boxplus$ được định nghĩa là LLR của tổng modul-2 của các bít dữ liệu độc lập thống kê cơ sở. Phương trình (2.13) lấy gần đúng với phương trình (2.12) và sẽ rất có lợi trong ví dụ về số mà ta sẽ xét sau này.

Tổng cuả LLRs, như được mô tả ở phương trình (2.12) hay (2.13) sinh ra mốt số kết quả đáng quan tâm sau khi mà LLRs là rất lớn hay rất nhỏ:

1. $$L(d) \boxplus \infty = -L(d)$$

Và

$$L(d) \boxplus 0 = 0$$

2.Giải mã hàng ngang và sử dụng phương trình (2.11), tạo ra LLR ngoại lai ngang

$$L_{eh}(\hat{d}) = L(\hat{d}) - L_c(x) - L(d)$$

3. Đặt $L(\hat{d}) = L_{eh}(\hat{d})$ cho việc giải mã dọc ở bước 4

4. Giải mã dọc, và sử dụng phương trình (2.11), chúng ta tạo ra LLR ngoại lai dọc

$$L_{ev}(\hat{d}) = L(\hat{d}) - L_c(x) - L(d)$$

5. Đặt $L(\hat{d}) = L_{ev}(\hat{d})$ cho việc giải mã dọc ở bước 2. Rồi lặp bước 2 tới bước 5

6. Sau khi lặp đủ (Lặp từ bước 2 tới 5) để tạo ra quyết định đáng tin cậy, chuyển tới bước 7.

7. Lối ra mềm là :

$$L(\hat{d}) = L_c(x) + L_{ev}(\hat{d}) + L_{eh}(\hat{d})$$

Phần tiếp theo, ta sẽ lấy ví dụ để minh hoạ ứng dụng của thuật giải đối với mã nhân đơn giản.

2.2.1. Mã chẵn lẻ - đơn hai chiều(Two–Dimensional Single – Parity Code)

Tại bộ mã hoá, các bít dữ liệu và bít chẵc lẻ đều có mối liên hệ giữa bít dữ liệu và chẵn lẻ trong hàng hay cột riêng biệt được diễn tả dưới dạng các số nhị phân (1,0)

$$d_i \boxplus d_j = p_{ij} \qquad (2.15)$$

Và

$$d_i = d_j \boxplus p_{ij} \qquad \text{i,j}\in\{(1,2),\ (3,4),\ (1,3),\ (2,4),\} \qquad (2.16)$$

Dấu $\oplus$ là phép cộng modul-2. Các bít phát được biểu diễn dưới các chuỗi

$$d_1, d_2, d_3, d_4, p_{12}, p, p_{13}, p_{24}$$

Ở lối vào bộ nhận, các bít nhiễu – sai được biểu diễn bởi chuỗi $\{x_i\}$, $\{x_{ij}\}$, ở đây $x_i = d_i + n$ cho mỗi bit dữ liệu nhận, $x_{ij} = p_{ij} + n$ cho mỗi bít chẵn lẻ, và *n* biểu diễn phân phối nhiễu, nó là thống kê độc lập với cả d_i và p_{ij}. Các chỉ số *i* và *j* biểu diễn vị trí trong mảng lối ra bộ mã hoá. Tuy nhiên, sẽ tiện lợi hơn khi chúng ta biểu diễn chuỗi nhận dưới dạng $\{x_k\}$ ở đây *k* là chỉ số thời gian. Cả hai quy ước sẽ được sử dụng. Chúng ta sử dụng *i* và *j* khi trọng tâm vào mối liên hệ về vị trí trong mã nhân, và sử dụng *k* khi trọng tâm trên khía cạnh chung về tín hiệu liên quan thời gian (Time – related signal). Sử dụng mối quan hệ được suy ra từ phương trình 2.7 đến 2.9, và giả sử cùng kiểu nhiễu AWGN, LLR cho phép đo kênh truyền của tín

hiệu x_k nhận được ở thời điểm k, được viết là :

$$L_c(x_k) = \log_e\left[\frac{p(x_k|d_k = +1)}{p(x_k|d_k = -1)}\right] \qquad (2.17a)$$

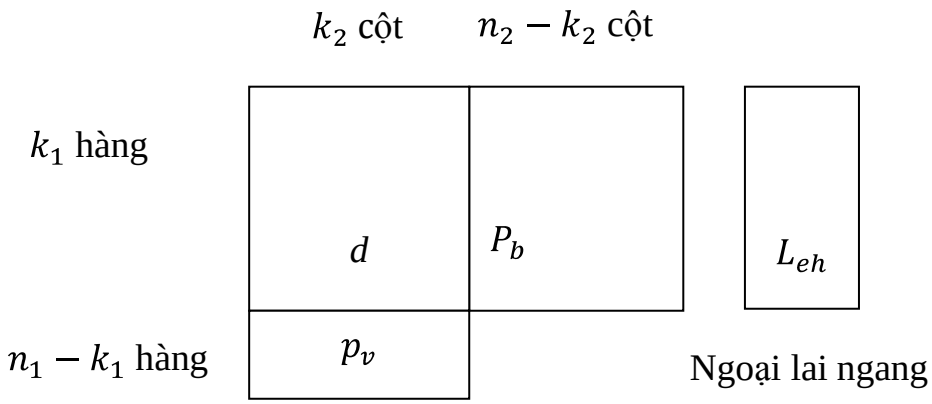
Như vậy, chúng ta đã tìm hiểu cơ sở lý thuyết về tỷ số log-hợp lệ (LLR),một khái niệm là nền tảng để xây dựng nên sơ đồ cấu trúc giải mãTurbo. Bây giờ để thấy rõ tác dụng của thuật toán trên, chúng ta xét trên ví dụ về mã nhân (tức mã được xây dựng trên cơ sở không gian hai chiều)

2.2.2 Mã nhân (PRODUCT CODE)

Xem xét mã hai chiều (mã nhân) được mô tả trên hình 2.4. Cấu trúc có thể được mô tả như là một mảng dữ liệu tạo bởi k_1 hàng và k_2 cột. k_1 hàng chứa các từ mã tạo bởi k_2 bít dữ liệu và $(n_2 - k_2)$ bit chẵn lẻ. Do vậy, mỗi k_1 hàng biểu diễn một từ mã từ mã $(n_2 - k_2)$ Tương tự, k_2 cột chứa các từ mã tạo bởi k_1 bít dữ liệu và $(n_1 - k_1)$ bít chẵn lẻ. Do vậy, mỗi k_2 cột biểu diễn một từ mã từ mã (n_1, k_1). Tỷ lệ khác nhau của cấu trúc được đặt tên d cho dữ liệu, p_h cho chẵn lẻ ngang (hướng theo các hàng), và p_v cho chẵn lẻ cột (hướng theo các cột). Kết quả là, khối $(k_1 \times k_2)$ bít dữ liệu được mã hoá với hai mã -mã ngang (horizontal code) và mã dọc (vetical code).

Ngoài ra, trong hình 2.4 có các khối được đặt tên là L_{eh} và L_{ev} chứa các giá trị LLR ngoại lai được biết từ các bước gải mã ngang và dọc, tương ứng. Các mã sửa lỗi nói chung cung cấp một vài cải thiện về chất lượng. Chúng ta sẽ xem xét rằng, LLRs ngoại lai miêu tả phép đo của việc cải thiện đó. Chú ý rằng, mã nhân này là một ví dụ đơn giản cho mã kề. Cấu trúc của nó chứa đựng 2 bước mã hoá riêng biệt – ngang và dọc. Chúng ta xem lại quyết định giải mã cuỗi cùng cho mỗi bít và điểm mấu chốt đáng tin cậy của nó trên giá trị của $L(\hat{d})$, như đã chỉ ở phương trình 2.11. Với phương trình này, thuật toán sinh ra LLRs ngoại lai (ngang và dọc) và $L(\hat{d})$ cuối cùng có thể được mô tả. Đối với mã nhân, quá trình của thuật toán giải mã lặp như sau :

Đặt LLR tiền nghiệm *L(d) = 0* (Trừ phi xác suất tiền nghiệm của các bít dữ liệu không có khả năng bằng nhau)



Hình 2.4 Tích nhân hai chiều

$d_1 = 1$	$d_2 = 0$	$p_{12} = 1$
$d_3 = 0$	$d_4 = 1$	$p_{34} = 1$
$p_{13} = 1$	$p_{24} = 1$	

Hình 2.5a Các chỉ số phân lối ra bộ mã hóa

$L_c(x_1) = 1.5$	$L_c(x_2) = 0.1$	$L_c(x_{12}) = 2.5$
$L_c(x_3) = 0.2$	$L_c(x_4) = 0.3$	$L_c(x_{34}) = 2.0$
$L_c(x_{13}) = 6.0$	$L_c(x_{2.4})$	

Hình 2.5.b Tỷ số log-hợp lệ lối vào bộ giải mã

Hình 2.5 Ví dụ tích nhân

$$L_c(x) = \log_e \left[\frac{\frac{1}{\sigma\sqrt{2\pi}} exp\left[-\frac{1}{2}\left(\frac{x_k - 1}{\sigma} \right) \right]^2}{\frac{1}{\sigma\sqrt{2\pi}} exp\left[-\frac{1}{2}\left(\frac{x_k + 1}{\sigma} \right) \right]^2} \right]$$

(2.17b)

$$= -\frac{1}{2}\left(\frac{x_k - 1}{\sigma}\right)^2 + \frac{1}{2}\left(\frac{x_k + 1}{\sigma}\right)^2 = \frac{2}{\sigma^2}x_k \quad (2.17c)$$

Thông thường thì nhiễu có varian $\sigma^2 = 1$, do đó ta có

$$L_{c(x)} = 2x_k \quad (2.18)$$

Ta xét chuỗi dữ liệu d_1, d_2, d_3, d_4 là các chữ số nhị phân 1 0 0 1.

Bằng cách sử dụng Phương trình (2.15), và chuỗi chẵn lẻ lần lượt là $p_{12}, p_{34}, p_{13}, p_{24}$ nhận các giá trị nhị phân là 1 1 1 1.

Do đó, chuỗi phát là :

$$\{d_i, p_{ij}\} = 1\ 0\ 0\ 1\ 1\ 1\ 1\ 1 \quad (2.19)$$

Khi các bít dữ liệu được diễn đạt dưới giá trị là thế lưỡng cực +1 và -1 tương ứng với các mức logic 1 và 0, chuỗi phát sẽ là :

$$\{d_i, p_{ij}\} = +1\ -1\ -1\ +1\ +1\ +1\ +1\ +1$$

Giả sử rằng nhiễu phát thay đổi chuỗi dữ liệu chẵn lẻ thành chuỗi nhận được là :

$$\{x_i, x_{ij}\} = 0.75, 0.05, 0.10, 0.15, 1.25, 1.0, 3.0, 0.05 \quad (2.20)$$

Ở đây các phần tử của $\{x_i\}$,$\{\ x_{ij}\}$ tương ứng vị trí bít dữ liệu và chẵn lẻ $\{d_i\}$,$\{p_{ij}\}$ được phát. Do đó, trong thuật ngữ ký hiệu vị trí, chuỗi nhận có thể được ký hiệu như sau :

$$\{L_c(x_i), L_c(x_{ij})\} = 1.5\ ,0.1\ ,0.20\ ,0.2\ ,2.5\ ,2.0\ ,6.0\ ,1.0$$

Các giá trị này được chỉ rõ trong hình 3.5b là các phép đo lối vào bộ giải mã Nó cho thấy, sẽ bằng xác suất tiền nghiệm đối với dữ liệu phát, nếu quyết định cứng được dựa trên cơ sở các giá trị $\{x_k\}$,hay $\{L_c(x_k)\}$ ở trên, quá trình sẽ cho kết quả với lỗi, từ khi d_1 và d_2 sẽ được sắp xếp không đúng như là bít 1.

Như thế ta đã chỉ ra được giá trị phép đo kênh truyền Lc(x), nhiễu,.. và việc quan trọng cuối cùng là ta phải tính giá trị LLR ngoại lai (ngang và dọc).

2.2.3. Hợp lệ ngoại lai (Extrinsic Likelihood)

Đối với ví dụ mã nhân trong hình 2.5, chúng ta sử dụng Phương trình (2.11) để mô tả lối ra mềm đối với tín hiệu nhận tương ứng với dữ liệu d_1 :

$$L(\widehat{d_1}) = L_c(x_1) + L(d_1) + ([L_c(x_2) + L(d_2) \boxplus L_c(x_{12})])$$

(2.22)

Ở đây số hạng : $[L_c(x_2) + L_d(x_2) \boxplus L_c(x_{12})]$ biểu diễn LLR ngoại lai được phân phối bởi mã (tương ứng việc thu dữ liệu d_2 và xác suất tiền nghiệm của nó, kết hợp với việc thu mã chẵn lẻ tương ứng p_{12}). Tổng quát lối ra mềm $L(\hat{d}_i)$ đối với tín hiệu nhận được tương ứng dữ liệu d_i là :

$$L(\hat{d}_i) = L_c(x_i) + L(d_i) + \left([L_c(x_j) + L(d_j) \boxplus L_c(x_{ij})]\right) \quad (2.23)$$

Ở đây $L_c(x_i)$, $L_c(x_j)$, $L_c(x_{ij})$ là các phép đo LLR kênh truyền của việc thu tương ứng $d(x_i)$, $d(x_j)$, $p(x_{ij})$

$L(d_i), L(d)_j$ là LLRs của xác suất tiền nghiệm của d_i và d_j tương ứng. Và : $[L_c(x_2) + L_d(x_2) \boxplus L_c(x_{12})]$ là phân phối ngoại lại từ các mã. Giả sử các tín hiệu có khả năng như nhau, lối ra mềm $L(\hat{d}_1)$ được mô tả bởi bộ tách sóng phép đo LLR của $L_c(x_1) = 1.5$ cho việc thu tương ứng dữ liệu d_1, giá trị dương LLR ngoại lai $[L_c(x_2) = 0.1] \boxplus L_c(x_{12}) = 2.5$ vay mượn từ dữ liệu d_2 và chẵn lẻ p_{12} bởi vậy cung cấp thông tin về dữ liệu d_1 như trong phương trình (2.15) và (2.16). Bây giờ ta sẽ tính toán các giá trị LLR ngoại lai

2.2.4 Tính toán hợp lệ ngoại lai

Vẫn xét ví dụ trong hình 3.5, ta sẽ tính toán $L_{eh}(\hat{d})$ và $L_{ev}(\hat{d})$:

$$L_{eh}(\widehat{d_1}) = [L_c(x_2) + L(d_2)] \boxplus L_c(x_{12}) \quad (2.24a)$$

$$L_{ev}(\widehat{d_1}) = [L_c(x_3) + L(d_3)] \boxplus L_c(x_{13}) \quad (2.24b)$$

$$L_{eh}(\widehat{d_2}) = [L_c(x_1) + L(d_1)] \boxplus L_c(x_{12}) \quad (2.25a)$$

$$L_{ev}(\widehat{d_2}) = [L_c(x_4) + L(d_4)] \boxplus L_c(x_{24}) \quad (2.25b)$$

$$L_{eh}(\widehat{d_3}) = [L_c(x_4) + L(d_4)] \boxplus L_c(x_{34}) \quad (2.26a)$$

$$L_{ev}(\widehat{d_3}) = [L_c(x_1) + L(d_1)] \boxplus L_c(x_{34}) \quad (2.26b)$$

$$L_{eh}(\widehat{d_4}) = [L_c(x_3) + L(d_3)] \boxplus L_c(x_{34}) \quad (2.27a)$$

$$L_{ev}(\widehat{d_4}) = [L_c(x_2) + L(d_2)] \boxplus L_c(x_{24}) \quad (2.27a)$$

Các giá trị LLR chỉ trong hình 2.5 được đưa vào biểu thức $L_{ch}(\hat{d})$ trong các phương trình (2.24) tới (2.27), và, giả sử là các tín hiệu có khả năng như nhau, Các giá trị *L(d)* ban đầu được đặt bằng 0, do đó tạo ra :

$$L_{eh}(\widehat{d_1}) = (0.1 + 0) \boxplus 2.5 \approx -0.1 = L(d_1) \text{ mới} \quad (2.28)$$

$$L_{eh}(\widehat{d_2}) = (1.5 + 0) \boxplus 2.5 \approx -1.5 = L(d_2) \text{ mới} \quad (2.29)$$

$$L_{eh}(\widehat{d_3}) = (0.3 + 0) \boxplus 2.0 \approx -0.3 = L(d_3) \text{ mới} \quad (2.30)$$

$$L_{eh}(\widehat{d_4}) = (0.2 + 0) \boxplus 2.0 \approx -0.2 = L(d_4) \text{ mới} \quad (2.31)$$

Ở đây phép cộng log-hợp lệ đã được tính toán một cách gần đúng, tức ta lấy xấp xỉ trong phương trình (2.13). Tiếp theo, chúng ta tiến hành tạo ra tính toán hàng dọc đầu tiên, sử dụng biểu thức $L_{ev}(\hat{d})$ trong Phương trình (2.24) tới (2.27). Bây giờ, các giá trị của L(d) có thể được tính toán nhanh chóng bằng cách sử dụng những giá trị mới *L(d)* vay mượn từ việc tính toán ngang đầu tiên, chỉ trong phương trình (2.28) tới (2.31). Đó là :

$$L_{ev}(\widehat{d_1}) = [0.2 - 0.3] \boxplus 6.0 \approx 0.1 = L(d_1) \text{ mới} \quad (2.24b)$$

$$L_{ev}(\widehat{d_2}) = [0.3 - 0.2] \boxplus 1.0 \approx -0.1 = L(d_2) \text{ mới} \quad (2.24b)$$

$$L_{ev}(\widehat{d_3}) = [1.5 - 0.1] \boxplus 6.0 \approx -1.4 = L(d_3) \text{ mới} \quad (2.24b)$$

$$L_{ev}(\widehat{d_4}) = [0.1 - 1.5] \boxplus 1.0 \approx 1.0 = L(d_4) \text{ mới} \quad (2.24b)$$

Như vậy, kết quả của phép lặp đầu tiên trong hai bước giải mã (ngang và dọc) như sau :

1.5	0.1
0.2	0.3

-0.1	-1.5
-0.3	-0.2

$L_{eh}(\hat{d})$ sau giải mã ngang đầu tiền

0.1	-0.1
-1.4	1.0

$L_{ev}(\hat{d})$ sau giải mã dọc đầu tiền

Mỗi bước giải mã cải thiện LLRs ban đầu cái mà chỉ dựa trên các phép đo kênh truyền. Điều này được thấy qua bởi việc tính toán LLR lối ra của bộ giải mã, sử dụng phương trình (2.14). LLR ban đầu dương, LLRs ngoại lệ dương tạo ra được sự cải thiện (ở đây ta không đề cập tới thuật ngữ về ngoại lai dọc) :

LLR được cải thiện đối với $L_{eh}(\hat{d})$

1.4	-1.4
-0.1	0.1

LLR ban đầu dương đối với cả hai LLR ngoại lệ ngang và dọc tạo ra được sự cải thiện như sau :

LLR được cải thiện đối với $L_{eh}(\hat{d}) + L_{ev}(\hat{d})$

1.5	-1.5
-1.5	1.1

Đối với ví dụ này, có thể thấy rằng, thông tin vay mượn từ việc giải mã ngang đơn lẻ là đủ để tạo ra quyết định cứng đúng đắn ở lối ra của bộ giải mã, nhưng đối với các bít dữ liệu d_3 và d_4 thì độ tin cậy là rất thấp. Sau khi kết hợp LLR ngoại lai dọc trong bộ giảI mã, giá trị LLR mới đưa ra mức độ cao hơn về độ tin cậy. Chúng ta sẽ tiếp tục thực hiện thêm phép lặp giải mã ngang và dọc để xác định xem có sự thay đổi nào đáng kể ở kết quả thu được.

Chúng ta lại sử dụng mối liên hệ chỉ trong phương trình (2.24) tới (2.27) và thực hiện với việc tính toán lần hai đối với $L_{eh}(\hat{d})$, sử dụng *L(d)* mới từ những tính toán hàng dọc, chỉ ỏ phương trình (2.32) tới (2.25), Do vậy :

$$L_{eh}(\widehat{d_1}) = (0.1 - 0.1) \boxplus 2.5 \approx 0 = L(d_1) \text{ mới} \quad (2.36)$$

$$L_{eh}(\widehat{d_2}) = (1.5 + 0.1) \boxplus 2.5 \approx -1.6 = L(d_2) \text{ mới} \quad (2.37)$$

$$L_{eh}(\widehat{d_3}) = (0.3 + 1.0) \boxplus 2.0 \approx -1.3 = L(d_3) \text{ mới} \quad (2.38)$$

$$L_{eh}(\widehat{d_4}) = (0.2 - 1.4) \boxplus 2.0 \approx 1.2 = L(d_4) \text{ mới} \quad (2.39)$$

Tiếp theo, chúng ta thực hiện tính toán đối với Lev(d), sử dụng L(d) mới từ những tính toán ngang thứ hai, chỉ trong Phương trình (2.36) tới (2.39) ta có :

$$L_{ev}(\widehat{d_1}) = [0.2 - 1.3] \boxplus 6.0 \approx 1.1 = L(d_1) \text{ mới} \quad (2.40)$$

$$L_{ev}(\widehat{d_2}) = [0.3 + 1.2] \boxplus 1.0 \approx -0.1 = L(d_2) \text{ mới} \quad (2.41)$$

$$L_{ev}(\widehat{d_3}) = [1.5 + 0] \boxplus 6.0 \approx -1.5 = L(d_3) \text{ mới} \quad (2.42)$$

$$L_{ev}(\widehat{d_4}) = [0.1 - 1.6] \boxplus 1.0 \approx 1.0 = L(d_4) \text{ mới} \quad (2.43)$$

Như vậy, việc lặp lần hai giải mã ngang và dọc, tạo ra giá trị trước đó, kết quả trong LLR lối ra mềm được tính lại từ Phương trình (3.14), được viết dưới dạng :

$$L(\hat{d}) = L_c(x) + L_{eh}(\hat{d}) + L_{ev}(\hat{d}) \quad (2.44)$$

LLR ngoại lai ngang và dọc của phương trình (2.36) đến (2.43) và kết qủa LLR bộ giải mã được thấy rõ. Trong ví dụ này, lặp ngang và dọc lần hai đưa ra sự cải thiện đáng kể so với lặp lần một. Kết quả chỉ ra sự cân bằng của giá trị đáng tin cậy trong số bốn quyết định dữ liệu.

Các phép đo $L_c(x)$ ban đầu :

1.1	-1.0
-1.5	1.0

0	-1.6
-1.3	1.2

$L_{ev}(\hat{d})$ sau giải mã dọc lần 2

2.6	-2.5
-2.6	2.5

$L_{eh}(\hat{d})$ sau giải mã ngang lần 2

Lối ra mềm $L(\hat{d}) + L_{eh}(\hat{d}) + L_{ev}(\hat{d})$, sau tất cả 4 lần lặp có giá trị như sau :

1.5	0.1
0.2	0.3

Như thế, ta có thể nhận xét thấy rằng ta có thể quyết định đúng đắn về 4 bít dữ liệu và đặc biệt mức độ tin cậy của quyết địh là rất cao. Ví dụ minh hoạ tiêu biểu đươc nguyên lý giải mã Turbo.

Chương 3

CẤU TRÚC MÃ TURBO VÀ BỘ GIẢI LẶP

THUẬT TOÁN GIẢI MÃ TURBO

3.1 GIỚI THIỆU

Mã Turbo được giới thiệu đầu tiên vào năm 1993, bao gồm hai mã chập hệ thống đệ qui (Recursive Systematic Convolution Code - RSCC) kết nối song song kết hợp bộ xáo trộn và thuật toán giải mã lặp.Các thuật toán giải mã Turbo thường có đặc tính giống nhau được kết hợp giữa thuật toán giải mã lặpvà các kiểu giải mã thành phần với lối vào mềm, lối ra mềm (Soft Input/ Soft Output-SISO). Có hai kiểu giải mã thành phần phổ biến cho mã Turbo là giải mã ước lượng theo chuỗi (Sequence Estimation) như SOVA (Soft Out Viterbi Algorithm) và thuật toán ước lượng theo ký hiệu (Symbol by Symbol) như MAP(Maximum a posteriori), cùng những cải tiến của chúng.

Thuật toán giải mã VA và MAP cơ bản là khác nhau về chỉ tiêu tối ưu. Thuật toán giải mã VA là thuật toán tìm kiếm chuỗi trạng thái có khả năng lớn nhất $\hat{s}$ với chuỗi tín hiệu thu y.

$$\hat{s} = arg\left\{\max_{s} P[s|y]\right\}$$

Thuật toán giải mã MAP khác với thuật toán VA lá xác định từng trạng thái $\widehat{s_i}$ cụ thể có khả năng lớn nhất với chuỗi tín hiệu thu y

$$\widehat{s_i} = arg\left\{\max_{s_i} P[s_i|y]\right\}$$

Điểm khác nhau về bản chất giữa chúng là trạng thái được ước lượng bởi thuật toán VA phải có dạng tuyến được kết nối qua lưới, trong khi đó các trạng thái được ước lượng bởi thuật toán MAP thì không cần phải kết nối thành tuyến.

Khi ứng dụng vào các hệ thống truyền dẫn số, thuật toán VA cho phép cực tiểu xác suất lỗi khung FER. Trong khi đó thuật toán MAP cho phép cực tiểu xác suất lỗi bit BER. Do cấu trúc mã Turbo gồm hai bộ mã chập thành phần kết nối song song , vì vậy quá trình giải mã có thể được xem như gồm hai quá trình

ước lượng chuỗi Mardkov(xem phụ lục) (Mỗi quá trình tương ứng với một bộ mã thành phần). Do cả hai quá trình này được thực hiện với cùng một chuỗi dữ liệu nên việc ước lượng có thể chia sẻ thông tin với nhau dưới dạng lặp. Như vậy, đầu ra của bộ giải mã này có thể sử dụng làm thông tin biết trước cho bộ giải mã kia. Nếu đầu ra của mỗi bộ giải mã có dạng quyết định cứng (có sử dụng bộ lượng tử) thì hiệu quả của việc chia sẻ thông tin sẽ không cao. Tuy nhiên, nếu đầu ra là ước lượng mềm thì có thể cải thiện đượcchất lượng một cách đáng kể.

Ta có thể mô tả sơ qua về thuật toán MAP với quá trình giải mã SISO :
Ta cần xác định :

$$\widehat{s_i} = arg\left\{\max_{s_i} P[s_i|y]\right\}$$

Theo công thức xác suất thành phần :

$$P(s_i|y) = \frac{P(y|s_i)P(s_i)}{P(y)} = \frac{f(y|s_i)f(s_i)}{P(y)}$$

Ở đây $P(s_i|y)$ là xác suất hậu nghiệm $f(y|s_i)$ là hàm *pdf, f(s_i)* là xác suất tiền nghiệm, thông thương ta giả sử rằng ban đầu ở lối vào thì

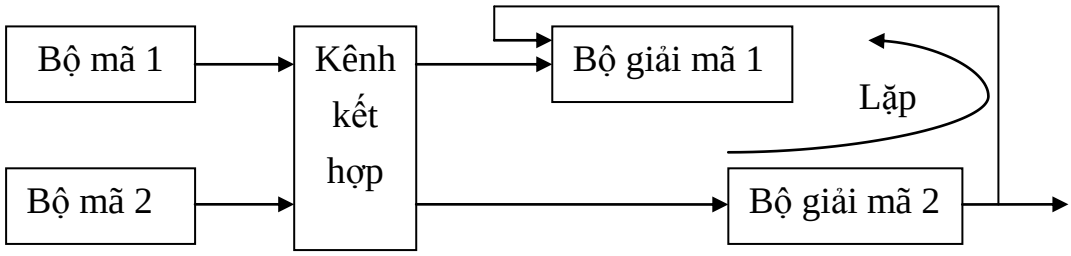
$$f(s_i = M = hằng\ số) \Leftrightarrow L(d) = 0$$

Do đó xác định s_i tương đương với xác định :

$$max\left\{\frac{f(y|s_i)f(s_i)}{P(y)}\right\} \Leftrightarrow max\{f(y|s_i)\}$$

Thay giá trị của y vào hàm $f(y|s_i)$ ta tính được $max\{f(y|s_i)\}$ từ đó $\Rightarrow s_i = f(s_i)$
Và thấy rằng lúc này $f(s_i) \neq M$ khi đó $f(s_i)$ sẽ được đưa tới lối vào của bộ giải mã SISO khác và nó đóng vai trò là thông tin ngoại lai.

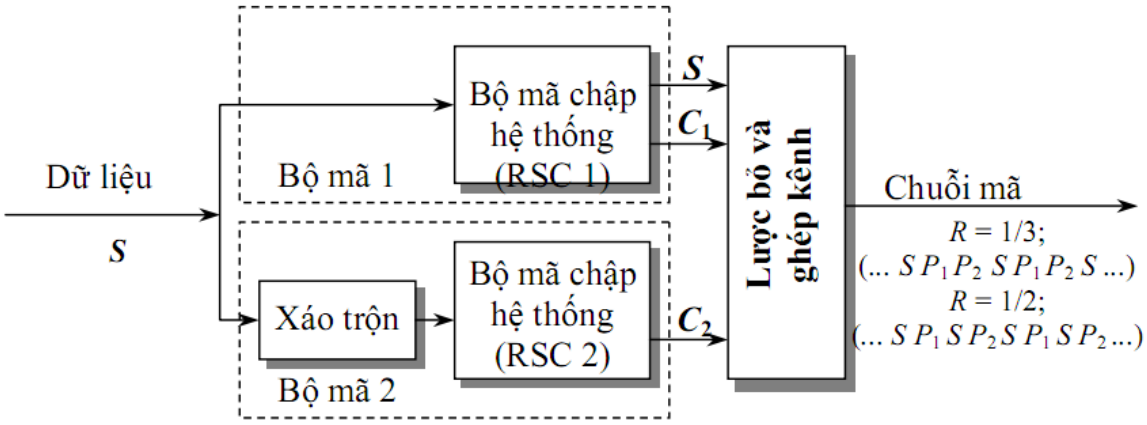
Mã Turbo có chất lượng kiểm soát lỗi trong khoảng vài phần mười dB tính từ giới hạn Shannon. Sự tăng lên một cách đột ngột về chất lượng được dựa trên khám phá và bổ sung các kết quả của Golay mà ông ta đã đưa ra lần đầu tiên từ năm 1950. Ngay sau đó, chất lượng đã được tăng lên khoảng 2dB nhờ vào kết quả việc điều khiển trong mã Turbo. Thành tựu to lớn đó được khuyến nghị ứng dụng vào hệ thống thông tin vô tuyến điện khi mà đòi hỏi về độ rộng băng tần ngày càng tăng do nhu cầu dịch vụ truyền số liệu



Như vậy, Mã Turbo có hai phần quan trọng. Đó là, mã xoắn kết nối song song và giải mã lặp.

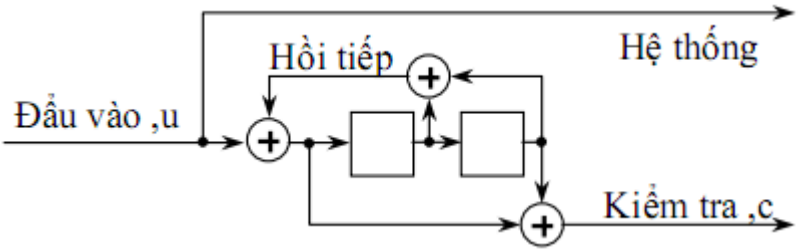
3.2 CẤU TRÚC BỘ MÃ HÓA VÀ GIẢI MÃ LẶP

Mã Turbo có cấu trúc gồm ít nhất hai mã RSC được kết nối song song kết hợp với bộ xáo trộn và thuật toán giải mã SISO:

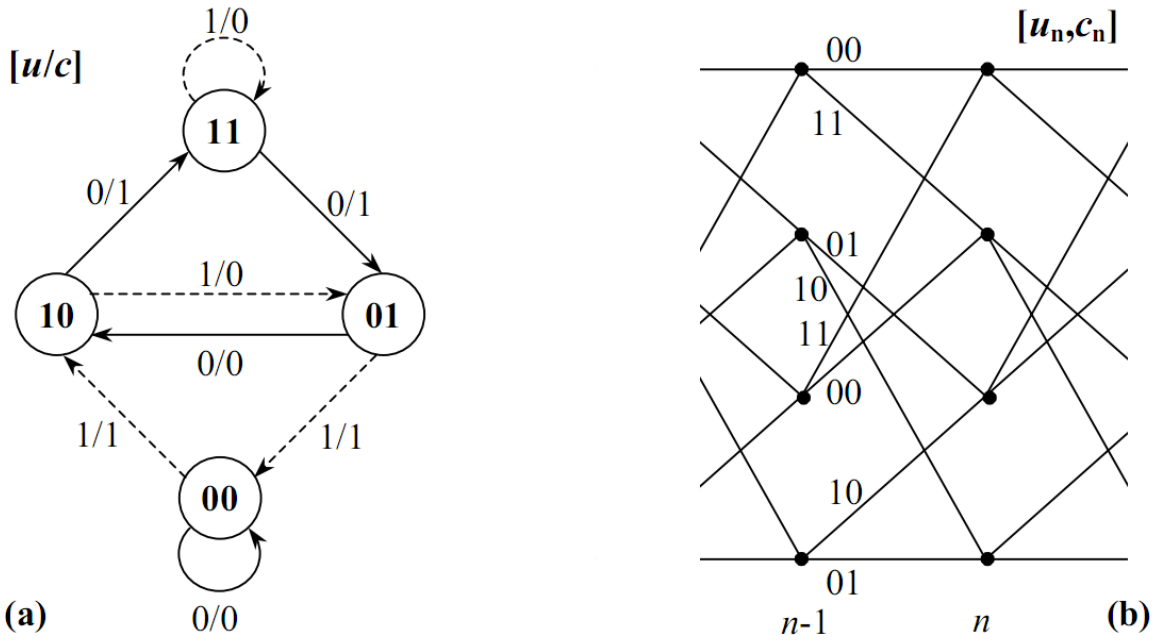


Hình 3.1 Sơ đồ má hóa mã Turbo

Như vậy, chuỗi dữ liệu hệ thống đầu vào S được đưa trực tiếp tới bộ mã chập RSC1 tạo ra các bit kiểm tra C_1, C_2 và đưa qua bộ xáo trộn tới RSC2 tạo ra C_2. Các bít hệ thống và kiểm tra C_1, C_2 được đưa tới bộ lược bỏ và ghép kênh để loại bỏ bớt các bít kiểm tra để tăng tốc độ mã hóa. Nếu ta loại bỏ xen kẽ C_1 và C_1, C_2 ta được tốc độ mã tổng cộng là 1/2, còn không loại bỏ thì tốc độ tổng cộng là 1/3. Tín hiệu đầu ra bộ mã hóa được điều chế và truyền qua kênh như hình 3.1 Hình 3.2 là ví dụ về sơ đồ mã RSC, trong đó chuỗi đầu vào được đưa ngay tới đầu ra gọi là chuỗi bít hệ thống. Sơ đồ trạng thái và sơ đồ lưới của ví dụ trong hình 3.2 được biểu diễn trong hình 3.3:



Hình 3.2 Mã RSC



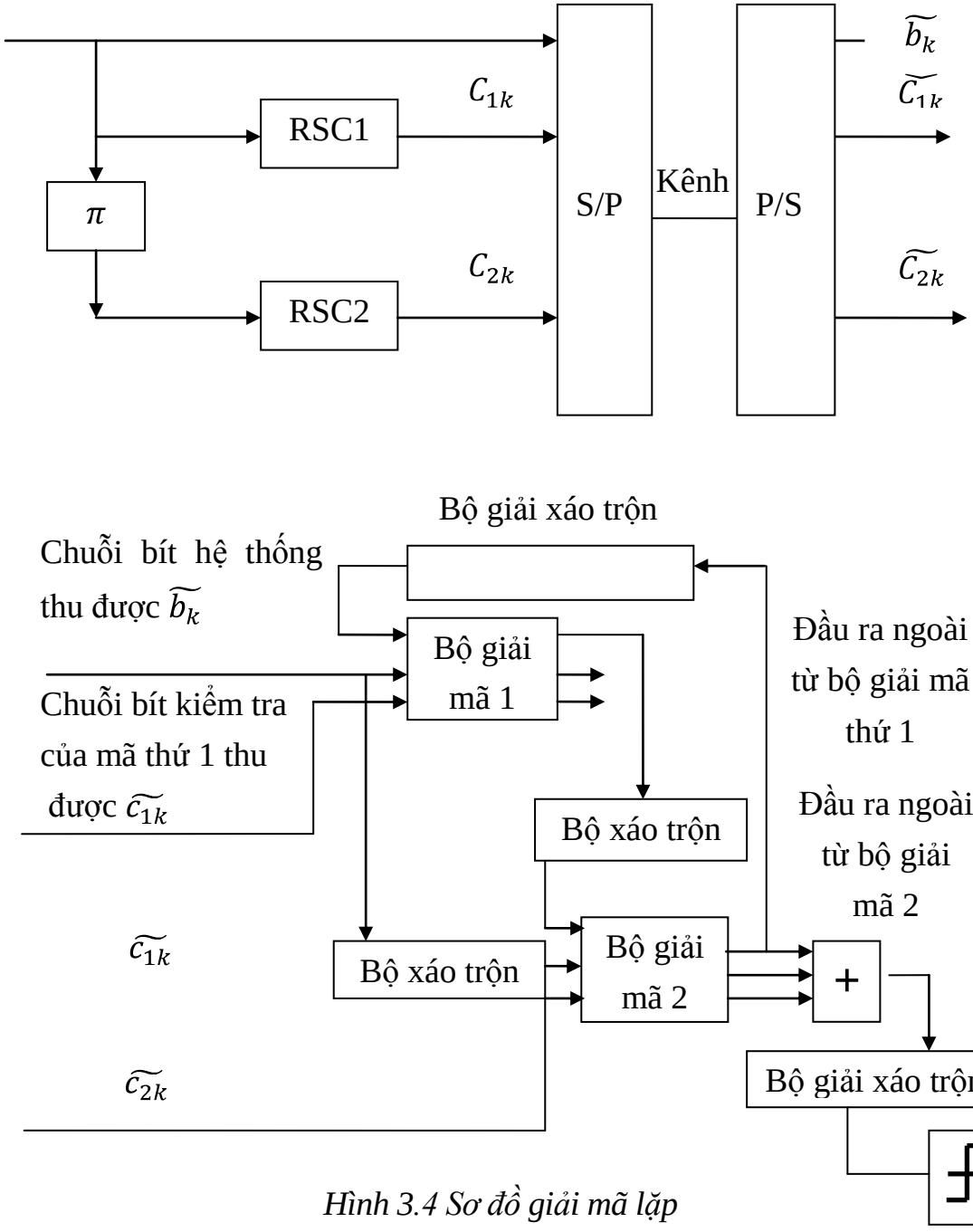
Hình 3.3 Sơ đồ trạng thái(a) và sơ đồ lưới của mã chập (b)

Tại bộ gải mã, bộ tách kênh sẽ tách ra các bít hệ thống và kiểm tra tương ứng với các bộ giải mã SISO.Bộ SISO là thiết bị giải mã với lối vào mềm, lối ra mềm, trong đó đầu vào là độ tin cậy kênh $(L_c(x))$, thông tin tiền nghiệm $L(\hat{d})$. Đầu ra gồm thông tin hậu nghiệm L(d), thông tin hệ thống dư $\left(L_e(\hat{d})\right)$ còn gọi là thông tinh ngoại lai (extrinsic information). Vấn đề này chúng ta đã xét ở chương 2.

Do đầu phát sử dụng bộ xáo trộn, nên trong bộ giải mã có các bộ xáo trộn giống hệt ở đầu phát và các bộ giải xáo trộn tương ứng. Bộ giải mã dùng thuật giải mã lặp nên thông tin dư được sử dụng làm thông tin tiền nghiệm cho bộ giải

mã SISO (RSC1) khác. Để nâng cao chất lượng giải mã người ta tăng số lần lặp n, số lần lặp có thể được quy định trước hoặc tự động dừng theo nhiều biện pháp đánh giá.

Do vậy bộ giải mã lặp có cấu trúc như sau :



Hình 3.4 Sơ đồ giải mã lặp

Như vậy, chuỗi mã tại đầu ra của bộ ghép kênh sẽ được đưa tới bộ phân kênh thông qua kênh truyền. Do đó, tại lối ra của bộ phân kênh chứa bit hệ thống, hai chuỗi bít kiểm tra (bít được mã hóa) và các chuỗi bit này sẽ bị lỗi do

ảnh hưởng của kênh truyền. Giả sử chuối bít hệ thống đầu vào là b_k, hai chuỗi bit mã hóa là C_{1k} và C_{2k}

Chuỗi bit hệ thống và kiểm tra thu được qua kênh truyền tương ứng là $\widetilde{b_k}, \widetilde{C_{1k}}, \widetilde{C_{2k}}$

Theo hình 3.4 thí đầu tiên cho chuỗi bít hệ thống $\widetilde{b_k}$ và $\widetilde{C_{1k}}$ qua bộ giải mã 1 khi đó ta thu được chuỗi bít,giả sử $\widetilde{b_k'}$. Cả hai chuỗi $\widetilde{b_k}$, $\widetilde{b_k'}$ cho qua bộ xáo trộn và sau đó kết hợp với chuỗi $\widetilde{C_{2k}}$ đưa tới lỗi vào của bộ giải mã 2,như thế lối ra của bộ giải mã sẽ là chuỗi $\widetilde{b_k''}$.Để thu được chuỗi thông tin hệ thống ban đầu cần phải cho chuỗi $\widetilde{b_k''}$ qua bộ giải xáo trộn.

Thực ra trong sơ đồ trên thì các bộ giải mã chính là bộ giải mã SISO. Ở đây thông tin ngoại lai dược đưa vào bộ giải mã để tạo nên quá trình lặp.Quá trình lặp cho đến khi nào mà xác suất lỗi bít của chuỗi hệ thống là cực tiểu điều này tương đương với việc lặp cho đến khi khôi phục được chuỗi đầu vào.

3.3 THUẬT TOÁN GIẢI MÃ TURBO

Phần này sẽ trình bày hai thuật toán giải mã Turbo đó là :

Thuật toán giải mã MAP

Thuật toán giải mã SOVA

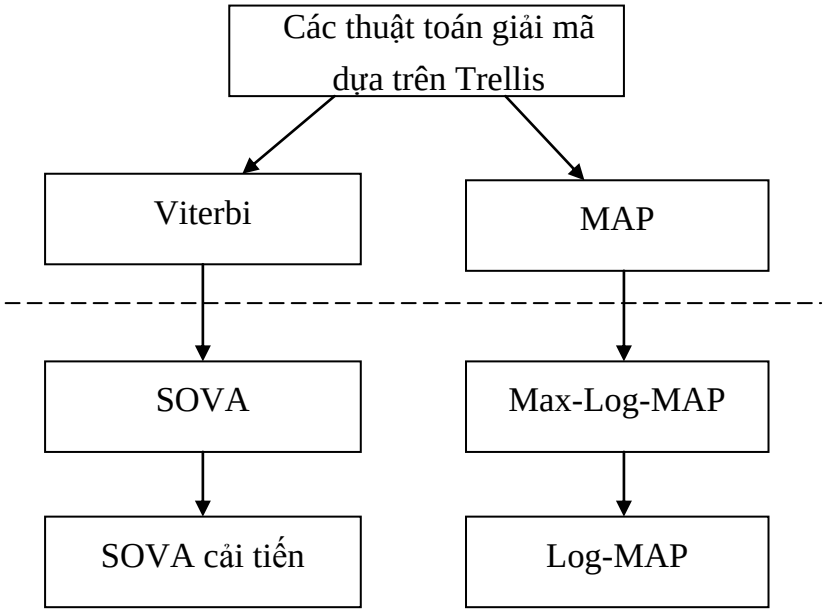
3.3.1 Tổng quan về các thuật toán giải mã

Ngoài sự kết nối các bộ mã tích chập cùng việc sử dụng một thành phần đặc biệt là các bộ chèn, còn một thành phần quan trọng khác trong chất lượng Turbo là qui trình giải mã mềm được thực hiện lặp đi lặp lại và độ phức tạp chỉ tăng tuyến tính theo kích thước khung. Mã PCCC có cấu trúc mã hoá kết nối song song tuy nhiên quá trình giải mã PCCC lại dựa trên sơ đồ giải mã kết nối nối tiếp. Mã Turbo sử dụng bộ giải mã kết nối nối tiếp vì sơ đồ kết nối nối tiếp có khả năng chia xẻ thông tin giữa các bộ giải mã kết nối, trong khi đó các bộ giải mã có sơ đồ kết nối song song chủ yếu giải mã độc lập nhau. Các thông tin này nhờ đặc tính mềm, được trao đổi, khai thác nhiều lần qua các vòng lặp sẽ làm tăng đáng kể chất lượng giải mã.

Trong khi thực hiện một vòng lặp giải mã các thông tin mềm được trao đổi giữa các bộ giải mã thành phần, Forney đã chứng minh được rằng ngõ ra mềm tối ưu cho bộ giải mã phải là xác suất a posteriori (APP) là xác suất của

một bit nào đó được truyền dựa trên tín hiệu nhận được. Vì độ phức tạp của các mã TC chủ yếu là do bộ giải mã lặp nên điều cần thiết trước nhất là tìm hiểu các thuật toán giải mã và tìm ra cách tốt nhất để giải mã mà không làm giảm chất lượng.

Phát triển các thuật toán giải mã hiệu quả là mối quan tâm hàng đầu khi cải tiến mã TC. Hình 3.5 trình bày cái nhìn tổng quan về các họ thuật toán giải mã dựa trên sơ đồ trellis.



Hình 3.5 : Tổng quan các thuật toán giải mã

Họ thứ nhất là họ các thuật toán MAP còn gọi là thuật toán BCJR (Bahl-Cocke- Jelinek-Raviv, tên bốn người đã tìm ra thuật toán này). Thuật toán này liên quan đến các thuật toán giải mã khả năng xảy ra lớn nhất (ML) nhằm làm giảm tối đa xác suất lỗi bit. Họ này bao gồm các thuật toán symbol-by-symbol MAP, là phương pháp tối ưu để tính các thông tin APP, đây là thuật toán dạng tích, độ phức tạp rất cao. Trong họ này còn có hai loại thuật toán làm gần đúng thuật toán MAP để trở thành thuật toán dạng tổng độ phức tạp ít hơn mà chất lượng giải mã gần như tương đương là Log-MAP và phiên bản gần đúng của Log-MAP là Max-log-MAP. Một họ thuật toán giải mã khác là một họ thuật toán dựa trên việc sửa đổi thuật toán Viterbi (VA) có sử dụng thêm metric bổ sung vì VA truyền thống không tính các thông tin APP,

metric bổ sung làm điều đó. Họ thuật toán giải mã này bao gồm thuật toán nổi tiếng là thuật toán Viterbi ngõ ra mềm (SOVA) và thuật toán ít được biết đến hơn là thuật toán Viterbi ngõ ra liệt kê nối tiếp (SLVA). Ngoài hai họ thuật toán giải mã này còn có một số kỹ thuật giải mã lặp khác.

Tuy cùng là các thuật toán ngõ ra mềm dựa trên sơ đồ trellis nhưng khác với VA là một thuật toán giải mã trellis ML và giảm thiểu xác suất lỗi từ mã, thuật toán MAP lại nhắm tới giảm tối đa xác suất lỗi bit. MAP là một phương pháp tối ưu để ước đoán các trạng thái và ngõ ra của các quá trình Markov trong điều kiện nhiễu trắng. Tuy nhiên MAP ít khả năng được ứng dụng thực tế bởi các khó khăn về số học liên quan đến việc biểu diễn xác suất, các hàm phi tuyến cùng một số các phép nhân và cộng khi tính toán các giá trị này.

Log-MAP là một biến thể của MAP, chất lượng gần như tương đương mà không gặp trở ngại trong việc ứng dụng trong thực tế. Log-MAP được thực hiện hoàn toàn trong miền logarit, nhờ đó phép nhân chuyển thành phép cộng và ta có được một hàm tương đối dễ thực hiện hơn.

Max-Log-MAP và SOVA là thuật toán gần tối ưu dùng để giảm bớt độ phức tạp tính toán nhưng trong kênh nhiễu Gauss thì chất lượng hai loại này cũng không cao, đặc biệ trong vùng SNR thấp. Max -Log-MAP hầu như giống với Log-MAP chỉ có duy nhất một điểm khác là sử dụng một hàm đơn giản hơn rất nhiều. Các nghiên cứu cho thấy Max-Log-MAP làm giảm chất lượng khoảng 0.5 dB so với MAP/Log-MAP trong kênh nhiễu Gauss.

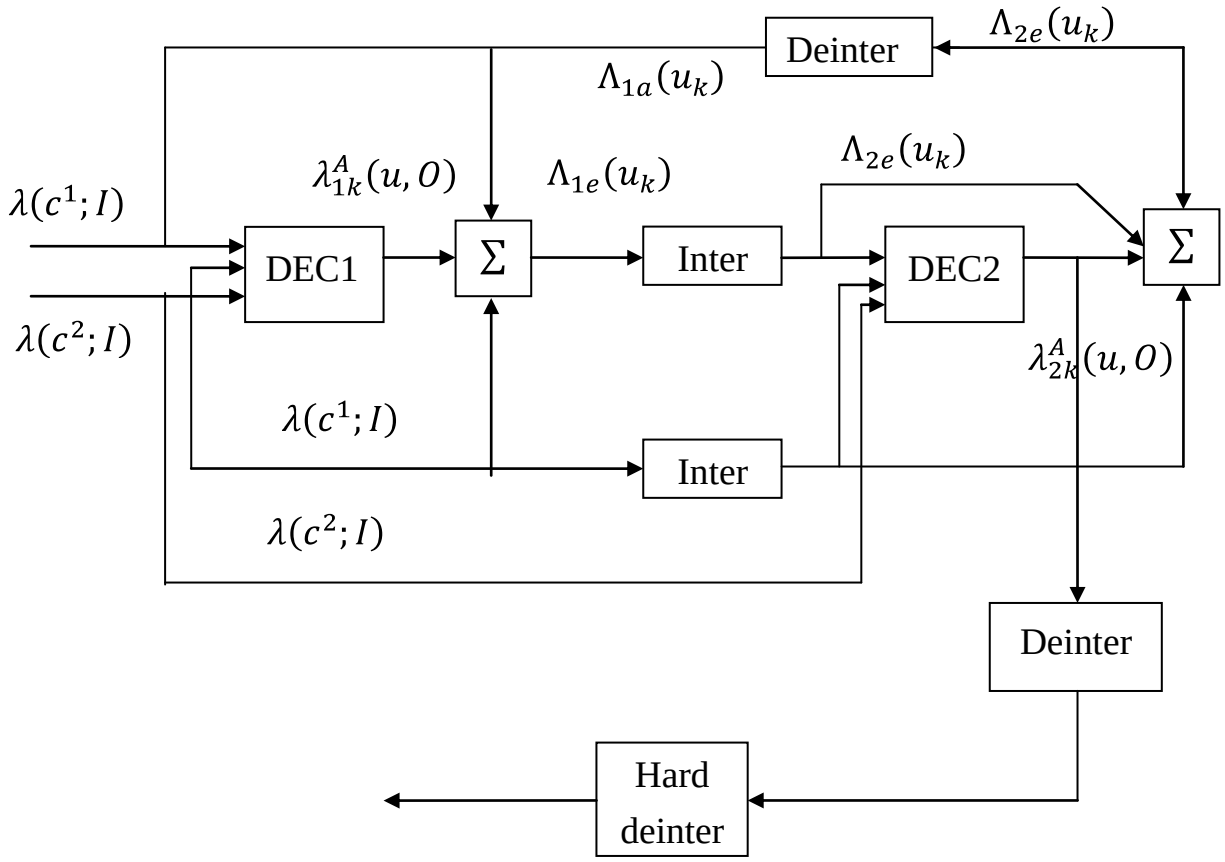
Các khác biệt trong việc thực hiện giữa các thuật toán giải mã này có thể giúp giải thích được sự khác biệt về chất lượng. Tại mỗi bước thứ k trong một trellis, MAP/Log-MAP chia tất cả các đường ra thành hai tập ; một tập các đường khi bit thông tin ngõ vào bằng 1 và một tập các đường khi bit thông tin ngõ vào bằng 0. MAP/Log-MAP sẽ tính tỉ số xác suất log (LLR) của hai tập này theo công thức. Ngược lại Max -Log-MAP sẽ tìm trong tất cả các đường để chọn các đường thích hợp, một đường có khả năng lớn nhất cho bit thông tin ngõ vào bằng 0. Ngõ ra mềm của Max-Log-MAP là LLR của hai đường này.

Còn SOVA thì bổ sung vào VA một số giá trị thực và lưu giữ. Thuật toán này chỉ tìm đường "tồn tại" và một đường cạch tranh với đường "tồn tại" đó. Về bản chất, SOVA sử dụng cùng một loại metric và có quyết định cứng như Max-log- MAP. Mặc dù, SOVA luôn tìm đường có khả năng lớn nhất nhưng đường cạnh tranh tốt nhất có thể bị loại ra trước khi kết hợp với đường ML. Kết quả là ngõ ra mềm của SOVA có thể bị sai đường so với ngõ ra mềm của Max-Log-MAP và chất lượng của bộ giải mã lặp SOVA kém hơn Max - Log-MAP.

Mặc dù thuật toán MAP tốt hơn thuật toán SOVA nhưng nó có cấu trúc phần cứng và quá trình tính toán giải mã lại phức tạp hơn nhiều.

3.3.2 Giải thuật MAP

Bộ giải mã là sự kết hợp của nhiều bộ giải mã (thường là hai bộ giải mã) và giải mã lặp (interatively). Phần lớn tập trung ở giải thuật Viterbi cung cấp giá trị ra mềm (soft output or reliability information) cho một bộ so sánh giá trị ra mềm được dùng để quyết định bit ngõ ra. Một giải thuật khác cũng được quan tâm là symbolby- *symbol Maximum A Posteriori (MAP)* của *Balh* được công bố



Hình 3.6: Bộ giải mã lặp MAP

Giải thuật giải mã được thực hiện như sau:

1.Tách tín hiệu nhận ra thành 2 chuỗi tương ứng cho bộ giải mã 1 và bộ giả mã 2

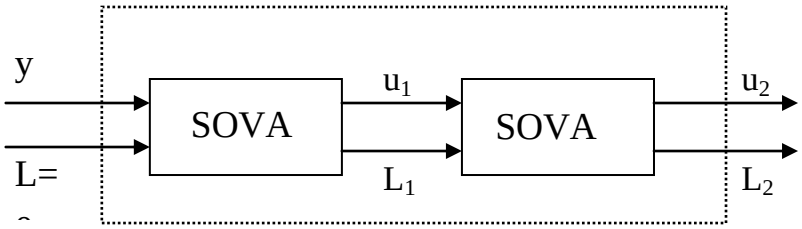
2.Ở vòng lặp đầu tiên,thông tin *a priori* (thông tin tiền nghiệm) của bộ giải mã 1 được đưa về 0. Sau khi bộ giải mã 1 đưa ra được thông tin *extrinsic* (thông tin ngoại lai)thì sẽ được chèn và đưa tới bộ giải mã 2 đóng vai trò là thông tin *a priori* của bộ giải mã này. Bộ giải mã 2 sau khi đưa ra thông tin *extrinsic* thì vòng lặp kết thúc.Thông tin *extrinsic* của bộ giải mã thứ 2 sẽ được giải chèn và đưa về bộ giải mã 1 như là thông tin *a priori*.

3. Quá trình giải mã giải mã cứ lặp lại như vậy cho đến khi thực hiện đủ số lần lặp đã qui định.

4. Sau vòng lặp cuối cùng, giá trị ước đoán có được tính bằng cách giải chèn thông tin ở bộ giải mã thứ 2 và đưa ra quyết định cứng.

3.3.3 Nguyên lý của bộ giải mã Viterbi ngõ ra mềm

Đối với các mã tích chập thì thuật toán Viterbi cho ra chuỗi ngõ ra ML. Còn đối với các mã Turbo, chúng ta gặp hai trở ngại khi sử dụng các bộ giải mã Viterbi thông thường. Thứ nhất, bộ giải mã Viterbi bên trong cho ra một loạt lỗi bit làm giảm đi việc thực hiện của các bộ giải mã Viterbi bên ngoài. Thứ hai, bộ giải mã Viterbi bên trong cho ra các ngõ ra quyết định cứng làm ngăn chặn bộ giải mã Viterbi bên ngoài nhận được các lợi điểm của các quyết định mềm. Cả hai trở ngại này có thể được khắc phục và việc thực hiện giải mã có thể được cải tiến một cách đáng kể nếu các bộ giải mã Viterbi có thể cho ra các giá trị tin cậy. Các giá trị tin cậy này đi qua các bộ giải mã Viterbi tiếp sau đó và được xem như là một thông tin ưu tiên nhằm để cải tiến việc thực hiện giải mã. Bộ giải mã Viterbi bổ sung này được tham khảo như là bộ giải mã thuật toán Viterbi ngõ ra mềm (SOVA)

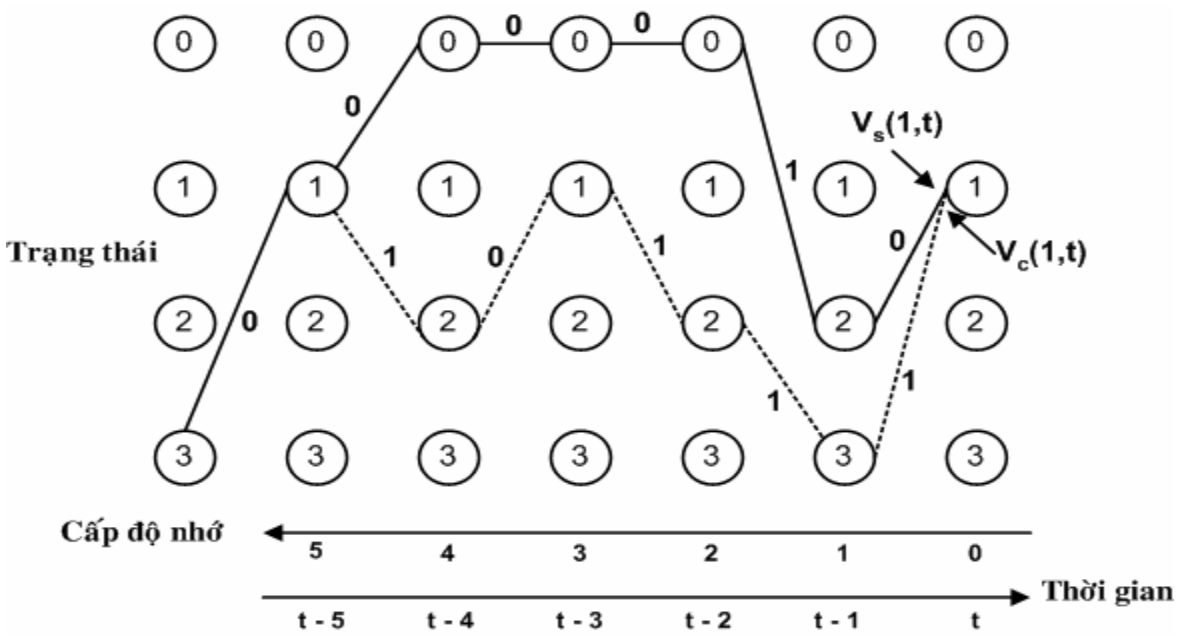


Hình 3.7 Bộ giải mã SOVA kết nối

Trong hình trên y biểu diễn các giá trị kênh nhận được, u biểu diễn các giá trị ngõ ra quyết định cứng, L biểu diễn các giá trị tin cậy liên kết.

3.3.3.1 Độ tin cậy của bộ giải mã SOVA tông quát

Độ tin cậy trong giải mã SOVA được tính toán từ biểu đồ trellis như hình 4.8



Hình 3.8: Các đường survivor và đường cạnh tranh để ước đoán độ tin cậy

Trong hình 3.8 trình bày biểu đồ Trellis 4 trạng thái. Đường liền nét chỉ ra đường survivor (giả thiết ở đây là một phần của đường ML) và đường đứt nét chỉ ra đường cạnh tranh (xảy ra đồng thời) tại thời điểm t đối với trạng thái 1. Để đơn giản thì các đường survivor và cạnh tranh cho các nút khác không được vẽ ra. Nhãn $S_{1,t}$ biểu diễn trạng thái 1 tại thời điểm t. Cũng vậy, các {0,1} được viết trên mỗi đường chỉ ra quyết định nhị phân được ước đoán cho các đường. Một metric tích lũy $V_s(S_{1,t})$ gán cho đường survivor đối với mỗi nút và metric tích lũy $V_c(S_{1,t})$ gán cho đường cạnh tranh đối với mỗi nút. Thông tin cơ bản cho việc gán giá trị tin cậy L(t) đến đường survivor của nút $S_{1,t}$ là giá trị tuyệt đối của 2 metric tích lũy.

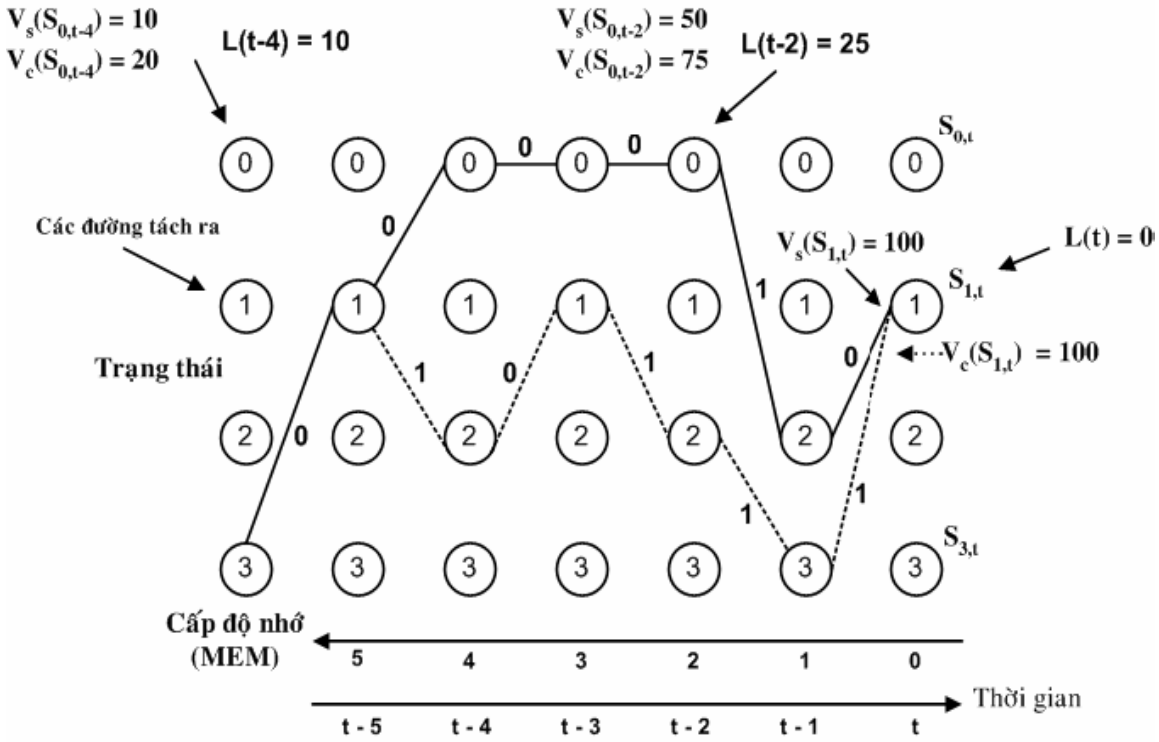
$$L(t) = |V_s(S_1,t) - V_c(S_1,t)| \quad (3.1)$$

Giá trị này càng lớn thì đường survivor càng đáng tin cậy. Để tính toán độ tin cậy này, giả thiết metric (số đo)tích lũy của survivor thì luôn luôn lớn hơn metric tích lũy của cạnh tranh. Để giảm độ phức tạp, các giá trị tin cậy chỉ cần được tính cho đường survivor ML và không cần thiết tính cho các đường survivor khác bởi vì chúng sẽ được bỏ qua sau này.

Để minh hoạ rõ hơn khái niệm độ tin cậy, hai ví dụ sau đây được đưa ra. Trong các ví dụ này, thuật toán Viterbi chọn đường survivor như là đường

có metric tích lũy nhỏ nhất. Trong ví dụ đầu tiên, giả thiết tại nút S(1,t) có metric survivor tích lũy là Vs(S1,t) = 50 và metric cạnh tranh tích lũy là Vc(S1,t) = 100. Giá trị tin cậy liên kết đến việc chọn đường survivor này là L(t) = |50 - 100| = 50. Trong ví dụ thứ hai, giả thiết metric survivor tích lũy không đổi Vs(S1,t)= 50 và metric cạnh tranh tích lũy là *Vc(S1,t) = 75.* Kết quả giá trị tin cậy là *L(t) = |50 - 75| = 25.* Mặc dù trong cả hai ví dụ này, đường survivor có cùng metric tích lũy, nhưng giá trị tin cậy được liên kết với đường survivor thì khác nhau. Giá trị tin cậy trong ví dụ đầu tiên có nhiều tin tưởng hơn (gấp 2 lần) trong việc chọn đường survivor hơn là giá trị trong ví dụ thứ hai. Hình 3.9 minh họa vấn đề sử dụng trị tuyệt đối giữa các metric survivor và cạnh tranh tích lũy như là phép đo độ tin cậy của quyết định.

Trong hình, các đường survivor và các đường cạnh tranh tại S1,t tách ra tại thời điểm t-5. Các đường survivor và các đường cạnh tranh cho ra các quyết định nhị phân ước đoán đối lập tại các thời điểm t, t - 2 và t - 4 như các chữ in đậm ở trong hình. Để minh họa, chúng ta giả thiết các metric tích lũy của survivor và cạnh tranh tại S1,t là bằng nhau, Vs(S1,t) = Vc(S1,t) = 100. Điều này có nghĩa là cả hai đường survivor và đường cạnh tranh có cùng xác suất là đường ML. Hơn nữa chúng ta giả thiết là metric tích lũy survivor thì tốt hơn metric tích lũy cạnh tranh tại thời điểm t - *2 và t - 4* được trình bày trong hình. Để giảm bớt độ phức tạp của hình vẽ, các đường cạnh tranh này tại các thời điểm t - 2 và t - 4 không đưa ra. Từ giả thiết này, chúng ta thấy rằng giá trị tin cậy gán cho đường survivor tại thời điểm t là L(t) = 0, điều này có nghĩa là không có độ tin cậy liên kết với việc chọn đường survivor. Tại các thời điểm t - 2 và t - 4, các giá trị tin cậy gán cho đường survivor thì lớn hơn 0 (L(t-2) = 25 và L(t-4) = 10) nghĩa là kết quả các metric tích lũy "tốt hơn" cho đường survivor. Tuy nhiên, tại thời điểm t, đường cạnh tranh cũng có thể là đường survivor bởi vì chúng có cùng metric. Vì vậy có thể có các quyết định nhị phân được ước đoán trái ngược nhau tại các thời điểm t, t- 2, t - 4 mà không làm giảm các giá trị tin cậy liên kết suốt dọc theo đường survivor.



Hình 3.9 : Ví dụ trình bày việc gán độ tin cậy bằng cách sử dụng các giá trị metric trực tiếp

Để cải tiến các giá trị tin cậy của đường survivor, một phép tính truy ngược để cập nhật các giá trị tin cậy được giả thiết. Thủ tục cập nhật này được tích hợp vào trong thuật toán Viterbi như sau :

* Đối với nút $S_{k,t}$ trong biểu đồ trellis (đáp ứng đến trạng thái k tại thời điểm t),lưu $L(t) = | V_s(S_{1,t}) - V_c(S_{1,t})|$.

* Nếu có nhiều hơn một đường cạnh tranh, thì sau đó nhiều giá trị tin cậy phải được tính và giá trị tin cậy nhỏ nhất được lấy là L(t)

* Khởi tạo giá trị tin cậy $S_{k,t}$ bằng $+\infty$ (tin cậy nhất)

* So sánh các con đường survivor và cạnh tranh tại $S_{k,t}$ và lưu lại các cấp độ nhớ (MEM) trong đó các quyết định nhị phân được ước đoán của hai con đường là khác nhau.

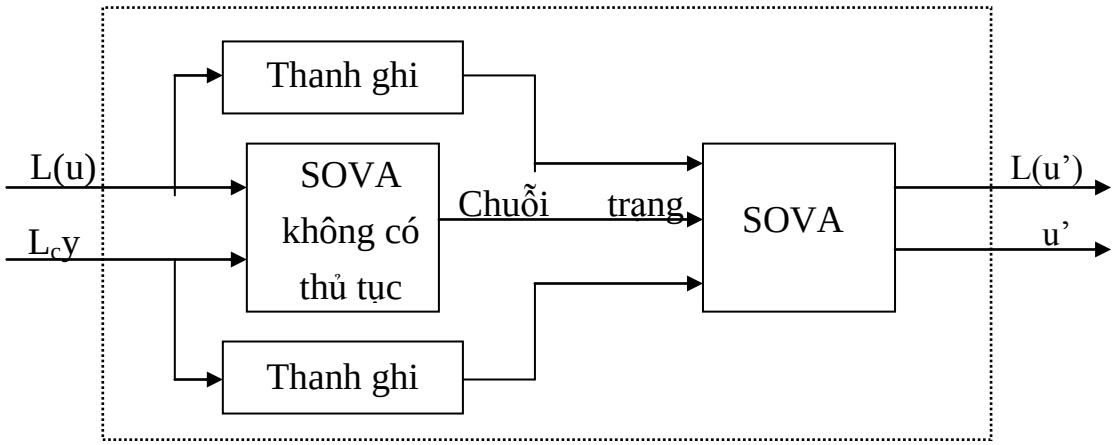
* Cập nhật các giá trị tin cậy tại các MEM này với thủ tục như sau :

+ Tìm MEM thấp nhất lớn hơn 0, coi như là MEM_{low} mà giá trị tin cậy của nó không được cập nhật.

+ Cập nhật giá trị tin cậy của MEM_{low} $L(t\text{-}MEM_{low})$ bằng cách gán giá trị tin cậy thấp nhất giữa $MEM = 0$ và $MEM = MEM_{low}$

3.3.3.2 Sơ đồ khối của bộ giải mã SOVA

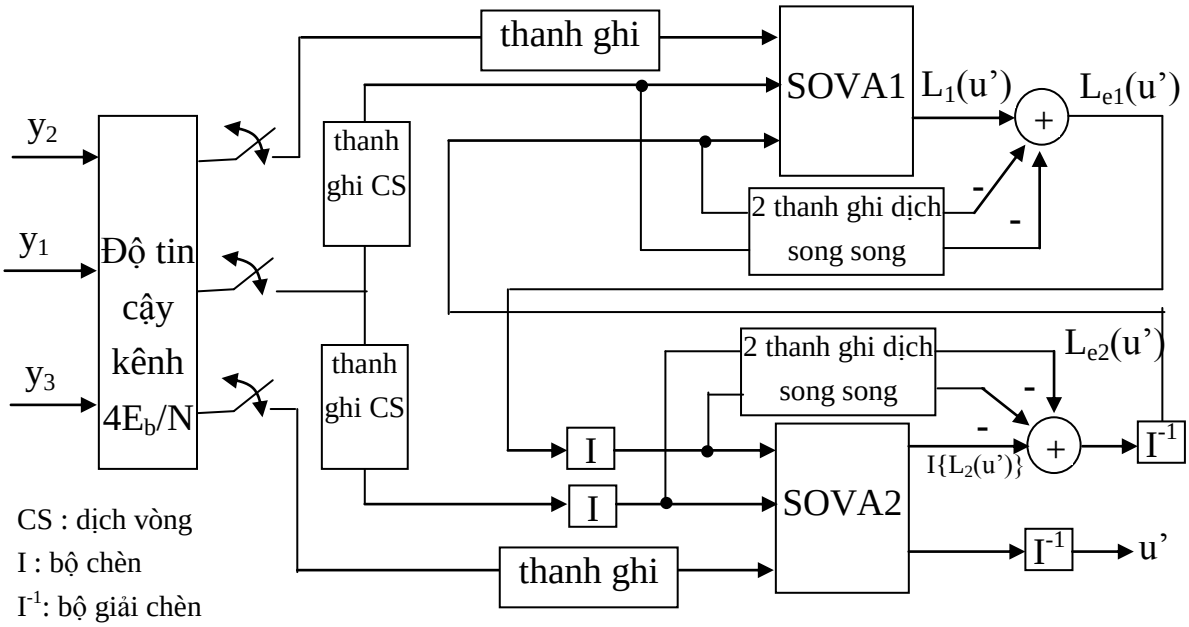
Bộ giải mã SOVA có thể được thực hiện theo nhiều cách khác nhau. Nhưng có lẽ theo khuynh hướng tính toán thì dễ dàng thực hiện bộ giải mã SOVA cho các mã có chiều dài bắt buộc K lớn và kích cỡ khung dài bởi vì sự cần thiết cập nhật tất cả các đường survivor. Do thủ tục cập nhật chỉ có ý nghĩa cho đường ML, nên việc thực hiện của bộ giải mã SOVA chỉ thực hiện thủ tục cập nhật đối với đường ML được trình bày trong hình 3.10



Hình 3.10: Sơ đồ khối bộ giải mã SOVA

Bộ giải mã SOVA lấy ngõ vào là L(u) và Lcy, là giá trị tin cậy và giá trị nhận được đã qua cân bằng tương ứng, và cho ra u' và L(u'), tương ứng là các quyết định bit ước đoán và các thông tin a posteriori L(u'). Việc thực hiện bộ giải mã SOVA này bao gồm hai bộ giải mã SOVA riêng biệt. Bộ giải mã SOVA đầu tiên chỉ tính các metric của đường ML và không tính (giữ lại) các giá trị tin cậy. Các thanh ghi dịch được sử dụng để đệm cho các ngõ vào trong khi bộ giải mã SOVA đầu tiên đang xử lý đường ML. Bộ giải mã SOVA thứ hai (có thông tin đường ML) tính lại đường ML và cũng tính và cập nhật các giá trị tin cậy. Ta thấy rằng phương pháp thực hiện này làm giảm độ phức tạp trong tiến trình cập nhật. Thay vì truy ngược và cập nhật 2m đường survivor, thì chỉ có đường ML cần được xử lý.

Một sơ đồ chi tiết của một bộ giải mã SOVA lặp được trình bày ở hình 3.11



Hình 3.11: Bộ giải mã SOVA lặp

Bộ giải mã xử lý các bit kênh nhận được trên một khung cơ bản. Như được trình bày trong hình 3.11, các bit kênh nhận được tách thành dòng bit hệ thống y_1 và *2* dòng bit parity y_2 và y_3 từ các bộ mã hóa 1 và 2 tương ứng. Các bit này được cân bằng bởi giá trị tin cậy kênh và được lấy ra qua các thanh ghi CS. Các thanh ghi trình bày trong hình được sử dụng như các bộ đệm để lưu trữ các chuỗi cho đến khi chúng ta cần. Các khóa chuyển được đặt ở vị trí mở nhằm ngăn ngừa các bit từ các khung kế tiếp đợi xử lý cho đến khi khung hiện hành được xử lý xong.

Bộ giải mã thành phần SOVA cho ra thông tin *a posteriori* $L(u_t')$ và bit được ước đoán u_t' (ở thời điểm *t*). Thông tin *a posteriori* $L(u_t')$ được phân tích thành 3 số hạng

$$L(u'_t)=L(u_t) + L_c y_{t,1} + L_e(u_t') \quad (3.2)$$

$L(u_t)$ là giá trị *a priori* và được sinh ra bởi bộ giải mã thành phần SOVA trước đó.

$L_c y_{t,1}$ là giá trị kênh hệ thống nhận được đã qua cân bằng.

$L_e(u_t')$ là giá trị *extrinsic* được sinh ra bởi bộ giải mã thành phần SOVA hiện tại. Tin tức đi xuyên qua giữa các bộ giải mã thành phần SOVA là giá trị *extrinsic*.

$L_e(u_t')=L(u'_t) – L_c y_{t,1} – L(u_t)$ (3.3)

Giá trị *a priori* $L(u_t)$ được trừ đi từ số bị trừ là thông tin *a osteriori* $L(u_t')$ để ngăn ngừa tin tức đi ngược lại bộ giải mã mà từ đó sinh ra nó. Cũng vậy, giá trị kênh hệ thống nhận được đã qua cân bằng $L_c y_{t,1}$ được trừ đi nhằm để xóa tin tức "thông thường" trong các bộ giải mã thành phần SOVA. Hình 4.11 trình bày bộ giải mã mã PCCC là sự kết nối theo thứ tự vòng kín của các bộ giải mã thành phần SOVA. Trong sơ đồ giải mã vòng kín này, mỗi một bộ giải mã thành phần SOVA ước đoán chuỗi tin bằng cách sử dụng dòng bit parity đã qua cân bằng. Hơn nữa, bộ giải mã PCCC thực hiện giải mã lặp nhằm cho ra các ước đoán *a priori* /độ tin cậy đáng tin tưởng hơn từ 2 dòng bit parity đã qua cân bằng khác nhau, với hy vọng thực hiện giải mã tốt hơn. Thuật toán mã Turbo lặp với lần lặp thứ n như sau:

1. Bộ giải mã SOVA1 có ngõ vào là chuỗi $L_c y_1$(hệ thống), $L_c y_2$ (parity), và cho ra chuỗi $L_{e2}(u')$. Đối với lần lặp đầu tin, chuỗi $L_{e2}(u')=0$ bởi vì không có giá trị *a priori* (không có giá trị *extrinsic* từ SOVA2). Thông tin *extrinsic* từ SOVA1 được tính bằng

$L_{e1}(u')= L_1(u') - L_{e2}(u')- L_c y_1$ (3.4)

trong đó : $Lc = 4\frac{E_b}{N_o \times rate}$

2. Các chuỗi $L_c y_1$ và $L_{e1}(u')$ được chèn là $I\{(L_c y_1)\}$ và $I\{L_{e1}(u')\}$.

3. Bộ giải mã SOVA2 có ngõ vào là các chuỗi $L_c y_1$ (hệ thống), và $I(L_c y_3)$(parity đã được chèn ở bộ giải mã) và $I\{L_{e1}(u')\}$ (thông tin *a priori*) và cho ra các chuỗi $I\{L_2(u')\}$ và $I\{u'\}$.

4.Thông tin *extrinsic* từ SOVA2 được lấy là:

$I\{L_{e2}(u')\} = I\{L_2(u')\} - I\{L_{e1}(u')\} - I(L_c y_1)$

Các chuỗi $I\{L_{e2}(u')\}$ và $I\{u'\}$ được giải chèn và là $L_{e2}(u')$ và u'.

5. $L_{e2}(u')$ được hồi tiếp về SOVA1 như là thông tin *a priori* cho lần lặp kế tiếp và u' là ngõ ra của các bit được ước đoán cho lần lặp thứ *n*.

Ch- ơng 4

ỨNG DỤNG MÃ TURBO TRONG HỆ THÔNG THÔNG TIN DI ĐỘNG

4.1 GIỚI THIỆU CH- ƠNG

Trong lĩnh vực viễn thông thì hai hệ thống gây nhiều khó khăn nhất là truyền thông không dây (Wireless Communication) và truyền thông đa phương tiện (Multi Media Communication - MMC) do một số điểm đặc thù của hai loại hệ thống này gây nhiều khó khăn cho việc truyền thông. Mã Turbo ra đời đã thúc đẩy một quá trình tìm tòi, phát triển mới nhờ những đặc tính ưu việt của nó. chương này trình bày các ứng dụng chung của mã Turbo trong hệ thống truyền thông và trình bày chi tiết ứng dụng trong hệ thống thông tin di động CDMA 2000

4.2 CÁC ỨNG DỤNG TRUYỀN THÔNG ĐA PH- ƠNG TIỆN

Ứng dụng trong truyền thông đa phương tiện là đề tài mới được nghiên cứu gần đây. Vì thế có một số nét chính về các vấn đề gặp phải và một số đề nghị khi ứng dụng TC trong truyền thông đa phương tiện.

4.2.1 Các hạn chế khi ứng dụng mã Turbo vào truyền thông đa ph- ơng tiện

Các ứng dụng MMC gặp phải các ràng buộc sau đây :

4.2.1.1. Tính thời gian thực

Một hạn chế quan trọng nhất của các ứng dụng MMC là thời gian eo hẹp. Ví dụ như xét một ứng dụng MMC là Video-On-Demand (VOD), máy chủ VOD truyền dữ liệu phim đến các khách hàng. Mỗi khung dữ liệu có thể là thông tin về một khung hình của bộ phim. Nếu các dữ liệu phim đến chậm thì khách hàng sẽ có cảm giác chất lượng phim không tốt, phim không được chiếu trơn tru. Từ đây ta có thể thấy dữ liệu multimedia có bản chất thời gian thực, sự chậm trễ của dữ liệu sẽ làm mất giá trị của thông tin. Vấn đề thời gian chính là một rào cản lớn trong các ứng dụng MMC.

Trong khi TC cần phải có một cấu trúc giải mã lặp để tăng chất lượng thì tính thời gian thực quả là một thách thức khi phải giải quyết mâu thuẫn giữa thời gian đáp ứng và tỉ lệ lỗi bit (BER). Đặc tính thời gian thực này cho thấy các mã Turbo ứng dụng trong MMC không thể có số vòng lặp lớn.

4.2.1.2. Khối lượng dữ liệu lớn:

Một đặc tính khác của các ứng dụng MMC là các khối dữ liệu lớn. Chỉ cần một hình ảnh trong bộ phim cũng cần tới cỡ hàng Megabit để biểu diễn. Cộng với đặc tính thời gian thực thì một số lượng lớn các dữ liệu sẽ phải được xử lý trong một khoảng thời gian giới hạn và rất ngắn, nếu không hệ thống sẽ gây lỗi. Kết quả là yêu cầu đối với các bộ mã hóa và giải Mã Turbo rất cao.

4.2.1.3. Băng thông giới hạn:

Băng thông là vấn đề luôn được quan tâm hàng đầu, nhất là trong các ứng dụng thực tiễn trong thời gian gần đây vì lượng thông tin con người mong muốn được truyền tải ngày càng lớn mà một tài nguyên quốc gia như băng thông không thể tăng. Băng thông sử dụng trong các ứng dụng MMC rất lớn (ví dụ như VOD thường sử dụng ATM), tuy nhiên do khối lượng dữ liệu cần truyền lớn nên băng thông lớn (so với các ứng dụng trong hệ thống khác) vẫn trở thành một giới hạn cho các ứng dụng MMC. Kết quả là các mã tốc độ thấp sẽ không hiệu quả.

4.2.1.4. Tìm hiểu các đặc tính của kênh truyền:

Các đặc tính của các kênh truyền trong MMC đơn giản và bất biến hơn nhiều so với môi trường không dây. Vì vậy ta có thể tìm hiểu được các đặc tính của kênh truyền và đưa ra các giải pháp thích hợp cho từng hệ thống. Một phương pháp để tìm hiểu các đặc tính của kênh truyền là dùng mạng Bayes. Các phương pháp thực hiện có thể tóm lược như sau :

* Dữ liệu được truyền qua kênh truyền và sử dụng nhiều loại mô hình Mã Turbo với các thông số khác nhau.

* Ghi lại các giá trị BER

* Thành lập một mạng Bayes với các độ chính xác của mã kết quả và các yếu tố ảnh hưởng là các nút mạng. Một đường nối trực tiếp sẽ đi từ các yếu tố đến các độ chính xác của mã kết quả.

* Sử dụng các giá trị BER ghi nhận để thử cho mạng này.

* Tìm ra một tập hợp các thông số để tối ưu hóa các sự điều chỉnh

4.2.2 Các đề xuất khi ứng dụng mã Turbo vào truyền thông đa ph- ơng tiện

4.2.2.1.Kích thước khung lớn:

Như đã đề cập ở trên, một đặc tính quan trọng của ứng dụng MCC là khối dữ liệu lớn. Từ đây gợi ra ý tưởng sử dụng kích thước khung lớn cho mã Turbo. Kích thước khung lớn đồng nghĩa với kích thước bộ chèn lớn và sẽ làm tăng đáng kể chất lượng của mã Turbo.

Với một băng thông lớn như của MMC thì một khối lượng dữ liệu lớn có thể truyền với một độ trễ chấp nhận được. Với kích thước khung lớn này độ lợi mã của TC có thể tăng bằng các cách sau :

* Giảm BER của kênh truyền

* Tăng thời gian đáp ứng bằng cách giảm số lần lặp giải mã hay sử dụng một số cải tiến giải mã trình bày dưới đây.

4.2.2.2.Cải tiến quá trình giải mã:

4.2.2.2.1 Giải mã động:

Phương pháp giải mã động gói gọn trong hai điểm sau :

* Đặt một ngưỡng vòng lặp, tức là số lần lặp tối đa cho một khung.

* Số vòng lặp thực sự để giải mã một khung sẽ nhỏ hơn hay bằng giá trị ngưỡng này và phụ thuộc vào kết quả giải mã. Điều kiện để ngưng quá trình giải mã là khung đã hết lỗi. Trong quá trình giải mã, kết quả giá trị ước lượng của vòng lặp giải mã trước được lưu lại và so sánh với kết quả của vòng lặp giải mã kế tiếp. Nếu hai kết quả giống nhau thì hết lỗi và tiếp tục giải mã cho khung kế tiếp.

Ý tưởng ở đây là một số khung chỉ cần số vòng lặp rất ít (chỉ khoảng 2 hay 3 vòng) đã loại bỏ hoàn toàn lỗi sai, trong khi một số khung khác rất nhiều lỗi thì cần số vòng lặp giải mã nhiều hơn để đạt được chất lượng cao hơn. Vì thế số vòng lặp thay đổi sẽ ảnh hưởng trực tiếp đến việc giảm độ trễ và có thể còn làm tăng chất lượng. Ví dụ như một hệ thống sử dụng số lần lặp giải mã cố định là 10. Khi sử dụng hệ thống này với phương pháp giải mã động có số vòng lặp tối đa là 15 thì số vòng lặp giải mã trung bình sẽ giảm rất nhiều, chỉ khoảng 5 -7 vòng. Như vậy ta đã tiết kiệm được rất nhiều thời gian,

tăng thời gian đáp ứng của hệ thống. Thậm chí có một số khung nhiều lỗi sai thì giải mã lặp đến 15 vòng có thể sẽ cho chất lượng cao hơn chỉ lặp 10 vòng cố định.

4.2.2.2.2 Giải mã ưu tiên:

Khối dữ liệu được truyền ngoài đặc tính là có số lượng bit lớn còn có một số đặc tính khác như :

* Dữ liệu nhận không cần chính xác 100%. Ví dụ như trong VOD, nếu một số phần nào đó của các khung nhận bị lỗi thì có thể gây ra một số suy giảm chất lượng trên một vài phần nào đó trong hình ảnh bộ phim. Nhưng nếu những sự suy giảm này khá nhỏ thì mắt người cũng khó nhận biết hoặc dễ dàng chấp nhận. Điều đó có nghĩa là MMC có thể chấp nhận một mức lỗi nhất định.

* Các dữ liệu truyền có tầm quan trọng khác nhau. Cũng xét ví dụ trên, nếu các lỗi xảy ra trong ở vùng trung tâm của hình ảnh thì khách hàng có thể phát hiện dễ dàng. Nhưng nếu các lỗi gây sự suy giảm chất lượng ở các vùng lân cận biên của hình ảnh thì khó gây sự chú ý hơn. Điều đó có nghĩa là các dữ liệu có tầm quan trọng thấp sẽ chấp nhận mức lỗi cao hơn.

Các đặc tính này làm nảy sinh thêm một ý tưởng là giải mã theo mức ưu tiên. Các ứng dụng MMC sẽ thêm các thông tin về độ ưu tiên vào trong khung tùy theo tầm quan trọng của khung. Sau khi nhận được chuỗi tin từ kênh truyền, bộ giải mã sẽ giải mã tìm các từ mã. Sau vòng lặp đầu tiên bộ giải mã có thể nhận được thông tin về mức độ ưu tiên của khung và sẽ quyết định số vòng lặp (hay phương pháp lặp) phù hợp với khung.

Theo mô hình giải mã này, lượng thời gian tiết kiệm được từ các khung có độ ưu tiên thấp sẽ được dùng để :

* Giảm BER của các khung có độ ưu tiên cao

* Tăng tốc độ đáp ứng của hệ thống nhờ giảm được số vòng lặp cho các khung có độ ưu tiên thấp.

Mô hình này không làm tăng chất lượng trung bình của hệ thống. Tỉ số BER có thể lớn hơn hay nhỏ hơn so với các phương pháp giải mã khác nhưng hiệu quả thực tế thì hơn hẳn. Ví dụ như trong trường hợp cụ thể trên thì hình ảnh sẽ được khách hàng đánh giá là tốt hơn.

4.3 CÁC ỨNG DỤNG TRUYỀN THÔNG KHÔNG DÂY

Truyền thông không dây ngày càng trở nên thông dụng nhờ những ưu điểm như số lượng dịch vụ lớn, kích thước thoại nhỏ và giá cả tương đối chấp nhận được so với những lợi ích của nó. Không như các tiến bộ vượt bậc về kỹ thuật trong kích thước thoại và số lượng dịch vụ, các giao thức hiện nay như GSM, CDMA vẫn sử dụng các mô hình đơn giản như các mã tích chập. Với tốc độ phát triển như hiện nay, các thành phần này sẽ được thay thế bằng loại mã chất lượng hơn như mã Turbo mà gần đây nhất là các hệ thống thông tin thế hệ thứ ba (CDMA 200).

Đặc biệt trong truyền thông không dây còn phải kể đến truyền thông vệ tinh hay thám hiểm vũ trụ. Hiện nay đã có xu hướng gia tăng cả về số lượng lẫn chất lượng các loại hình thông tin vệ tinh cũng như thông tin vũ trụ do khoa học kỹ thuật đã tiến bộ ở rất nhiều nước. Các hệ thống tiêu biểu cho thông tin vệ tinh là hệ thống định vị toàn cầu (GPS), hệ thống thông tin địa lý (GIS), hệ thống truyền hình qua vệ tinh.

4.3.1 Các hạn chế khi ứng dụng mã Turbo trong truyền thông không dây

4.3.1.1.Kênh truyền:

Đối với nhiều kênh truyền thì mô hình kênh AWGN với nhiễu tĩnh rất thích hợp. Tuy nhiên, trong môi trường không dây thì thường không tĩnh do có fading của các tín hiệu truyền. Fading là hậu quả bản chất vật lý của kênh truyền với độ tăng biên độ là một quá trình ngẫu nhiên biểu diễn bởi một hàm mật độ xác suất và một hàm tự tương quan.

Trong kênh AWGN, các bit chỉ bị tác động bởi nhiễu :

$Y_k = ax_k + n_k$ *với* n_k là nhiễu *và* $a = 1$ đối với kênh *AWGN*

Trong fading Rayleigh, các từ mã bị tác động bởi cả nhiễu và fading biến đổi theo thời gian trong kênh vô tuyến di động.

$y_k = ax_k + n_k$ *với* n_k là nhiễu *và* a là một biến ngẫu nhiên của phân bố fading Rayleigh.

Phân bố fading Rayleigh thường được sử dụng để mô tả bản chất thay đổi theo thời gian theo thống kê của đường bao nhận được của một tín hiệu fading phẳng, hay đường bao của một thành phần riêng lẻ trong hệ thống đa đường. Phân bố Rayleigh là một hàm mật độ xác suất cho bởi :

$$P(r) = \frac{r}{\sigma^2} e^{-\frac{r^2}{\sigma^2}} với\ r < 0$$

$$P(r) = 0\ với\ r \geq 0$$

Kênh truyền trong truyền thông không dây có mức nhiễu cao hơn ở môi trường truyền dây. Vì thế các mã kênh phải có đủ khả năng đương đầu với mức nhiễu lớn. Đặc biệt nếu dùng trong công tác nghiên cứu vũ trụ thì mức nhiễu còn cao hơn nữa.

Để triệt fading thì còn có nhiều cách khác nhau ví dụ như trải phổ. Khi đã triệt được một phần fading và sử dụng thêm mã Turbo nữa thì chất lượng đạt được sẽ rất cao.

Ngoài ra, môi trường truyền còn luôn luôn biến đổi. Ví dụ như một thuê bao điện thoại di động có thể vừa đàm thoại vừa di chuyển, môi trường truyền xung quanh cũng biến đổi, thông số môi trường cũng thay đổi. Chính vì sự bất ổn định của kênh truyền mà việc tìm được một loại mã thích hợp là một việc rất khó khăn. Và đây chính là lĩnh vực ứng dụng chủ yếu của TC nhờ các đặc tính ưu việt.

4.3.1.2. Hạn chế về thời gian:

Cũng như các ứng dụng thời gian thực khác, truyền thông không dây cũng có những yêu cầu về thời gian rất khắt khe. Nhất là các thông tin thoại yêu cầu phải đáp ứng nhanh. Thông tin thoại mà đáp ứng chậm sẽ trở nên vô giá trị.

4.3.1.3. Kích thước khung nhỏ:

Trong truyền thông không dây thì kích thước khung truyền không được lớn vì:

* Kênh truyền không tin cậy, nếu truyền khung lớn thì tỉ lệ lỗi trong khung sẽ cao hơn. Nếu khung bị mất hay không thể khôi phục thì dữ liệu tại đầu nhận sẽ bị mất.

* Do đặc tính thời gian thực nên không chấp nhận độ trễ lớn khi truyền một khung có kích thước lớn.

Như vậy, với kích thước khung nhỏ thì không tận dụng được các đặc tính ưu việt của TC.

4.3.1.4. Băng thông giới hạn:

Truyền thông không dây chỉ sử dụng một khoảng phổ tần số đã được phân, mỗi công ty điện thoại di động lại chỉ được phân cho một khu vực trong khoảng này để cung cấp dịch vụ cho khách hàng. Như vậy băng thông rất hạn chế 0có nghĩa là mô hình mã hóa phải có càng ít bit redundant càng tốt, tức là đòi hỏi tốc độ mã cao.

4.4 MÃ HÓA TURBO TRONG CDMA 2000

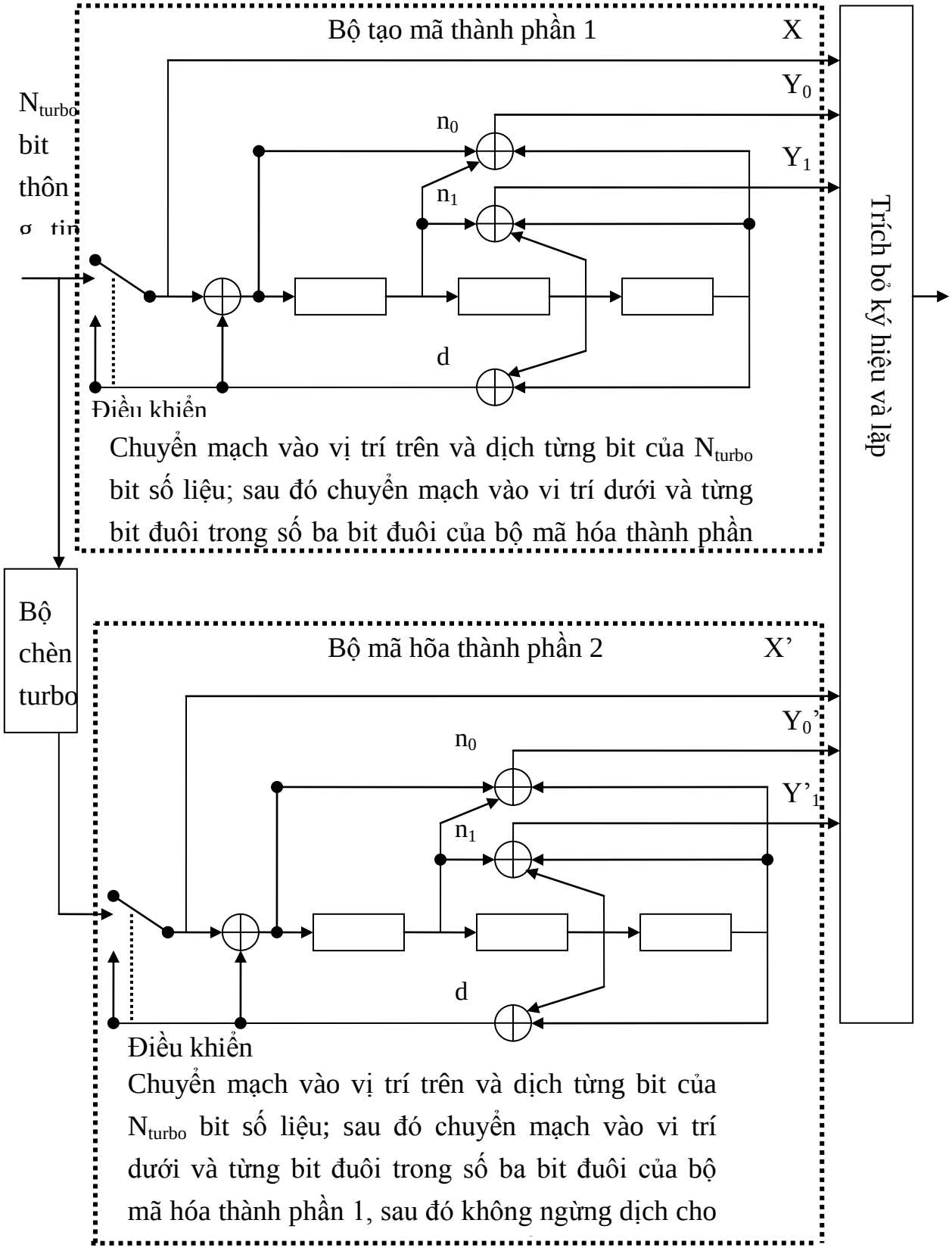
Bộ mã hóa turbo thực hiện mã hóa số liệu, chỉ thị chất lượng khung (CRC) và hai bit dành trước cho mã Turbo và cộng chuỗi đuôi mã hóa đầu ra. Nếu tổng các bit số liệu, các bit chất lượng khung và các bit dành trước là N_{turbo}, thì bộ mã hóa tạo ra N_{turbo}/R các ký hiệu số liệu cùng với 6/R các ký hiệu đuôi ở đầu ra, trong đó R là tỷ lệ mã bằng 1/2, 1/3 hay 1/4. Bộ mã hóa turbo sử dụng hai bộ mã hóa tích chập hệ thống, đệ quy mắc song song kết hợp với bộ chèn, trong đó bộ chèn đứng trước bộ mã tích chập thứ hai, hai mã tích chập đệ quy này được gọi các mã thành phần của mã Turbo. Các đầu ra của các bộ mã hóa thành phần được trích bỏ và được lặp để đạt được (N_{turbo} +6)/R các ký hiệu ra

4.4.1 Các bộ mã hóa turbo tỷ lệ 1/2, 1/3, 1/4 :

Một mã thành phần chung được sử dụng cho các mã Turbo tỷ lệ 1/2, 1/3, và 1/4. Hàm truyền đạt của mã này có dạng sau:

$$G(D)=\left[1 \frac{n_0(D)}{d(D)} \frac{n_1(D)}{d(D)}\right] \quad (3.1)$$

Trong đó: $d(D) = 1 + D^2 + D^3$, $n_0(D) = 1+D + D^3$ và $n_1(D) = 1 +D + D^2 + D^3$. bộ tạo mã Turbo này sẽ tạo ra chuỗi ký hiệu đầu ra giống như chuỗi được tạo ra bởi bộ mã cho ở hình 4.1


Bộ tạo mã thành phần 1
X
Y_0
Y_1
N_{turbo} bit thôn g tin
n_0
n_1
d
Điều khiển
Chuyển mạch vào vị trí trên và dịch từng bit của N_{turbo} bit số liệu; sau đó chuyển mạch vào vi trí dưới và từng bit đuôi trong số ba bit đuôi của bộ mã hóa thành phần
Trích bỏ ký hiệu và lặp
Bộ chèn turbo
Bộ mã hõa thành phần 2
X'
Y_0'
Y'_1
n_0
n_1
d
Điều khiển
Chuyển mạch vào vị trí trên và dịch từng bit của N_{turbo} bit số liệu; sau đó chuyển mạch vào vi trí dưới và từng bit đuôi trong số ba bit đuôi của bộ mã hóa thành phần 1, sau đó không ngừng dịch cho

Hình 4.1

Khởi đầu các trạng thái của các thanh ghi dịch trong các bộ mã hóa thành phần được đặt về "0". Sau đó, các bit được dịch vào các bộ mã hóa thành phần theo vị trí của các chuyển mạch trên hình vẽ. mạch thay đổi chu kỳ từng bit số liệu và bit đuôi.

Các ký hiệu ra của số liệu sau mã hóa được tạo ra bằng cách dịch các bộ mã hóa thành phần N_{turbo} lần khi các khóa ở vị trí trên và trích bỏ các đầu ra theo như quy định ở bảng 4.1**.** '0' ở mẫu trích bỏ có nghĩa là ký hiệu này sẽ bị xóa và '1' có nghĩa là ký hiệu này được cho qua. Đối với mỗi bit vào, đầu ra của các bộ lập mã thành phần sẽ được đặt vào chuỗi X, Y_0,Y_1, X', Y_0', Y_1'. Trong quá trình tạo ra các ký hiệu từ số liệu vào mã hóa sẽ không thực hiện lặp

4.4.2 Kết cuối mã Turbo:

Bộ mã hóa turbo tạo ra 6/R các ký hiệu đuôi đầu ra tiếp sau các ký hiệu của các bit số liệu được mã hóa. Chuỗi ký hiệu đuôi đầu ra cũng giống như chuỗi được bộ mã hóa tạo ra ở hình 4.1. Các ký hiệu ra được tạo ra sau khi N_{turbo} bit được dịch vào các bộ mã hóa thành phần với các khóa ở vị trí trên. 3/R ký hiệu đuôi ra đầu tiên được tạo ra bằng cách dịch bộ mã hóa thành phần 3 lần với khóa tương ứng ở vị trí dưới và đồng thời trích bỏ cũng như lặp các ký hiệu ra của bộ mã hóa thành phần này. 3/R các ký hiệu đuôi ra nhận được bằng cách dịch bộ mã hóa thành phần 2 ba lần với khóa tương ứng của nó ở vị trí dưới quá trình này kết hợp với trích bỏ và lặp các ký hiệu đầu ra của bộ mã hóa thành phần này. Các đầu ra của các bộ mã hóa thành phần đới với từng chu kỳ bit đuôi sẽ được đặt vào chuỗi X, Y_0, Y_1, X', Y'_0, Y'_1 với X ra trước.

Mẫu trích bỏ và lặp ký hiệu ra của bộ mã hóa thành phần được quy định ở bảng 4.2**.** Trong mẫu trích bỏ, '0' nghĩa là ký hiệu bị xóa còn '1' nghĩa là ký hiệu được cho qua. Đối mã Turbo 1/2, các ký hiệu đuôi ra đối với 3 chu kỳ bit đuôi đầu tiên sẽ là XY_0 còn các ký hiệu đuôi ra đối với ba chu kỳ bit còn lại sẽ là $X'Y'_0$. Đối với mã Turbo 1/3, các ký hiệu đuôi ra đối với 3 chu kỳ bit đuôi đầu tiên sẽ là XXY_0 còn các ký hiệu đuôi ra đối với ba chu kỳ bit đuôi còn lại sẽ là $X'X'Y'_0$. Đối với mã Turbo 1/4, các ký hiệu đuôi ra đối với 3 chu bit đuôi đầu tiên sẽ là XXY_0Y_1 còn các ký hiệu đuôi đối với 3 chu kỳ bit còn lại sẽ là $X'X'Y'_0Y_1$.

4.4.3. Các bộ chèn Turbo:

Bộ chèn turbo là một bộ phận của bộ mã hóa turbo có nhiệm vụ chèn khối cho số liệu, chỉ thị chất lượng khung (CRC) và các bit dành trước nhận được ở đầu vào của bộ mã hóa Turbo.

Bộ chèn turbo hoạt động như sau. Toàn bộ chuỗi bit đầu vào của bộ chèn turbo được viết vào ma trận nhớ lần lượt theo một trình tự các địa chỉ và sau đó toàn bộ chuỗi này được đọc ra từ bộ nhớ theo một trình tự các địa chỉ được xác định theo thủ tục trình bày dưới đây

Đầu ra	Tỷ lệ mã		
	1/2	1/3	1/4
X	11	11	11
Y_0	10	11	11
Y_1	00	00	10
X’	00	00	00
Y'_0	01	11	01
Y'_1	00	00	11

Bảng 4.1. Mẫu trích bỏ cho các chu kỳ của bit số liệu

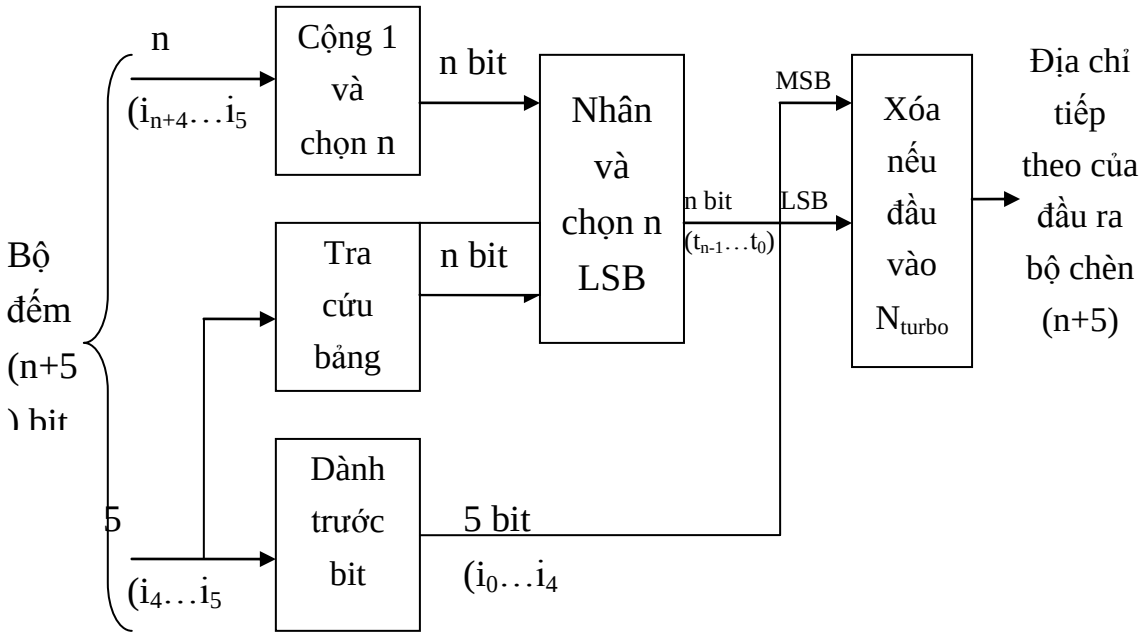
Lưu ý : Đối với từng tỷ lệ mã bảng trích bỏ sẽ được đọc từ trên xuống dưới sau đó từ trái sang phải

Đầu ra	Tỷ lệ mã		
	1/2	1/3	1/4
X	111000	111000	111000
Y_0	111000	111000	111000
Y_1	000000	000000	111000
X’	000111	000111	000111
Y'_0	000111	000111	000111
Y'_1	000000	000000	000111

Bảng 4.2. Mẫu trích bỏ cho các chu kỳ bit đuôi

Lưu ý: Đối với mã Turbo 1/2, bảng trích bỏ được đọc từ trên xuống dưới sau đó từ trái sang phải. Đối với các mã Turbo 1/3 và 1/4 bảng trích bỏ được đọc từ trên xuống dưới đồng thời với lặp X và X' sau đó đọc từ trái sang phải

Giả sử trình tự của các địa chỉ vào là từ 0 đến N_{turbo}-1, trong đó N_{turbo} là số các ký hiệu ở bộ chèn sẽ được xác định theo thủ tục được cho ở hình 5.2 như sau:



Hình 4.2 Thủ tục tinh toán địa chỉ đầu ra bộ chèn xen turbo

1. Xác định thông số bộ chèn: n, trong đó n là số nguyên nhỏ nhất để $N_{turbo} \leq 2^{n+5}$. Bảng 5.3 cho các thông số này.
2. Khởi đầu bộ đếm (n+5) bit vào "0".
3. Lấy ra n bit trọng số cao nhất (MSB) từ bộ đếm và cộng 1 để được giá trị mới. Sau đó xóa tất cả trừ n bit trọng số thấp nhất (LSB) của giá trị này.
4. Tra cứu bảng 3.4 theo địa chỉ đọc bằn năm bit trọng số thấp nhất (LSB) của bộ đếm. lưu ý rằng bảng này phụ thuộc vào giá trị n.
5. Nhân các giá trị nhận được ở bước 3 và 4 rồi xóa tất cả trừ n bit trọng số thấp nhất (LSB).

6. Đảo vị trí cho năm bit trọng số thấp nhất (LSB) của bộ đếm.
7. Tạo địa chỉ ra thử với các bít trọng số cao (MSBs) nhận được ở bước 6 và các bit trọng số thấp (LSB) nhận được ở bước 5.
8. Tiếp nhận địa chỉ ra thử này nếu nó không lớn $N_{turbo,}$ ngược lại xóa bỏ.
9. Tăng bộ đếm và lặp lại các bước từ 3 đến 8 cho đến khi nhận được tất cả N_{turbo} địa chỉ ra cho bộ chèn.

Kích thước khối của bộ chèn Turbo Turbo (N_{turbo})	Thông số của bộ chèn Turbo n
378	4
570	5
762	5
1.146	6
1.530	6
2.298	7
3.066	7
4.602	8
6.138	8
9.210	9
1.282	9
20.730	10

Bảng 4.3. Thông số của bộ chèn turbo

Bảng chỉ số	n=4	n=5	n=6	n=7	n=8	n=9	n=10
0	5	27	3	15	3	13	1
1	15	3	27	127	1	335	349
2	5	1	15	89	5	87	303
3	15	15	13	1	83	15	721

4	1	13	29	31	19	15	973
5	9	17	5	15	179	1	703
6	9	23	1	61	19	333	761
7	15	13	31	47	99	11	327
8	13	9	3	127	23	13	453
9	15	3	9	17	1	1	95
10	7	15	15	119	3	121	241
11	11	3	31	15	13	155	187
12	15	13	17	57	13	1	497
13	3	1	5	123	3	175	909
14	15	13	39	95	17	421	769
15	5	29	1	5	1	5	349
16	13	21	19	85	63	509	71
17	15	19	27	17	131	215	557
18	9	1	15	55	17	47	197
19	3	3	13	57	131	425	499
20	1	29	45	15	211	295	409
21	3	17	5	41	173	229	259
22	15	25	33	93	231	427	335
23	1	29	15	87	171	83	253
24	13	9	13	63	23	409	677
25	1	13	9	15	147	387	717
26	9	23	15	13	243	193	313
27	15	13	31	15	213	57	757
28	11	13	17	81	189	501	189
29	3	1	5	57	51	313	15
30	15	13	15	31	15	489	75
31	5	13	33	69	67	391	163

Bảng 4.4. Quy định bảng tra cứu cho bộ chèn Turbo

4.4.4. Phối hợp tốc độ trong hệ thống CDMA 2000:

Phối hợp tốc độ có nghĩa là lặp hoặc trích bỏ các ký hiệu ở kênh truyền tải (viết tắt là TrCH) để đạt được tốc độ ký hiệu như nhau cho các kênh có tốc độ bít khác nhau ở các kênh vô tuyến khác nhau. Lớp cao sẽ ấn định thuộc tính của phối hợp tốc độ cho từng TrCH. Thuộc tính này là bán cố định và chỉ có thể thay đổi theo thông báo của lớp cao. Thuộc tính phối hợp tốc độ được sử dụng để tính số bit cần lặp hoặc trích bỏ.

Trong hệ thống CDMA 2000, lặp ký hiệu và trích bỏ ký hiệu được thực hiện ở sau bộ mã hóa kênh. Mẫu lặp và trích bỏ phụ thuộc vào cấu hình vô tuyến (RC: Radio configuration). Dưới đây ta xét một số thí dụ về lặp và trích bỏ ký hiệu ở CDMA 2000

Lặp ký hiệu mã

Các ký hiệu mã ở đầu ra của bộ mã hóa hiệu chỉnh lỗi thuận sẽ được lặp theo quy định ở bảng 4.5

Kiểu kênh		Số ký hiệu mã sau lặp/ ký hiệu mã
Kênh truy nhập (chỉ cho tốc độ trải phổ một)		2
Kênh truy nhập cải tiến		4 (9600 bit/s) 2 (19200 bit/s) 1 (38400 bit/s)
Kênh điều khiển chung đường lên		4 (9600 bit/s) 2 (19200 bit/s) 1 (38400 bit/s)
Kênh điều khiển riêng đường lên		2
Kênh cơ bản đường lên	RC 1 hay 2	8 (1200 hay 1800 bit/s) 4 (2400 hay 3600 bit/s) 2 (4800 hay 7200 bit/s) 1 (9600 hay 14400 bit/s)
	RC 3,4,5 hay 6	16 (1500 hay 1800 bit/s)

		8 (2700 hay 3600 bit/s) 4 (4800 hay 7200 bit/s) 2 (9600 hay 14400 bit/s)
Kênh mã bổ sung đường lên	(RC 1hay 2)	1
Kênh bổ sung đường l		1

Bảng 4.5 Quy định bảng tra cứu cho bộ đan xen turbo

4.4.5. Chèn trong CDMA 2000:

Chèn thường đi với mã hóa kênh để tăng hiệu quả của sửa lỗi. trong thông tin di động do pha đinh sâu, các bit thường xẩy ra từng cụm dài. Tuy nhiên mã hóa kênh đặc biệt là mã tích chập chỉ hiệu quả nhất khi sửa các lối ngẫu nhiên đơn lẻ hoặc các cụm lỗi không quá dài. Để đối phó với vấn đề này người ta chia khối bản tin cần gửi thành các cụm ngắn rồi hoán vị các cụm này với các cụm của các khối bản tin khác, nhờ vậy khi xẩy ra cụm lỗi dài mỗi khối bản tin chỉ mất đi một cụm nhỏ và phần còn lại của khối bản tin vẫn cho phép các dạng mã hóa kênh khôi phục được khối đúng sau khi đã sắp xếp lại các cụm của khối bản tin theo thứ tự như phía phát. Chẳng hạn ta có 4 khối bản tin hình 4.3 ta chia mỗi khối thành 4 cụm và đánh số cho các cụm này từ 1 đến 4, sau đó hoán vị các cụm với nhau bằng cách ghép chung các cụm 1 vào một khối và cụm vào một khối… giả sử ở phía thu các cụm 2 bị mắc lỗi, sau khi sắp xếp lại các khối bản tin các cụm 1, 3, 4 còn lại sẽ cho phép mã hóa kênh khôi phục lại khối đúng

4.4.5.1. Chèn khối:

Đối với kênh đồng bộ, các kênh tìm gọi, các kênh quảng bá, kênh ấn định chung, kênh điều khiển và các kênh lưu lượng đường xuống, tất cả các ký hiệu sau lặp và trích bỏ (nếu có) sẽ được chèn khối.

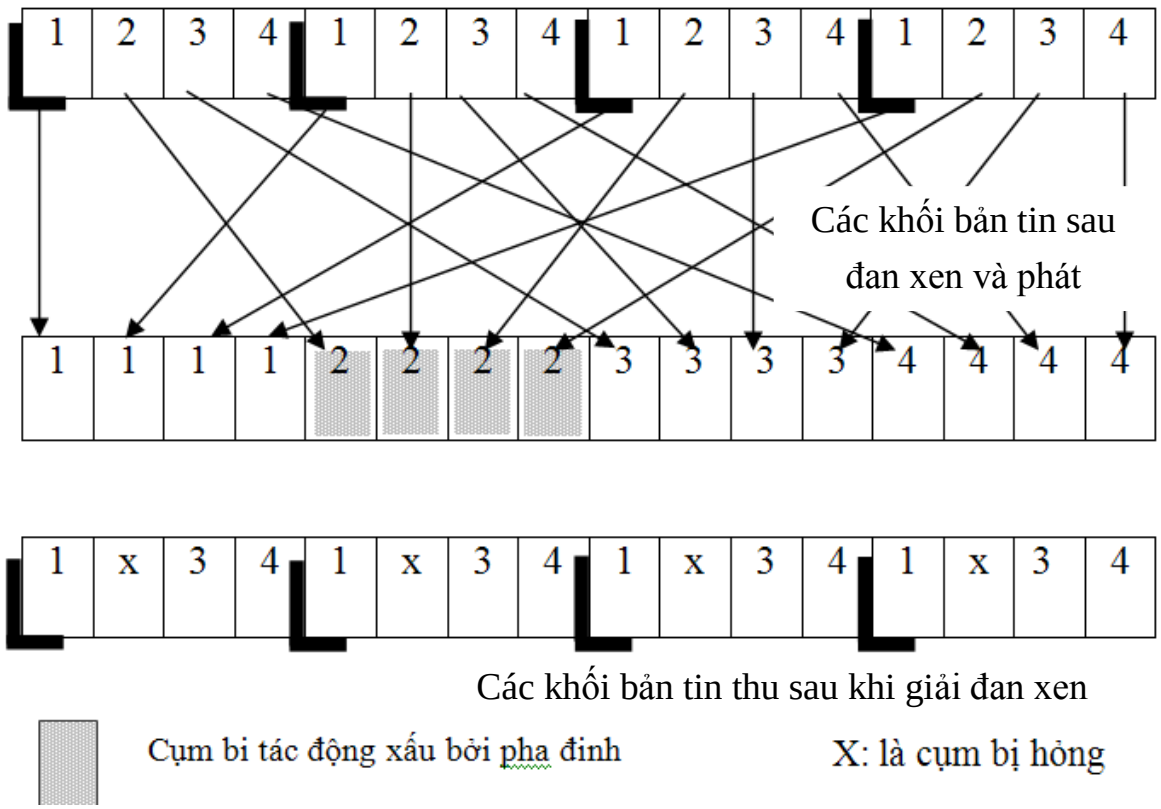
Các ký hiệu đầu vào lần lượt được viết vào bộ đan xen theo địa chỉ từ 0 đến kích thước khối (N) trừ một. Các ký hiệu sau đan xen được đọc ra theo trính tự hoán vị từ địa chỉ A_i như sau:

$$A_i = 2^m(i \bmod j) + BRO_m([i/j]) \quad (4.2)$$

Trong đó :

i = 0 đến N-1

[x] biểu thị số nguyên lớn nhất nhỏ hơn x và $BRO_m(y)$ biểu thị giá trị m bit đảo của y (chẳng hạn BRO3(6) = 3).



Hình 4.3. Nguyên lý đan xen

Các thông số của bộ đan xen: m và j được quy định ở bảng 4.6 hình 4.4 cho thấy cấu hình của bộ đan xen đối với các chế độ DS non – OTD (direct spreading – non Orthogonal Transmit diversity: đơn sóng mang không sử dụng phân tập phát trực giao), DS OTD (đơn sóng mang với phân tập phát trực giao) và MC (Multicarrier: đa sóng mang).

4.4.5.2. Chèn đa khung:

Khi số bit của kênh bổ sung đường xuống là 360 hay lớn hơn, BS phải cho phép đan xen kênh bổ sung ở hai hoặc khung liên tiếp theo quy định được xác định bởi MULTI_FRAME_LENGTH.

Cấu trúc của bộ chèn khối n khung (n=2 hay 4) giống như cấu trúc của bộ chèn đơn khung. Thông số đan xen cho bộ chèn n khung được quy định ở bảng 4.7

Kích thước đan xen	m	j	Kích thước đan xen	m	j
48	4	3	144	4	9
96	5	3	288	5	9
192	6	3	576	5	18
384	6	6	1.152	6	18
768	6	12	2.304	6	36
1.536	6	24	4.608	7	36
3.072	6	48	9.216	7	72
6.144	7	48	18.432	8	72
12.288	7	96	36.864	8	144
72	3	9	128	7	1

Bảng 4.6 Các thông số chèn

TTI	Số cột C_1	Các mẫu hoán vị giữa các cột
10 ms	1	{0}
20 ms	2	{0,1}
40 ms	4	{0,2,1,3}
80 ms	8	{0,4,2,6,1,5,3,7}

Bảng 4.7. Các mẫu hoán vị giữa các cột

4.4.5.3. Chèn OTD:

Khi sử dụng chế độ OTD, bộ đan xen khối sẽ phân luồng các ký hiệu vào thành hai luồng. Mỗi luồng được đan xen theo thủ tục được trình bày ở trên. Khối thứ hai được dịch vòng N/4 ký hiệu sau đó hai khối được ghép

chung. Dịch vòng được thực hiện bằng cách dịch 3N/4 ký hiệu đến cuối khối này và N/4 ký hiệu đến đầu khối này. Thủ tục dịch vòng được cho ở hình 4.4

4.4.5.4 Chèn MC:

Đối với chế độ MC, bộ đan xen khối sẽ phân luồng các ký hiệu đầu vào thành ba khối N/3 ký hiệu.

Các ký hiệu vào k bộ đan xen khối (k = 0,1,2) lần lượt được ghi vào bộ nhớ ở các địa chỉ từ 0 đến N/3 – 1. các ký hiệu sau đan xen được đọc ra theo trình tự hoán vị từ địa chỉ A_i như sau:

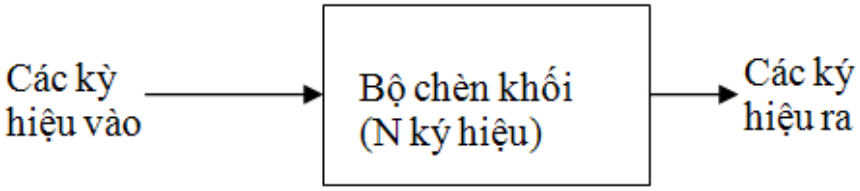
$A_i = 2^m((i+[kN/9] \bmod J) + BRO_m([(i+[kN/9])/J])$ (4.3)

Trong đó:

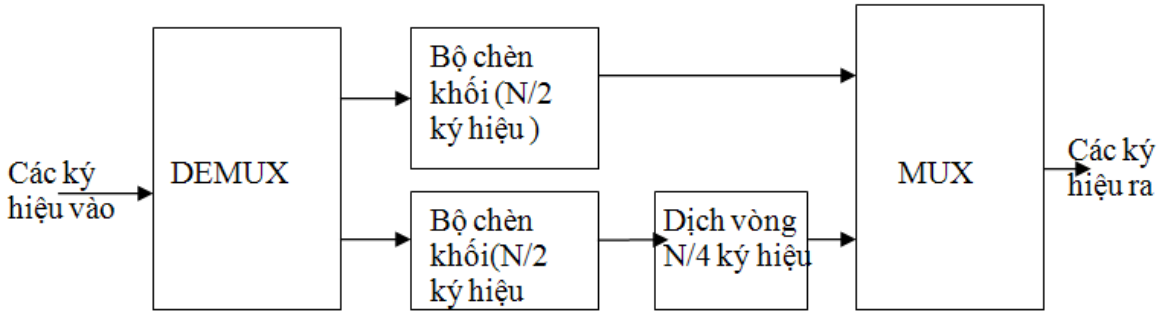
i = 0 đến N/3-1

[x] chỉ thị số nguyên lớn nhất nhỏ hơn hay bằng x và $BRO_m(y)$ chỉ thị giá trị m bị đảo vị trí (chẳng hạn $BRO_3(6)=3$).

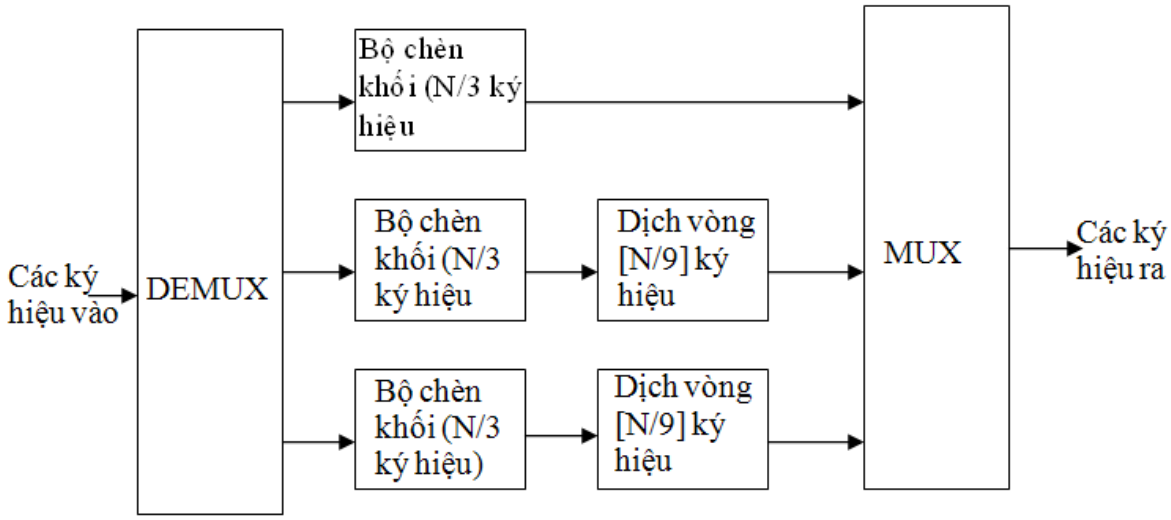
Sau đó ba đầu ra của các khối đan xen ghép chung với nhau. Quá trình đan xen khối MC được thực hiện đồng thời cho cả đan xen khối và dịch vòng khối như ở hình 4.4



a) Chế độ DS Non – OTD



b) Chế độ DS OTD



c) Các chế độ MC

chức năng của DEMUX là phân phối lần lượt các ký hiệu vào cho đường trên và đường dưới.

chức năng của MUX là kết hợp lần lượt các ký hiệu vào từ đường trên và đường xuống dưới.

Hình 4.4. cấu trúc các bộ đan xen khối N ký hiệu

4.5 KẾT LUẬN:

Qua chương này ta biết được những ứng dụng của mã Turbo vào trong truyền thông đa phương tiện và truyền thông không dây, cụ thể là ứng dụng trong CDMA 2000. Những khó khăn khi thực hiện để tìm ra những cách khắc phục những nhược điểm đó nhằm mang lại những tiện ích hơn nữa của công nghệ CDMA đối với cuộc sống con người. Bộ mã sử dụng trong CDMA 2000 sẽ được mô phỏng trong chương tiếp theo giúp ta hiểu sâu hơn về nó.

Chương 5

CH- ƠNG TRÌNH MÔ PHỎNG MÃ TURBO TRONG HỆ THỐNG THÔNG TIN DI ĐỘNG CDMA 2000

5.1 GIỚI THIỆU CH- ƠNG

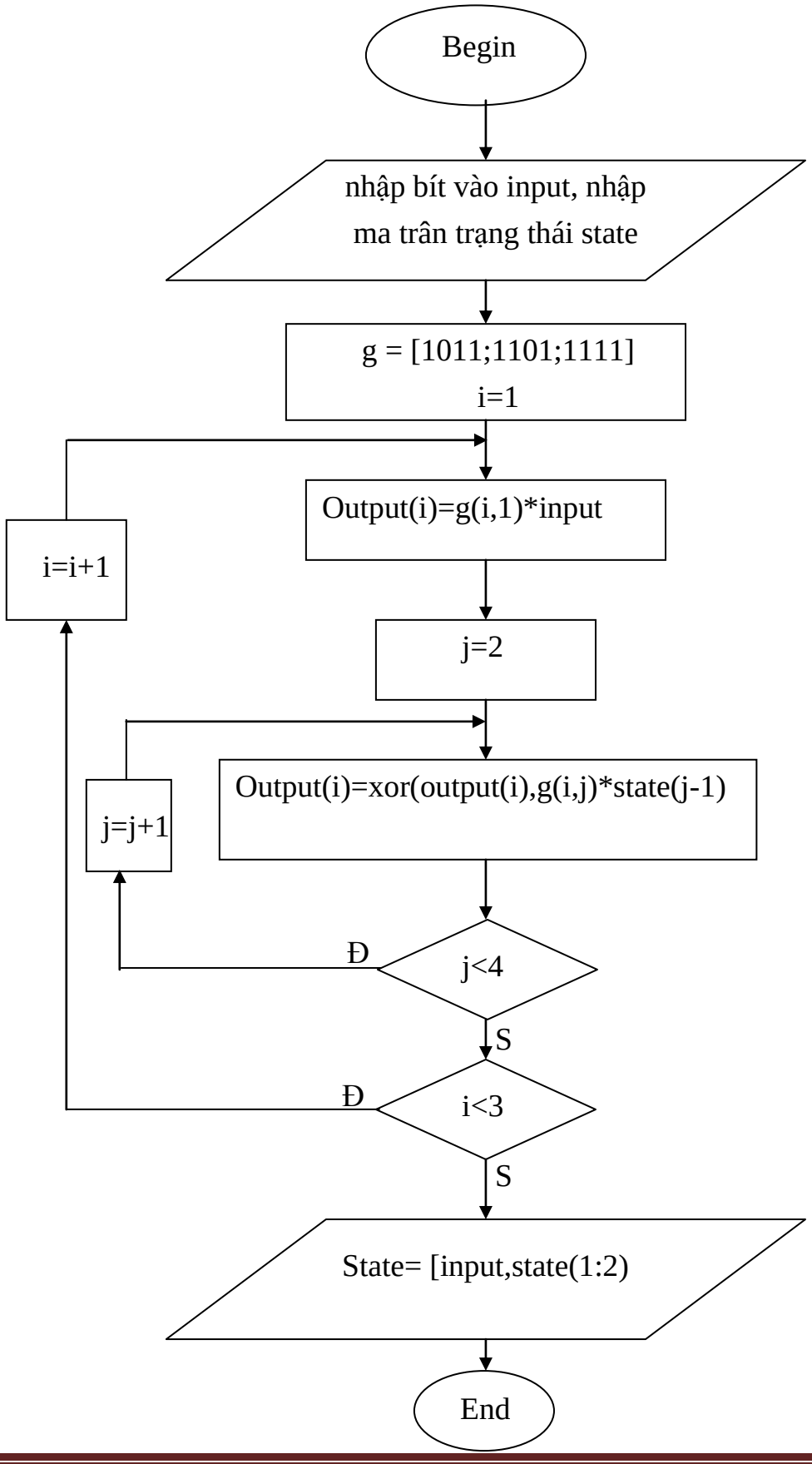
Trong chương này trình bày chương trình mô phỏng bộ mã turbo sử dụng trong hệ thống thông tin di động thế hệ 3 theo chuẩn CDMA2000. Chương trình được viết bằng ngôn ngữ Matlab, thông qua chương trình mô phỏng giúp ta kiểm tra lại lý thuyết và hiểu sâu hơn về mã turbo,cũng như khả năng ứng dụng của mã turbo khi tốc độ bít cao Qua đó cho chúng ta đánh giá được những đặc điểm như khả năng sửa lỗi...mà các loại mã hóa kênh khác không có. Trong chương trình mô phỏng ta nhập các bit số liệu vào khác nhau, số lần lặp giải mã khác nhau, cũng như số bit khung để thu được kết quả giải mã, BER khác nhau. Bộ mã này có hàm truyền như sau:

$$G(D)=\left[1 \quad \frac{n_0(D)}{d(D)} \quad \frac{n_1(D)}{d(D)}\right]$$

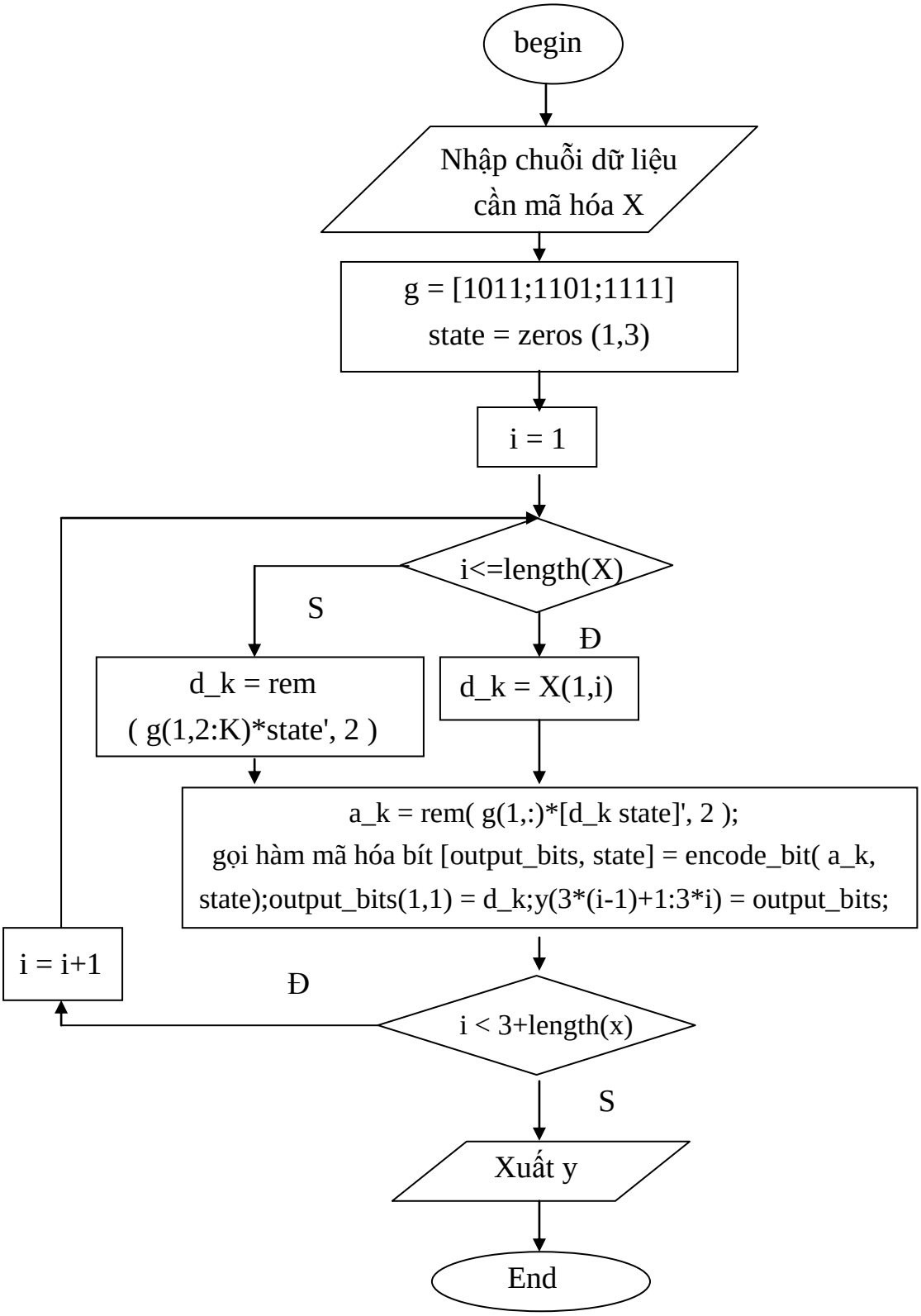
Trong đó $d(D) = 1+D^2+D^3$, $n_0(D) = 1+D+D^3$ và $n_1(D) = 1+D+D^2+D^3$

5.2 L- U ĐỒ THUẬT TOÁN

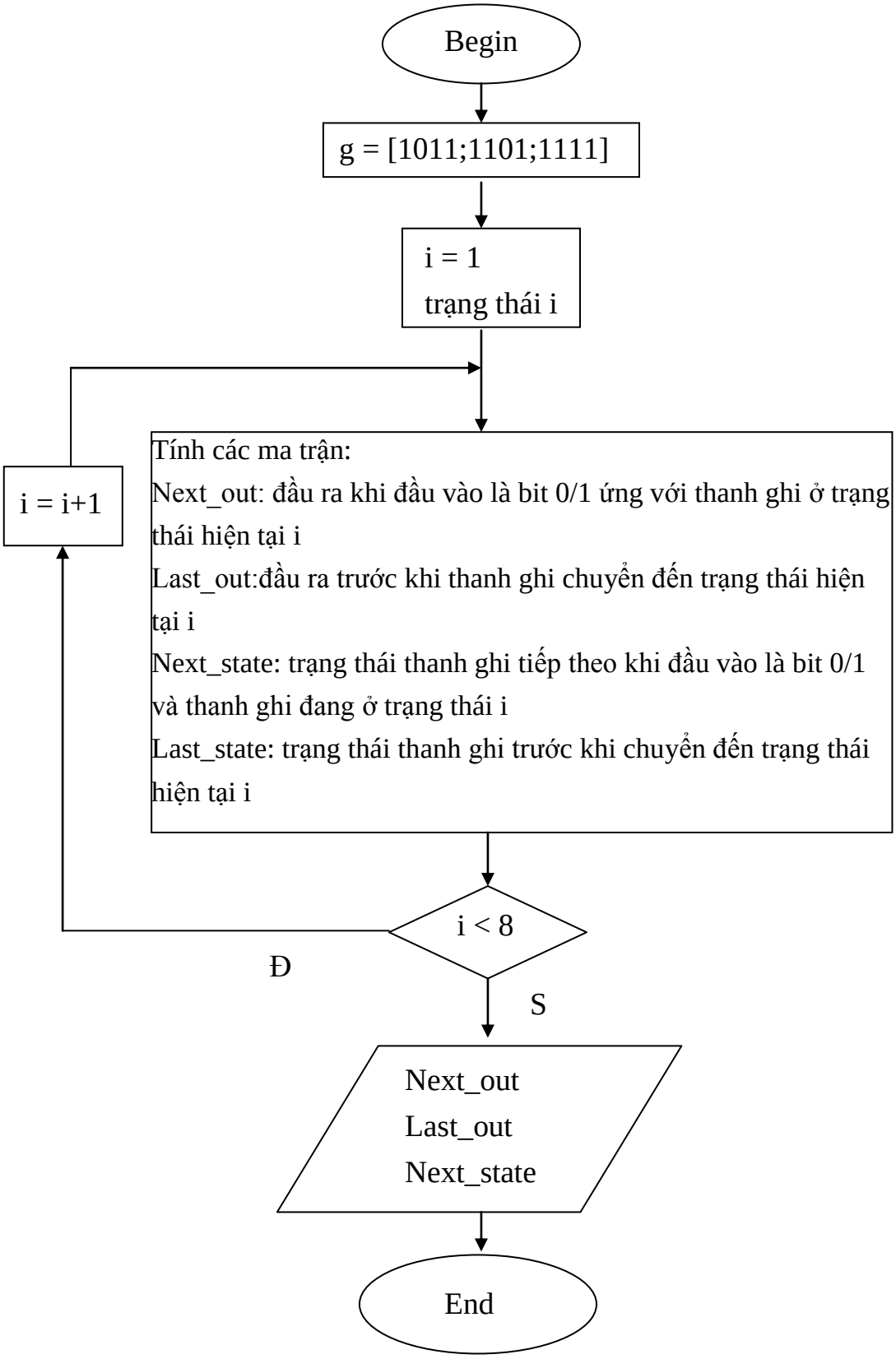
5.2.1 L- u đồ thuật toán ch- ơng trình mã hóa theo bit



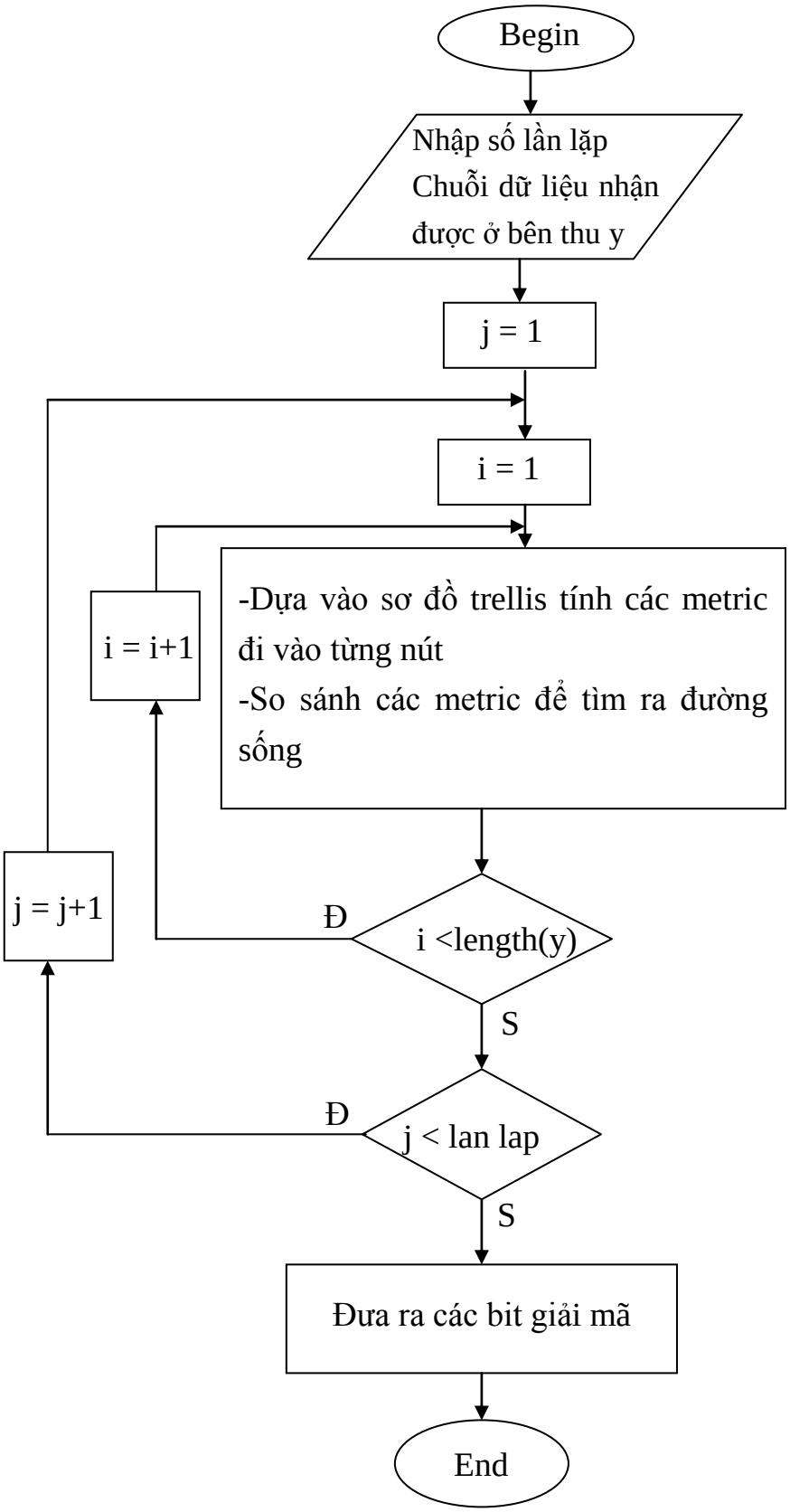
5.2.2 L- u đồ thuật toán mã hóa chuỗi dữu liệu đầu vào



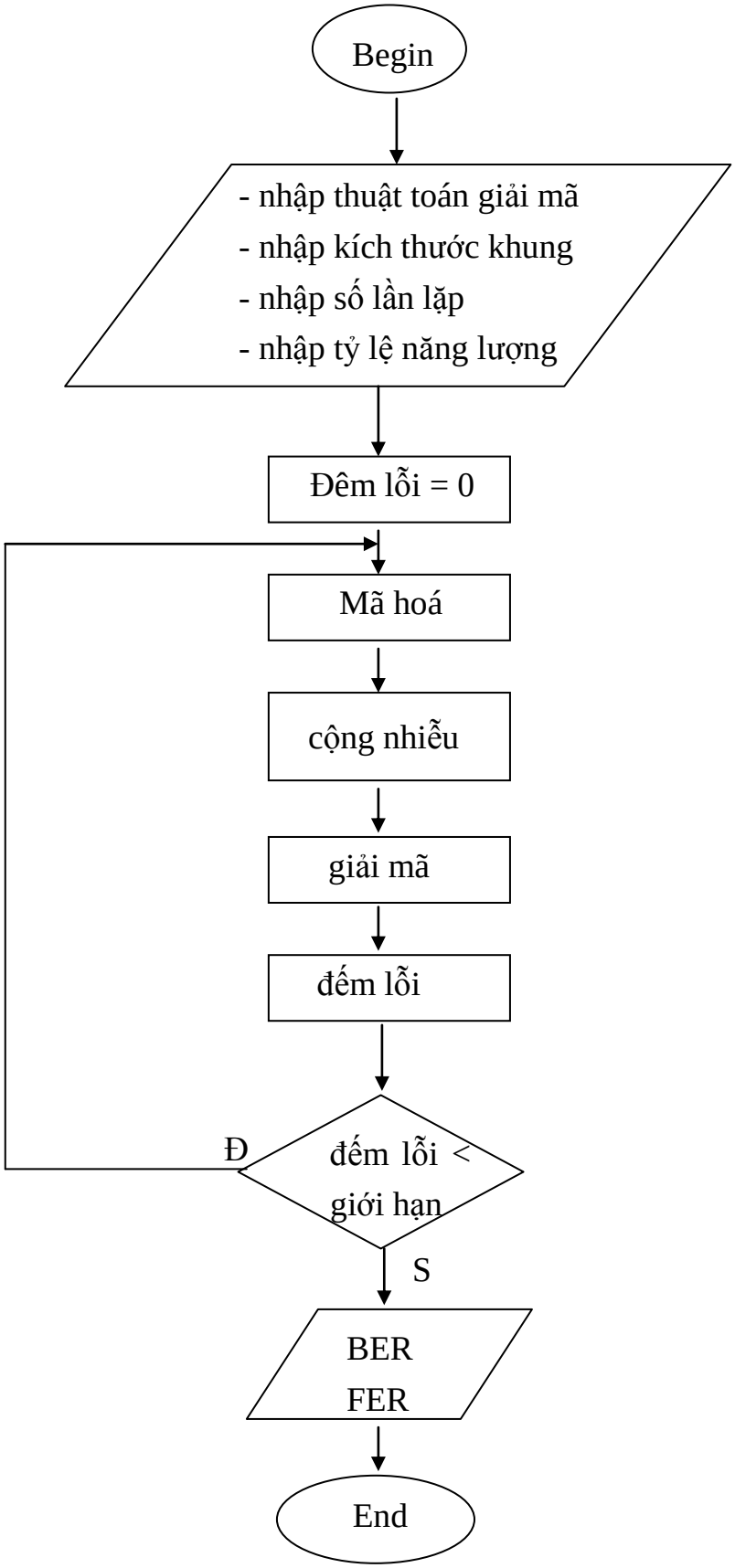
5.2.3. Lưu đồ thuật toán tính các ma trận của trạng thái Trellis:



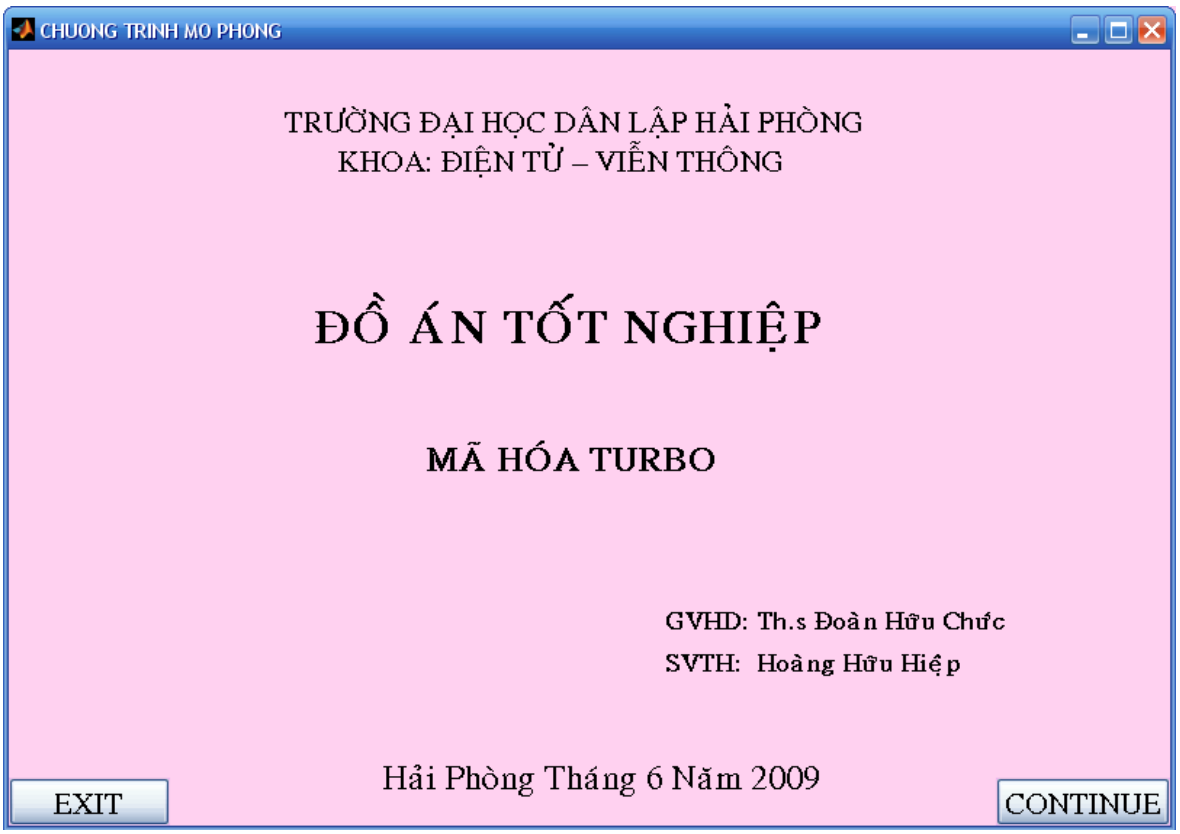
5.2.4 L- u đồ thuật toán giải mã Turbo



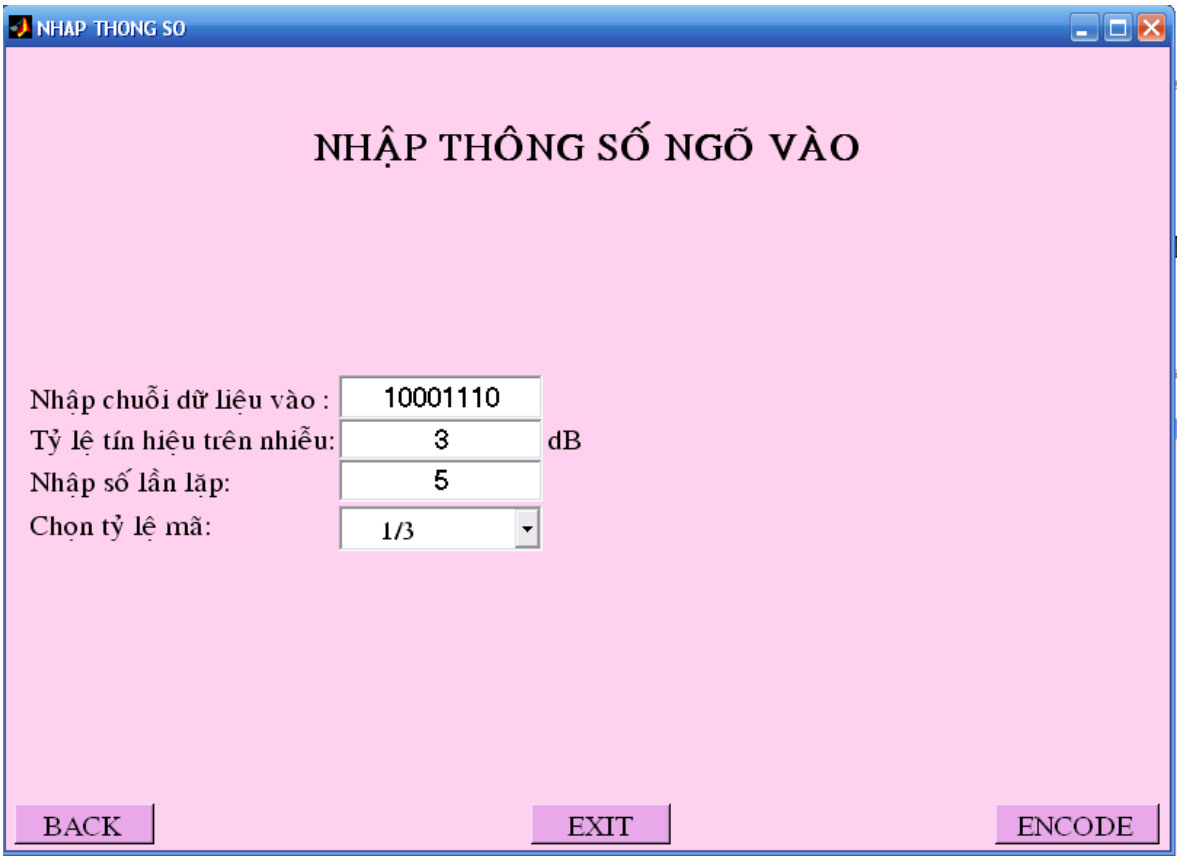
5.2.5. Lưu đồ thuật toán tính lỗi bit và lỗi khung:



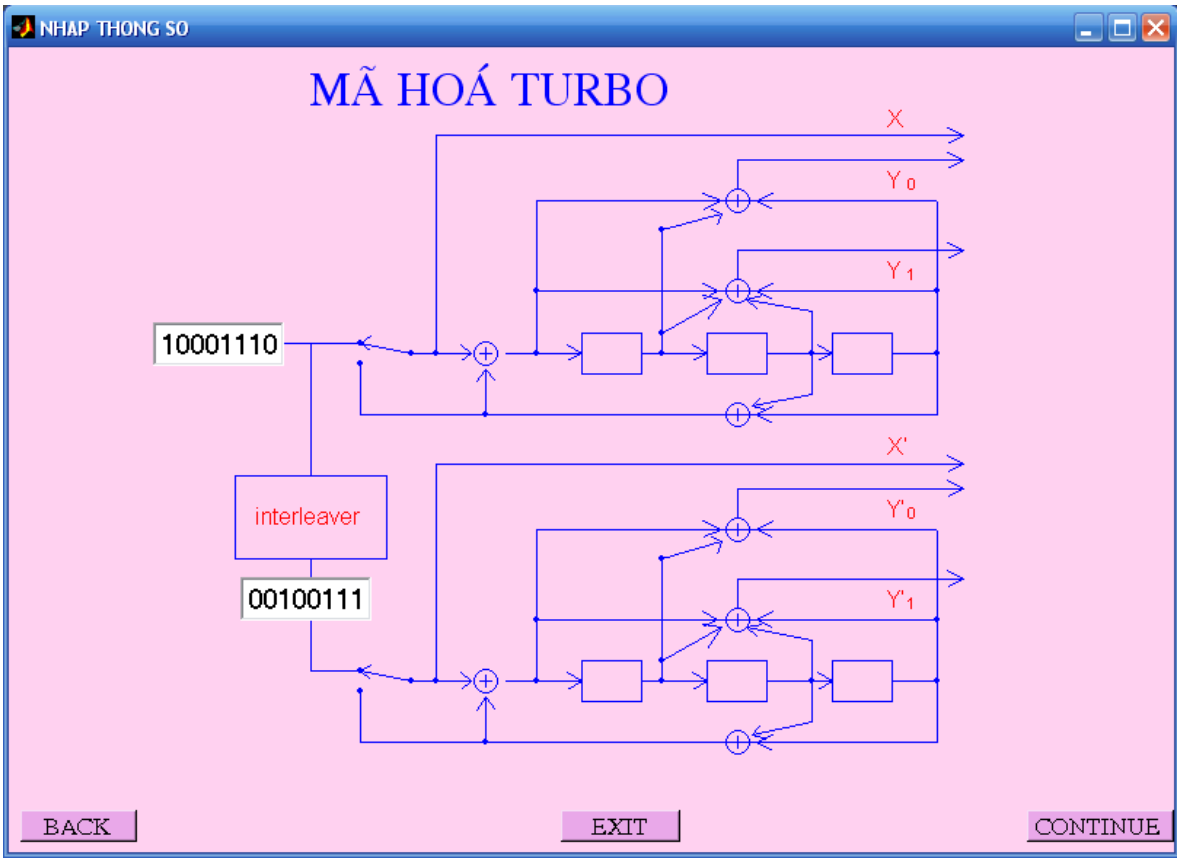
5.3. Giao diện và kết quả chương trình mô phỏng:



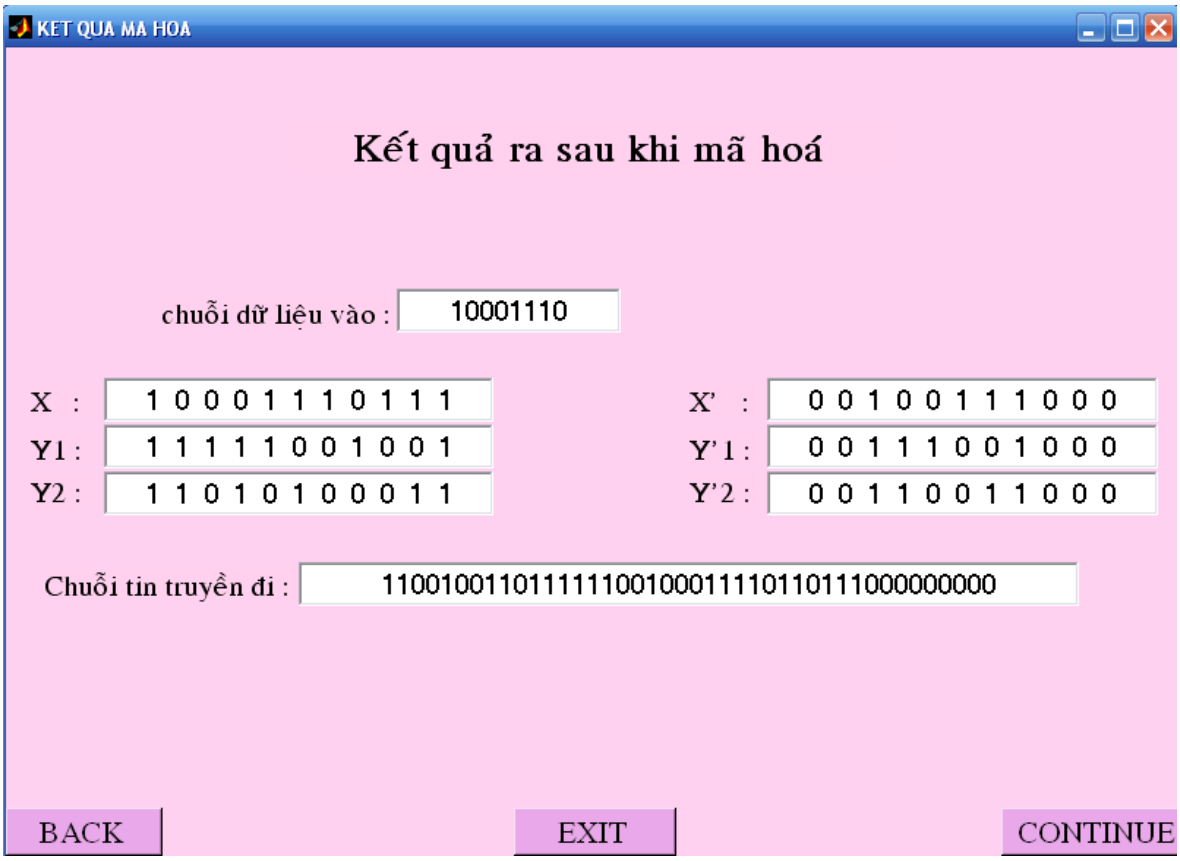
Khi chọn "Exit " chương trình sẽ thoát còn chọn "continue" chương trình sẽ tiếp tục cho ra trang nhập thông số vào



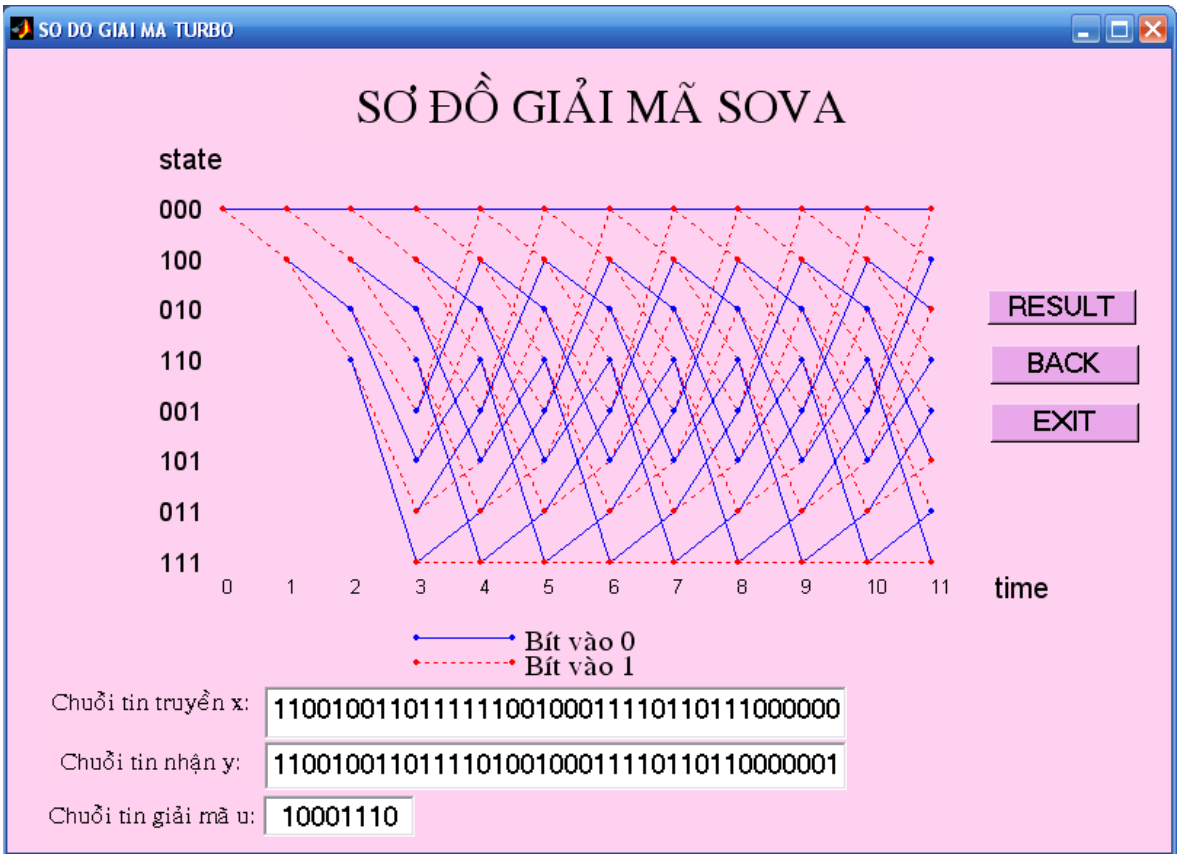
Ta nhập chuỗi dữ liệu vào, tỷ lệ tín hiệu trên nhiễu, số lần lặp giải mã, tỷ lệ mã truyền đi có 3 tỷ lệ là 1/2, 1/3, 1/4. chọn "ENCODE" để tiếp tục tới trang mã hoá



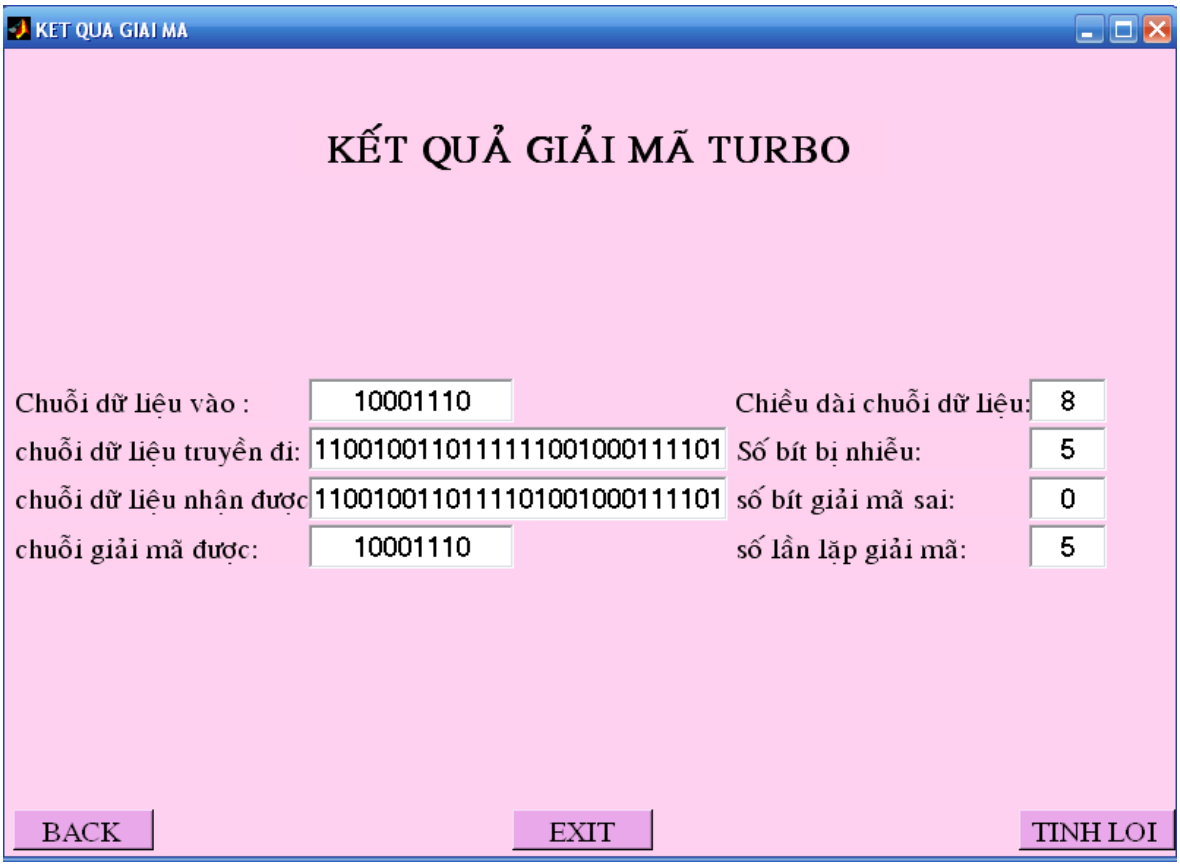
xuất hiện bộ mã hoã Turbo CDMA2000 chuỗi dữ liệu đưa vào sau khi qua bộ chèn hoán vị ngẫu nhiên cho ra chuỗi mới để đưa vào bộ mã hoá thành phần thứ hai. Ta chọn "CONTINUE" để đưa ra kết quả mã hoá.



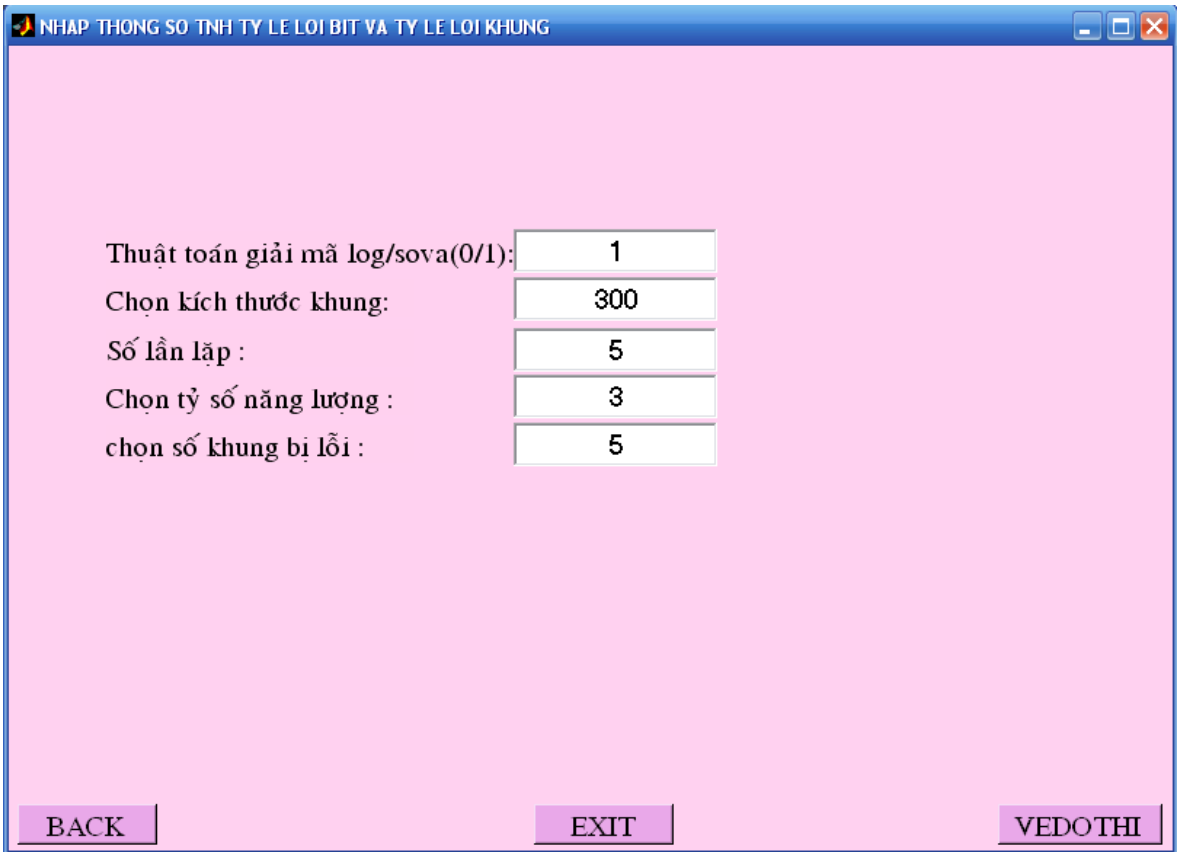
Ta thu được kết quả mã hoá như trên hình và đưa ra chuỗi tin cần truyền đi phụ thuộc vào việc chọn tỷ lệ mã trước. chọn "BACK" để quay về trang trước, chọn "EXIT" để thoát, chọn "CONTINUE" để tiếp tục đến trang sau.



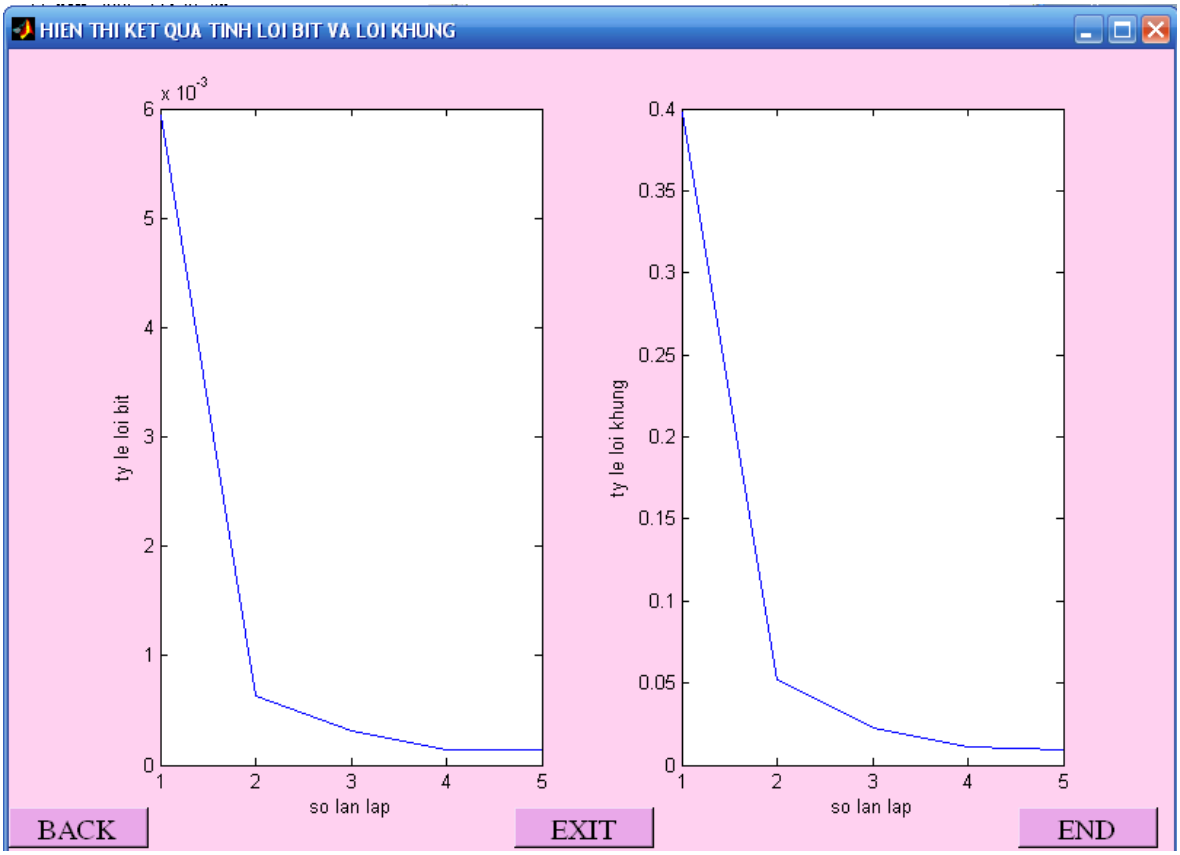
Xuất hiện sơ đồ lưới dùng để giải mã, tiến hành giải mã chuỗi tin nhận được y chuỗi này có một số bít lỗi khác với chuỗi truyền. ta tiếp tục chọn "RESULT" để đưa ra kết quả giải mã.



ta tiếp tuc chạy chương trinh tính tỷ lệ lỗi bit và lỗi khung khi nhập các thông số đầu vào khác nhau thì ta thu được những gia trị lỗi bít và lỗi khung khác nhau. Và có đồ thị khác nhau



Sau khi tính lỗi xong nó sẽ đưa kết quả đến để vẽ đồ thị



Ta chạy chương trình mô phỏng nhiều lần ta đưa ra một số nhận xét như sau:

+ Khi số lần lặp tăng từ thì tỉ lệ lỗi bit cũng như tỉ lệ lỗi khung đều giảm. việc thực hiện mã Turbo được cải tiến nhiều, điều này là do sau khi thông tin được chia sẽ giữa các bộ giải mã có nhiều thông tin về ngõ vào và vì vậy đưa ra quyết định chính xác hơn Khi số lần lặp tăng lớn hơn 2 thì việc thực hiện của mã Turbo cũng được cải tiến.Tuy nhiên,mức độ cải tiến không được cao, điều này là do sau lần lặp,các bộ giải mã đã lấy được hết thông tin của mã ngõ vào và do dó : không cho ra ở ngõ ra các giá trị biến đổi nữa như trong lần lặp thứ nhất.Vì vậy, có thể nói việc thực hiện của mã Turbo sẽ đạt đến mức ngưỡng sau vài lần lặp Nếu số lần lặp tăng hơn mức ngưỡng thì việc thực hiện mã Turbo sẽ bị giảm xuống, sau mức ngưỡng thì các lần lặp sau không đem đến thông tin khác hơn đến các bộ giải mã

Như vậy,việc thực hiện mã Turbo tăng khi số lần lặp tăng và thời gian sử dụng giải mã cũng tăng tuyến tính theo số lần lặp.Vì vậy,người thiết kế phải điều chỉnh số lần lặp sao cho phù hợp giữa việc thực hiện của mã và thời gian giải mã.

Tuy nhiên,trong quá trình giải mã,thuật toán SOVA phải chịu 2 loại méo Méo thứ nhất là các ngõ ra mềm vượt quá tối ưu thường được bù bằng hệ số chia mức Méo thứ hai là sự tương quan giữa thông tin bên ngoài và bên trong hay sự tương quan giữa ngõ ra mềm của mỗi bộ giải mã tương ứng với các bit kiểm tra chẵn lẽ của nó và chuỗi dữ liệu ngõ vào thông tin

+ Nếu số lượng khung đưa vào càng lớn thì BER và FER càng thấp

+ Mã sẽ hoạt động tốt khi ta lựa chọn kích thước khung lớn.

+ Tỉ lệ lỗi khung(FER) thường lớn hơn tỉ lệ lỗi bit(BER) nhưng lần lặp càng lớn thì BER~FER

PHỤ LỤC MÔ PHỎNG BẰNG MATLAP

```
++++++++++++++++++++++++++++++++++++++++++++++
FiLe main.m
++++++++++++++++++++++++++++++++++++++++++++++
function main
h0 = figure('Units','points',...
        'Color',[1 0.819607843137255 0.941176470588235],...
        'MenuBar','none','Name','CHUONG TRINH MO PHONG',...
        'NumberTitle','off',...
        'PaperPosition',[18 180 576 432],...
        'PaperUnits','points',...
        'Position',[0 25 600 400.5],...
        'Tag','Fig1','ToolBar','none');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'Callback','close;',...
        'FontName','vni-times', 'FontSize',16,...
        'ListboxTop',0,...
        'Position',[0 0 83.25 24.75],...
        'String','EXIT', 'Tag','Pushbutton1');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'FontName','vni-times',...
        'FontSize',16,'ListboxTop',0,...
        'callback','close all;input1',...
        'Position',[510 0 90 25],...
        'String','CONTINUE','Tag','Pushbutton2');
```

```
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
        'FontName','vni-times', 'FontSize',16,...
        'ListboxTop',0,...
        'Position',[96.75 363 392.25 31.5],...
        'String', ',...
        'Style','text', 'Tag','StaticText1');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
        'FontName','VNI-TIMES','FontSize',16,...
        'ListboxTop',0,...
        'Position',[95.25 342.75 392.25 31.5],...
        'String',TRUONG DAI HOC DAN LAP HAI PHONG',...
        'Style','text','Tag','StaticText2');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
        'FontName','VNI-TIMES','FontSize',16,...
        'ListboxTop',0,...
        'Position',[88.5 322.5 392.25 31.5],...
        'String',KHOA DIEN TU-VIEN THONG',...
        'Style','text', 'Tag','StaticText3');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
        'FontName','VNI-TIMES','FontSize',25,...
        'FontWeight','bold','ListboxTop',0,...
        'Position',[92.25 243.75 392.25 31.5],...
        'String',DO AN TOT NGHIEP,...
        'Style','text','Tag','StaticText4');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
```

```
        'FontName','VNI-TIMES','FontSize',18,...
        'FontWeight','bold','ListboxTop',0,...
        'Position',[92.25 145.5 395.25 56.25],...
        'String', MA HOA TURBO ',...
        'Style','text','Tag','StaticText5');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
        'FontName','vni-times','FontSize',13,...
        'FontWeight','bold','ListboxTop',0,...
        'Position',[339 95.25 222.75 19.5],...
        'horizontalalignment','left',...
        'String','GVHD: Th.S DOAN HUU CHUC',...
        'Style','text','Tag','StaticText6');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
        'FontName','vni-times','FontSize',13,...
        'FontWeight','bold','ListboxTop',0,...
        'Position',[339 73.5 222.75 19.5],...
        'horizontalalignment','left',...
        'String','SVTH: HOANG HUU HIEP,...
        'Style','text','Tag','StaticText7');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
        'FontName','VNI-TIMES','FontSize',18,...
        'ListboxTop',0,...
        'Position',[110.25 5.75 392.25 31.5],...
        'String',Hải Phòng Tháng 07 Năm 2009',...
        'Style','text','Tag','StaticText4');
if nargout > 0, fig = h0;
end;
```

```
++++++++++++++++++++++++++++++++++++++++++++++++++++
File input1.m
++++++++++++++++++++++++++++++++++++++++++++++++++++
clear;
h0 = figure('Color',[1 0.819607843137255 0.941176470588235],...
        'NumberTitle','off',...
        'MenuBar','none','Name','NHAP THONG SO',...
        'PaperPosition',[18 30 576 432],...
        'PaperUnits','points','Position',[1 29 800 553],...
        'Tag','Fig1','ToolBar','none');

h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
        'FontName','vni-times','FontSize',20,...
        'FontWeight','bold','ListboxTop',0,...
        'Position',[147 350.25 305.25 27.75],...
        'String','NHẬP THÔNG SỐ NGÕ VÀO',...
        'Style','text','Tag','StaticTextdau');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[12.75 225.25 158.25 18.75],...
        'String','Nhập chuỗi dữ liệu vào :',...
        'Style','text','Tag','StaticText1');
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[171.75 222.75 105 22.5],...
```

```
        'String','',...
        'Style','edit','Tag','EditText1dlv',...
        'callback', 'dauvao');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
        'FontName','vni-times',...
        'FontSize',15,...
        'HorizontalAlignment','left',...
        'ListboxTop',0,...
        'Position',[12.75 204.10 170.5 18.75],...
        'String','Tỷ lệ tín hiệu trên nhiễu:',...
        'Style','text','Tag','StaticText2');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
        'FontName','vni-times',...
        'FontSize',15,...
        'HorizontalAlignment','left',...
        'ListboxTop',0,...
        'Position',[280 204.10 20.5 18.75],...
        'String','dB',...
        'Style','text','Tag','StaticText2');
hra = uicontrol('Parent',h0,...
         'Units','points',...
        'BackgroundColor',[1 1 1],...
        'FontSize',13,...
        'ListboxTop',0,...
        'Position',[171.75 201.75 105 21],...
        'String','',...
```

```
        'Style','edit','Tag','EditText13nangluong',...
        'callback','tyso_EbNo');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
        'FontName','vni-times',...
        'FontSize',15,...
        'HorizontalAlignment','left',...
        'ListboxTop',0,...
        'Position',[12.75 182.25 112.5 18.75],...
        'String','Nhập số lần lặp:',...
        'Style','text','Tag','StaticText3');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[1 1 1],...
        'FontSize',13,...
        'ListboxTop',0,...
        'Position',[171.75 180 105 21.75],...
        'String','',...
        'Style','edit','Tag','EditText3solan',...
        'callback','solan_lap');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'FontName','vni-times',...
        'FontSize',15,'HorizontalAlignment','left',...
        'ListboxTop',0,...
        'Position',[12.75 160 112.5 18.75],...
        'String','Chọn tỷ lệ mã:',...
        'Style','text','Tag','StaticText5');
```

```
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontName','vni-times',...
        'FontSize',14,'ListboxTop',0,...
        'Position',[171.75 161.25 105 16.5],...
        'String',' 1/2 | 1/3 | 1/4',...
        'Style','popupmenu',...
        'Tag','PopupMenu1tylema','Value',1,...
        'callback','tyle_ma');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'Callback','close',...
        'FontName','vni-times',...
        'FontSize',15,...
        'ListboxTop',0,...
        'Position',[270.75 3.25 72 21],...
        'String','EXIT','Tag','Pushbutton1');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'FontName','VNI-TIMES',...
        'FontSize',15,'ListboxTop',0,...
        'Position',[510.25 3.25 84.75 21],...
        'callback','close all ; mahoa2',...
        'String','ENCODE','Tag','Pushbutton2');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
```

```
        'Callback','close all ; main',...
        'FontName','vni-times',...
        'FontSize',15,...
        'ListboxTop',0,...
        'Position',[4.75 3.25 72 21],...
        'String','BACK','Tag','Pushbutton3');
if nargout > 0, fig = h0; end
++++++++++++++++++++++++++++++++++++++++++++++++++
File mahoa2.m
++++++++++++++++++++++++++++++++++++++++++++++++++
h0 = figure('Color',[1 0.819607843137255 0.941176470588235],...
        'NumberTitle','off',...
        'MenuBar','none','Name','NHAP THONG SO',...
        'PaperPosition',[18 30 576 432],...
        'PaperUnits','points','Position',[1 29 800 553],...
        'Tag','Fig1','ToolBar','none');
axis([0 18 0 16]);
axis off;
hold on;
grid;
text(3,16.5,'MÃ HOÁ TURBO','fontname','vni-times','fontsize',24,'color','b');
% bo ma hoa 1
p1=line([13.4 14.6 14.6 13.4 13.4],[9.5 9.5 10.5 10.5 9.5],'color','b');
p2=line([10.9 12.1 12.1 10.9 10.9],[9.5 9.5 10.5 10.5 9.5],'color','b');
p3=line([8.4 9.6 9.6 8.4 8.4],[9.5 9.5 10.5 10.5 9.5],'color','b');
plot(6.5,10,'ob','markersize',12);
plot(6.5,10,'+b');
plot(11.5,8.5,'ob','markersize',12);
plot(11.5,8.5,'+b');
p4=line([6.9 8.4],[10 10],'color','b');
```

```
p5=line([8.1 8.4 8.1],[10.2 10 9.8],'color','b');
p6=line([9.6 10.9],[10 10],'color','b');
p7=line([12.1 13.4],[10 10],'color','b');
p8=line([10.6 10.9 10.6],[10.2 10 9.8],'color','b');
p9=line([13.1 13.4 13.1],[10.2 10 9.8],'color','b');
plot(11.5,11.5,'ob','markersize',12);
plot(11.5,11.5,'+b');
plot(11.5,13.7,'ob','markersize',12);
plot(11.5,13.7,'+b');
p6=line([14.6 15.5],[10 10],'color','b');
p6=line([10 10],[10 13],'color','b');
p6=line([15.5 15.5],[8.5 13.7],'color','b');
p6=line([13 13],[9 11],'color','b');
p6=line([7.5 7.5],[10 13.7],'color','b');
p6=line([7.5 15.5],[13.7 13.7],'color','b');
p6=line([10.8 11.2 10.8],[13.9 13.7 13.5],'color','b');
p6=line([12.2 11.9 12.2],[13.9 13.7 13.5],'color','b');
p6=line([10 11.2],[13 13.4],'color','b');
p6=line([11 11.2 11.1],[13.5 13.4 13.1],'color','b');
p6=line([7.5 15.5],[11.5 11.5],'color','b');
p6=line([10.8 11.1 10.8],[11.7 11.5 11.3],'color','b');
p6=line([12.2 11.9 12.2],[11.7 11.5 11.3],'color','b');
p6=line([13 11.7],[11 11.3],'color','b');
p6=line([11.9 11.7 12],[11.1 11.3 11.4],'color','b');
p6=line([11.5 11.5],[14 14.7],'color','b');
p6=line([11.5 16],[14.7 14.7],'color','b');
p6=line([15.7 16 15.7],[14.9 14.7 14.5],'color','b');
p6=line([10 11.2],[10.5 11.3],'color','b');
p6=line([10.8 11.2 10.9],[11.4 11.3 10.8],'color','b');
p6=line([11.5 11.5],[11.8 12.5],'color','b');
```

```
p6=line([11.5 16],[12.5 12.5],'color','b');
p6=line([15.7 16 15.7],[12.7 12.5 12.3],'color','b');
p6=line([13 11.8],[9 8.7],'color','b');
p6=line([5 6.2],[10 10],'color','b');
p6=line([2 4],[10.25 10.25],'color','b');
p6=line([4 4],[9.75 8.5],'color','b');
p6=line([4 15.5],[8.5 8.5],'color','b');
p6=line([6 6.2 6],[10.2 10 9.8],'color','b');
p6=line([5.5 5.5],[10 15.3],'color','b');
p6=line([11.95 11.8 12.1],[8.9 8.7 8.6],'color','b');
p6=line([5.5 16],[15.3 15.3],'color','b');
p6=line([6.5 6.5],[8.5 9.6],'color','b');
p6=line([4 5],[10.25 10],'color','b');
p6=line([4.2 4 4.2],[10.4 10.25 10.05],'color','b');
p6=line([6.3 6.5 6.7],[9.3 9.6 9.3],'color','b');
p6=line([15.7 16 15.7],[15.5 15.3 15.1],'color','b');
p6=line([12.2 11.9 12.2],[8.7 8.5 8.3],'color','b');
plot([5 5.5 7.5 7.5 10 13 15.5 6.5 4 4 10 10 15.5],[10 10 10 11.5 10 10 10 8.5
10.25 9.75 10.5 13 11.5],'.b','markersize',12);
% bo ma hao 2
p1=line([13.4 14.6 14.6 13.4 13.4],[1.5 1.5 2.5 2.5 1.5],'color','b');
p2=line([10.9 12.1 12.1 10.9 10.9],[1.5 1.5 2.5 2.5 1.5],'color','b');
p3=line([8.4 9.6 9.6 8.4 8.4],[1.5 1.5 2.5 2.5 1.5],'color','b');
plot(6.5,2,'ob','markersize',12);
plot(6.5,2,'+b');
plot(11.5,0.5,'ob','markersize',12);
plot(11.5,0.5,'+b');
p4=line([6.9 8.4],[2 2],'color','b');
p5=line([8.1 8.4 8.1],[2.2 2 1.8],'color','b');
p6=line([9.6 10.9],[2 2],'color','b');
```

```
p7=line([12.1 13.4],[2 2],'color','b');
p8=line([10.6 10.9 10.6],[2.2 2 1.8],'color','b');
p9=line([13.1 13.4 13.1],[2.2 2 1.8],'color','b');
plot(11.5,3.5,'ob','markersize',12);
plot(11.5,3.5,'+b');
plot(11.5,5.7,'ob','markersize',12);
plot(11.5,5.7,'+b');
p6=line([14.6 15.5],[2 2],'color','b');
p6=line([10 10],[2 5],'color','b');
p6=line([15.5 15.5],[0.5 5.7],'color','b');
p6=line([13 13],[1 3],'color','b');
p6=line([7.5 7.5],[2 5.7],'color','b');
p6=line([7.5 15.5],[5.7 5.7],'color','b');
p6=line([10.8 11.2 10.8],[5.9 5.7 5.5],'color','b');
p6=line([12.2 11.9 12.2],[5.9 5.7 5.5],'color','b');
p6=line([10 11.2],[5 5.4],'color','b');
p6=line([11 11.2 11.1],[5.5 5.4 5.1],'color','b');
p6=line([7.5 15.5],[3.5 3.5],'color','b');
p6=line([10.8 11.1 10.8],[3.7 3.5 3.3],'color','b');
p6=line([12.2 11.9 12.2],[3.7 3.5 3.3],'color','b');
p6=line([13 11.7],[3 3.3],'color','b');
p6=line([11.9 11.7 12],[3.1 3.3 3.4],'color','b');
p6=line([11.5 11.5],[6 6.7],'color','b');
p6=line([11.5 16],[6.7 6.7],'color','b');
p6=line([15.7 16 15.7],[6.9 6.7 6.5],'color','b');
p6=line([10 11.2],[2.5 3.3],'color','b');
p6=line([10.8 11.2 10.9],[3.4 3.3 2.8],'color','b');
p6=line([11.5 11.5],[3.8 4.5],'color','b');
p6=line([11.5 16],[4.5 4.5],'color','b');
p6=line([15.7 16 15.7],[4.7 4.5 4.3],'color','b');
```

```
p6=line([13 11.8],[1 0.7],'color','b');
p6=line([5 6.2],[2 2],'color','b');
p6=line([3 4],[2.25 2.25],'color','b');
p6=line([4 4],[1.75 0.5],'color','b');
p6=line([4 15.5],[0.5 0.5],'color','b');
p6=line([6 6.2 6],[2.2 2 1.8],'color','b');
p6=line([5.5 5.5],[2 7.3],'color','b');
p6=line([11.95 11.8 12.1],[0.9 0.7 0.6],'color','b');
p6=line([5.5 16],[7.3 7.3],'color','b');
p6=line([6.5 6.5],[0.5 1.6],'color','b');
p6=line([4 5],[2.25 2],'color','b');
p6=line([4.2 4 4.2],[2.4 2.25 2.05],'color','b');
p6=line([6.3 6.5 6.7],[1.3 1.6 1.3],'color','b');
p6=line([15.7 16 15.7],[7.5 7.3 7.1],'color','b');
p6=line([12.2 11.9 12.2],[0.7 0.5 0.3],'color','b');
plot([5 5.5 7.5 7.5 10 13 15.5 6.5 4 4 10 10 15.5],[2 2 2 3.5 2 2 2 0.5 2.25 1.75 2.5
5 3.5],'.b','markersize',12);
% bo chen
p6=line([1.5 4.5 4.5 1.5 1.5],[5 5 7 7 5],'color','b');
p6=line([3 3],[10.25 7 ],'color','b');
p6=line([3 3],[5 2.25],'color','b');
text(1.9,6,'interleaver','fontsize',11.5,'color','r');
text(14.5,15.7,'X','fontsize',12,'color','r');
text(14.5,14.2,'Y','fontsize',12,'color','r');
text(14.9,14.1,'0','fontsize',8,'color','r');
text(14.5,12,'Y','fontsize',12,'color','r');
text(14.9,11.9,'1','fontsize',8,'color','r');
text(14.5,7.7,'X'','fontsize',12,'color','r');
text(14.5,6.2,'Y'','fontsize',12,'color','r');
text(14.9,6.1,'0','fontsize',8,'color','r');
```

```
text(14.5,4,'Y''','fontsize',12,'color','r');
text(14.9,3.9,'1','fontsize',8,'color','r');
uicontrol('Parent',h0,...
        'BackgroundColor',[ 0.917647058823529 0.658823529411765
        0.917647058823529 ],...
        'FontName','VNI-TIMES',...
        'Callback','close all;ketqua_mh;',...
        'FontSize',13,...
        'Position',[700 8.25 100.75 21.75],...
        'String','CONTINUE',...
        'Tag','Pushbutton3');
uicontrol('Parent',h0,...
        'BackgroundColor',[ 0.917647058823529 0.658823529411765
        0.917647058823529 ],...
        'Callback','close all; input1;',...
        'FontName','VNI-TIMES',...
        'FontSize',13,...
        'Position',[8.75 8.25 80.25 21.75],...
        'String','BACK',...
        'Tag','Pushbutton4');
uicontrol( 'Parent',h0,...
        'FontSize',13, 'HorizontalAlignment','center',...
        'Backgroundcolor','w','Position',[100 335 90 30],...
        'String',vao,...
        'Style','edit','Tag','EditText');
uicontrol('Parent',h0,...
        'BackgroundColor',[ 0.917647058823529 0.658823529411765
        0.917647058823529 ],...
        'FontName','VNI-TIMES',...
        'Callback','close all;',...
```

```
        'FontSize',13,...
        'Position',[380 8.25 81.75 21.75],...
        'String','EXIT',...
        'Tag','Pushbutton3');
        [kqchen,n_nh,dodai]=chen(vao);
uicontrol( 'FontSize',13, 'HorizontalAlignment','center',...
        'Backgroundcolor','w','Position',[160 160 90 30],...
        'String',kqchen,...
        'Style','edit','Tag','EditText8sc');
++++++++++++++++++++++++++++++++++++++++++++++++++
File ketqua_mh.m
++++++++++++++++++++++++++++++++++++++++++++++++++
h0 = figure('Color',[1 0.819607843137255 0.941176470588235],...
        'NumberTitle','off',...
        'MenuBar','none','Name','KET QUA MA HOA',...
        'PaperPosition',[18 30 576 432],...
        'PaperUnits','points','Position',[1 29 800 553],...
        'Tag','Fig1','ToolBar','none');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
        'FontName','vni-times','FontSize',20,...
        'FontWeight','bold','ListboxTop',0,...
        'Position',[147 350.25 305.25 27.75],...
        'String','Kết quả ra sau khi mã hoá',...
        'Style','text','Tag','StaticTextdau');
[y,y1,y2]=mahoa_turbo(vao);
y2=num2str(y2);
y1=num2str(y1);
y=num2str(y);
```

```
[x,x1,x2]=mahoa_turbo(kqchen);
x2=num2str(x2);
x1=num2str(x1);
x=num2str(x);
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[12.75 225.25 30.25 18.75],...
        'String','X :',...
        'Style','text','Tag','StaticText1');
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[50 222.75 200 22.5],...
        'String',y,...
        'Style','edit','Tag','EditText6');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[12.75 200.25 30.25 18.75],...
        'String','Y1 :',...
        'Style','text','Tag','StaticText1');
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[50 198.75 200 22.5],...
        'String',y1,...
        'Style','edit','Tag','EditText5');
```

```
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[12.75 178.25 30.25 18.75],...
        'String','Y2 :',...
        'Style','text','Tag','StaticText1');
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[50 175 200 22.5],...
        'String',y2,...
        'Style','edit','Tag','EditText1');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[350 225.25 35.25 18.75],...
        'String','X' :',...
        'Style','text','Tag','StaticText1');
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[390 222.75 200 22.5],...
        'String',x,...
        'Style','edit','Tag','EditText');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
```

```
        'HorizontalAlignment','left',...
        'Position',[350 200.25 35.25 18.75],...
        'String','Y’1 :',...
        'Style','text','Tag','StaticText1');
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[390 198.75 200 22.5],...
        'String',x1,...
        'Style','edit','Tag','EditText3');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[350 178.25 35.25 18.75],...
        'String','Y’2 :',...
        'Style','text','Tag','StaticText1');
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[390 175 200 22.5],...
        'String',x2,...
        'Style','edit','Tag','EditText1');
y=str2num(y);
y1=str2num(y1);
y2=str2num(y2);
x=str2num(x);
x1=str2num(x1);
x2=str2num(x2);
dodai=length(vao);
out=chuoi_truyen(y,y1,y2,x,x1,x2,tyle,dodai)
```

```
for i=1:length(out)
 tr(i)=num2str(out(i));
end
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[20 130.25 130.25 18.75],...
        'String','Chuỗi tin truyền đi :',...
        'Style','text','Tag','StaticText1');
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[150 128 400 22.5],...
        'String',tr,...
        'Style','edit','Tag','EditText1');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[80 270 150.25 18.75],...
        'String','chuỗi dữ liệu vào :',...
        'Style','text','Tag','StaticText1');
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[200 268.75 115 22.5],...
        'String',vao,...
        'Style','edit','Tag','EditText1');
h1 = uicontrol('Parent',h0,...
```

```
        'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'Callback','close;',...
        'FontName','vni-times', 'FontSize',16,...
        'ListboxTop',0,...
        'Position',[260 0 83.25 24.75],...
        'String','EXIT', 'Tag','Pushbutton1');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'FontName','vni-times',...
        'FontSize',16,'ListboxTop',0,...
        'callback','close all;sdgm_turbo3',...
        'Position',[510 0 100 25],...
        'String','CONTINUE','Tag','Pushbutton2');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'FontName','vni-times',...
        'FontSize',16,'ListboxTop',0,...
        'callback','close all;mahoa2',...
        'Position',[0 0 80 25],...
        'String','BACK','Tag','Pushbutton2');


++++++++++++++++++++++++++++++++++++++++++++++
File sdgm_turbo3.m
++++++++++++++++++++++++++++++++++++++++++++++


n=dodai;
```

```
h0 = figure('Color',[1 0.819607843137255 0.941176470588235],...
        'MenuBar','none', 'Name','SO DO GIAI MA TURBO',...
        'NumberTitle','off', 'PaperPosition',[18 180 576 432],...
        'PaperUnits','points','Position',[1 29 800 553],...
        'Tag','Fig1','ToolBar','none');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
        'FontName','vni-times',...
        'FontSize',25,...
        'ListboxTop',0,...
        'Position',[148.5 366.75 317.25 36],...
        'String','SƠ ĐỒ GIẢI MÃ SOVA',...
        'Style','text',...
        'Tag','StaticText1');
h2 = uicontrol('Parent',h0, 'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
        'FontName','vni-times', 'FontSize',13, 'ListboxTop',0,...
        'Position',[18 65.25 114 21.75],'String','Chuỗi tin truyền x:',...
        'Style','text','Tag','StaticText4');
h1 = uicontrol('Parent',h0, 'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',15,'ListboxTop',0,...
        'Position',[132.75 58.5 300 27],'String',tr,...
        'Style','edit','Tag','EditText3');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
        'FontName','vni-times','FontSize',13,'ListboxTop',0,...
        'Position',[18.75 36.75 112.5 19.5], 'String','Chuỗi tin nhận y:',...
        'Style','text', 'Tag','StaticText5');
```

```
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
        'FontName','vni-times','FontSize',13,'ListboxTop',0,...
        'Position',[18.75 7.5 113.25 21],'String','Chuỗi tin giải mã u:',...
        'Style','text','Tag','StaticText6');
h1 = uicontrol('Parent',h0,'Units','points', 'BackgroundColor',[1 1 1],...
        'FontSize',15,'ListboxTop',0,'Position',[132 8.25 78 21],...
        'String',vao,'Style','edit','Tag','EditText5');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'FontSize',15,'ListboxTop',0,...
        'Position',[505.5 272.25 76.5 18],'String','RESULT','Tag','Pushbutton2',...
        'callback','close ;ketqua_giaima');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'FontSize',15,'ListboxTop',0,'Position',[507 241.5 76.5 20.25],...
        'String','BACK', 'Tag','Pushbutton4',...
        'callback','close ;ketqua_mh');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'FontSize',15,'ListboxTop',0,'Position',[507 211.5 76.5 20.25],...
        'String','EXIT', 'Tag','Pushbutton4',...
        'callback','close all');
h1 = axes('Parent',h0, 'Box','on','CameraUpVector',[0 1 0],...
        'CameraUpVectorMode','manual','Color',[1 1 1],...
        'NextPlot','add','Tag','Axes1','Visible','off','XColor',[0 0 0],...
        'XGrid','on','XLim',[-1 5+n],'XLimMode','manual',...
```

```
        'YColor',[0 0 0],'YGrid','on','YLim',[0 13],'YLimMode','manual',...
        'ZColor',[0 0 0],'ZGrid','on');
plot([0 3],[11 11],'b.-',[0 1],[11 10],'r.:');
plot([1 2],[11 10],'r.:');
plot([1 2],[10 9],'b.-',[1 2],[10 8],'r.:');
plot([2 3],[11 10],'r.:');
plot([2 3],[10 9],'b.-',[2 3],[10 8],'r.:');
plot([2 3],[9 7],'r.:',[2 3],[9 6],'b.-');
plot([2 3],[8 5],'r.:',[2 3],[8 4],'b.-');
for i=3:3+n-1
 plot([i i+1],[11 11],'b.-',[i i+1],[11 10],'r.:');
 plot([i i+1],[10 9],'b.-',[i i+1],[10 8],'r.:');
 plot([i i+1],[9 7],'r.:',[i i+1],[9 6],'b.-');
 plot([i i+1],[8 5],'r.:',[i i+1],[8 4],'b.-');
 plot([i i+1],[7 11],'r.:',[i i+1],[7 10],'b.-');
 plot([i i+1],[6 9],'r.:',[i i+1],[6 8],'b.-');
 plot([i i+1],[5 7],'b.-',[i i+1],[5 6],'r.:');
 plot([i i+1],[4 5],'b.-',[i i+1],[4 4],'r.:');
end;
text(-1,11,'000','fontsize',13,'color','k');
text(-1,10,'100','fontsize',13,'color','k');
text(-1,9,'010','fontsize',13,'color','k');
text(-1,8,'110','fontsize',13,'color','k');
text(-1,7,'001','fontsize',13,'color','k');
text(-1,6,'101','fontsize',13,'color','k');
text(-1,5,'011','fontsize',13,'color','k');
text(-1,4,'111','fontsize',13,'color','k');
plot([3 4.5],[2.5 2.5],'b.-');
plot([3 4.5],[2 2],'r.:');
text(4.7,2.5,'Bít vào 0','fontname','vni-times','fontsize',15,'color','k');
```

```
text(4.7,2,'Bít vào 1','fontname','vni-times','fontsize',15,'color','k');
text(-1,12,'state','fontsize',15,'color','k');
for i=0:3+n
 text(i,3.5,num2str(i),'fontsize',10,'color','k');
end
text(4+n,3.5,'time','fontsize',15,'color','k');
nhan=chuoinhan(y,y1,y2,x,x1,x2,tyle,dodai,EbN0);
for i=1:length(nhan)
 ch_nh(i)=num2str(nhan(i));
end
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',15, 'ListboxTop',0,...
        'Position',[132.75 32.25 300 24.75],'String',ch_nh,...
        'Style','edit','Tag','EditText4');
++++++++++++++++++++++++++++++++++++++++++++++++++
File ketqua_giaima.m
++++++++++++++++++++++++++++++++++++++++++++++++++
h0 = figure('Color',[1 0.819607843137255 0.941176470588235],...
        'NumberTitle','off',...
        'MenuBar','none','Name','KET QUA GIAI MA',...
        'PaperPosition',[18 30 576 432],...
        'PaperUnits','points','Position',[1 29 800 553],...
        'Tag','Fig1','ToolBar','none');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[1 0.819607843137255 0.941176470588235],...
        'FontName','vni-times','FontSize',20,...
        'FontWeight','bold','ListboxTop',0,...
        'Position',[147 350.25 305.25 27.75],...
        'String','KẾT QUẢ GIẢI MÃ TURBO',...
```

```
        'Style','text','Tag','StaticText4');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[5.75 225.25 130.25 18.75],...
        'String','Chuỗi dữ liệu vào :',...
        'Style','text','Tag','StaticText1');
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[155.75 222.75 105 22.5],...
        'String',vao,...
        'Style','edit','Tag','EditText3');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[5.75 200.25 160.25 18.75],...
        'String','chuỗi dữ liệu truyền đi:',...
        'Style','text','Tag','StaticText1');
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[155.75 197.75 215 22.5],...
        'String',tr,...
        'Style','edit','Tag','EditText3');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
```

```
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[5.75 175.25 160.25 18.75],...
        'String','chuỗi dữ liệu nhận được:',...
        'Style','text','Tag','StaticText1');
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[155.75 172.75 215 22.5],...
        'String',ch_nh,...
        'Style','edit','Tag','EditText4');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[5.75 150.25 160.25 18.75],...
        'String','chuỗi giải mã được:',...
        'Style','text','Tag','StaticText1');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[375 225.25 160.25 18.75],...
        'String','Chiều dài chuỗi dữ liệu:',...
        'Style','text','Tag','StaticText1');
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[525 222.75 40 22.5],...
        'String',num2str(dodai),...
```

```
        'Style','edit','Tag','EditText3');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[375 200.25 160.25 18.75],...
        'String','Số bít bị nhiễu:',...
        'Style','text','Tag','StaticText1');
loi=xor(nhan,out);
s=0;
for i=1:length(loi)
 s=s+loi(i);
end
s=num2str(s);
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[525 197.75 40 22.5],...
        'String',s,...
        'Style','edit','Tag','EditText3');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[375 175.25 160.25 18.75],...
        'String','số bít giải mã sai:',...
        'Style','text','Tag','StaticText1');
kq = gaima_turbo(vao,EbN0,lan);
for i=1:dodai
```

```
 vao(i)=str2num(vao(i));
end
s=0;
v=xor(vao,kq(1:dodai));
for i=1:dodai
 ketqua(i)=num2str(kq(i));
 s=s+v(i);
end
loi=num2str(s);
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[525 172.75 40 22.5],...
        'String',loi,...
        'Style','edit','Tag','EditText3');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[375 150.25 160.25 18.75],...
        'String','số lần lặp giải mã:',...
        'Style','text','Tag','StaticText1');
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[525 147.75 40 22.5],...
        'String',lan,...
        'Style','edit','Tag','EditText3');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
```

```
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'Callback','close all ;sdgm_turbo3',...
        'FontName','vni-times',...
        'FontSize',15,...
        'ListboxTop',0,...
        'Position',[4.75 3.25 72 21],...
        'String','BACK','Tag','Pushbutton3');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'Callback','close all',...
        'FontName','vni-times',...
        'FontSize',15,...
        'ListboxTop',0,...
        'Position',[260 3.25 72 21],...
        'String','EXIT','Tag','Pushbutton3');
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[155.75 147.75 105 22.5],...
        'String',ketqua,...
        'Style','edit','Tag','EditText3');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'Callback','close all;tinhloi',...
        'FontName','vni-times',...
        'FontSize',15,...
```

```
        'ListboxTop',0,...
        'Position',[520 3.25 80 21],...
        'String','TINH LOI','Tag','Pushbutton3');
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
File tinhloibit_loikhung.m
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

function [ber,fer]= tinhloibit_loikhung(EbN0db,dec_alg,L_total,niter,ferrlim)
EbN0db=str2num(EbN0db);
% Frame size
L_total=str2num(L_total);
dec_alg=str2num(dec_alg);
niter=str2num(niter);
ferrlim=str2num(ferrlim);
g = [ 1 1 1;1 0 1 ];
[n,K] = size(g);
m = K - 1;
nstates = 2^m;
%puncture = 0, puncturing into rate 1/2;
puncture = 0;
% Code rate
rate = 1/(2+puncture);
% Fading amplitude; a=1 in AWGN channel
a = 1;
en = 10^(EbN0db/10); % convert Eb/N0 from unit db to normal numbers
L_c = 4*a*en*rate;          % reliability value of the channel
sigma = 1/sqrt(2*rate*en);                    % standard deviation of
AWGN noise
% Clear bit error counter and frame error counter
errs = zeros(1,niter);
```

```
nferr = zeros(1,niter);
nframe = 0; % clear counter of transmitted frames
while nferr(niter)<ferrlim
 nframe = nframe + 1;
 x = round(rand(1, L_total-m)); % info. bits
 [temp, alpha] = sort(rand(1,L_total)); % random interleaver mapping
 en_output = encoderm( x, g, alpha, puncture ) ; % encoder output (+1/-1)

 r = en_output+sigma*randn(1,L_total*(2+puncture)); % received bits
 yk = demultiplex(r,alpha,puncture); % demultiplex to get input for decoder 1
and 2

% Scale the received bits
 rec_s = 0.5*L_c*yk;

% Initialize extrinsic information
 L_e(1:L_total) = zeros(1,L_total);

 for iter = 1:niter
% Decoder one
 L_a(alpha) = L_e; % a priori info.
 if dec_alg == 0
 L_all = logmapo(rec_s(1,:), g, L_a, 1); % complete info.
 else
 L_all = sova0(rec_s(1,:), g, L_a, 1); % complete info.
 end
 L_e = L_all - 2*rec_s(1,1:2:2*L_total) - L_a; % extrinsic info.

% Decoder two
 L_a = L_e(alpha); % a priori info.
```

```
if dec_alg == 0
L_all = logmapo(rec_s(2,:), g, L_a, 2); % complete info.
else
L_all = sova0(rec_s(2,:), g, L_a, 2); % complete info.
end
L_e = L_all - 2*rec_s(2,1:2:2*L_total) - L_a; % extrinsic info.

% Estimate the info. bits
xhat(alpha) = (sign(L_all)+1)/2;

% Number of bit errors in current iteration
err(iter) = length(find(xhat(1:L_total-m)~=x));
% Count frame errors for the current iteration
if err(iter)>0
nferr(iter) = nferr(iter)+1;
end
end                          %iter

% Total number of bit errors for all iterations
errs(1:niter) = errs(1:niter) + err(1:niter);



end                                                  %while
ber=errs/nframe/ (L_total-m );
fer=nferr/nframe;
```

++

File encoder_bit.m

++

```
function [output, state] = encode_bit(g, input, state)
```

```
[n,k] = size(g);
m = k-1;

for i=1:n
 output(i) = g(i,1)*input;
 for j = 2:k
 output(i) = xor(output(i),g(i,j)*state(j-1));
 end;
end

state = [input, state(1:m-1)];
```

++

File encoder.m

++

```
%------------------------------------------------------------------------
function y = rsc_encode(g, x, terminated)

[n,K] = size(g);
m = K - 1;
if terminated>0
 L_info = length(x);
 L_total = L_info + m;
else
 L_total = length(x);
 L_info = L_total - m;
end

% initialize the state vector
```

```
state = zeros(1,m);

% generate the codeword
for i = 1:L_total
 if terminated<0 | (terminated>0 & i<=L_info)
 d_k = x(1,i);
 elseif terminated>0 & i>L_info
 % terminate the trellis
 d_k = rem( g(1,2:K)*state', 2 );
 end

 a_k = rem( g(1,:)*[d_k state]', 2 );
 [output_bits, state] = encode_bit(g, a_k, state);
 % since systematic, first output is input bit
 output_bits(1,1) = d_k;
 y(n*(i-1)+1:n*i) = output_bits;
end
```

++

File tinhloi.m

++

```
h0 = figure('Color',[1 0.819607843137255 0.941176470588235],...
        'NumberTitle','off',...
        'MenuBar','none','Name','NHAP THONG SO TNH TY LE LOI BIT VA TY
        LE LOI KHUNG',...
        'PaperPosition',[18 30 576 432],...
        'PaperUnits','points','Position',[1 29 800 553],...
        'Tag','Fig1','ToolBar','none');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
```

```
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[50 300 300 18.75],...
        'String','Thuật toán giải mã log/sova(0/1):',...
        'Style','text','Tag','StaticText1');
hdl = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[260 298 105 22.5],...
        'String','',...
        'Style','edit','Tag','EditText1thuattoan',...
        'callback', 'thuattoan');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[50 275 158.25 18.75],...
        'String','Chọn kích thước khung:',...
        'Style','text','Tag','StaticText1');
hd2 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[260 273 105 22.5],...
        'String','',...
        'Style','edit','Tag','EditText2kthuoc',...
        'callback', 'kichthuoc');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
```

```
        'Position',[50 250 158.25 18.75],...
        'String','Số lần lặp :',...
        'Style','text','Tag','StaticText1');
hd3 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[260 247 105 22.5],...
        'String','',...
        'Style','edit','Tag','EditText3lanlap',...
        'callback', 'lanlap');

h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[50 225 158.25 18.75],...
        'String','Chọn tỷ số năng lượng :',...
        'Style','text','Tag','StaticText1');
hd4 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[260 223 105 22.5],...
        'String','',...
        'Style','edit','Tag','EditText4nangluong',...
        'callback', 'nangluong');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 0.819607843137255
        0.941176470588235],'ListboxTop',0,...
        'FontName','vni-times', 'FontSize',15,...
        'HorizontalAlignment','left',...
        'Position',[50 200 158.25 18.75],...
```

```
        'String','chọn số khung bị lỗi :',...
        'Style','text','Tag','StaticText1');
hd5 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[1 1 1],'FontSize',13,...
        'ListboxTop',0,'Position',[260 198 105 22.5],...
        'String','',...
        'Style','edit','Tag','EditText5sokhungloi',...
        'callback', 'khungloi');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'Callback','close',...
        'FontName','vni-times',...
        'FontSize',15,...
        'ListboxTop',0,...
        'Position',[270.75 3.25 72 21],...
        'String','EXIT','Tag','Pushbutton1');
h1 = uicontrol('Parent',h0,'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'FontName','VNI-TIMES',...
        'FontSize',15,'ListboxTop',0,...
        'Position',[510.25 3.25 84.75 21],...
        'callback','close all ;vedothi',...
        'String','VEDOTHI','Tag','Pushbutton2');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
```

```
        'Callback','close all ; ketqua_giaima',...
        'FontName','vni-times',...
        'FontSize',15,...
        'ListboxTop',0,...
        'Position',[4.75 3.25 72 21],...
        'String','BACK','Tag','Pushbutton3');
```

++

File vedothi.m

++

```
h0 = figure('Color',[1 0.819607843137255 0.941176470588235],...
        'MenuBar','none', 'Name','HIEN THI KET QUA TINH LOI BIT VA LOI
        KHUNG',...
        'NumberTitle','off', 'PaperPosition',[18 180 576 432],...
        'PaperUnits','points','Position',[1 29 800 553],...
        'Tag','Fig1','ToolBar','none');
axis([0 18 0 16]);
axis ;
hold on;
grid off;
[ber,fer]= tinhloibit_loikhung(EbN0db,dec_alg,L_total,niter,ferrlim);
niter=str2num(niter);
x=(1:1:niter);
subplot(1,2,1);
plot(x,ber);
xlabel('so lan lap');
ylabel('ty le loi bit');
subplot(1,2,2);
plot(x,fer);
xlabel('so lan lap');
ylabel('ty le loi khung');
```

```
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'Callback','close all',...
        'FontName','vni-times',...
        'FontSize',15,...
        'ListboxTop',0,...
        'Position',[520 3.25 72 21],...
        'String','END','Tag','Pushbutton3');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'Callback','close all;tinhloi',...
        'FontName','vni-times',...
        'FontSize',15,...
        'ListboxTop',0,...
        'Position',[0 3.25 72 21],...
        'String','BACK','Tag','Pushbutton3');
h1 = uicontrol('Parent',h0,...
        'Units','points',...
        'BackgroundColor',[0.917647058823529 0.658823529411765
        0.917647058823529],...
        'Callback','close all',...
        'FontName','vni-times',...
        'FontSize',15,...
        'ListboxTop',0,...
        'Position',[260 3.25 72 21],...
        'String','EXIT','Tag','Pushbutton3');
```

MỘT SỐ TÀI LIỆU THAM KHẢO

1. Alister Burr *"Modulation and Coding for wireless Communications"*, Prentice Hall,2001
2. C.Richard Johnson et al., *"Telecommunication Breakdown",* Prentice Hall, 2004.
3. John B. Anderson, *"Digital Transmission Engineering",* Prentice Hall,1999.
4. M. C. Valenti, *Turbo Codes and Iterative Processing*, in proc IEEE New Zealand Wireless Comm., Symp. '98 (Aukland, New Zealand), Nov. (1998).
5. *Performance of Multi Binary Turbo Codes on Rice Flat Fading Channels* - Horia Balta, Maria Kovaci,Miranda Naforniță Department of Communication, University of Timişoara, Faculty ofTelecommunication, Postal address, 12345 Timişoara, România, E-Mail: balta@etc.upt.ro, kmaria@etc.upt.ro, monica.nafornita@etc.upt.ro
6. Sergio Benedetto and Ezio Biglieri, *"Principles of Digital Transmission with wireless amplication",* Kluwer, 1999.
7. Sklar, *"Digital Communication"* 2 edition, Mc Graw-Hill, 2001.
8. *Turbo Code Applications* a Journey from a Paper to realization Sripimanwat, Keattisak (Ed.)2005, XXII, 386 p., Hardcover. ISBN: 978-1-4020-3686-6

9. *Turbo Codes* - Desirable and Designable Giulietti, Alexandre, Bougard, Bruno, Van Der Perre, Liesbet 2003, 162 p., Hardcover ISBN: 978-1-4020-7660-2
10. *Turbo Codes* - Principles and Applications. Series: The Springer International Series in Engineering and Computer Science, Vol. 559. Vucetic, Branka, Jinhong Yuan 2000, 344 p., Hardcover. ISBN: 978-0-7923-7868-6
11. *Turbo Coding* - Series: The Springer International Series in Engineering and Computer Science, Vol. 476. Heegard, Chris, Wicker, Stephen B.1999, 232 p., Hardcover.ISBN: 978-0-7923-8378-9
12. X. Wang, H. V. Poor, *Iterative (Turbo) Soft Interference Cencellation anh Decoding for Coded CDMA*, IEEE Trans. on Comm., Vol. 47, No. 7, July (1999).
13. Yufei, *" Matlab code for experiment on turbo codes"* (updated June 07,1999)
14. *"Turbo",* Luận Văn Thạc Sĩ Kỹ Thuật, Học viện khoa học kỹ thuật quân sự, Bộ Quốc phòng, 2003
15. *Chất lượng các bộ tách song trong hệ thông đa truy cập CDMA có mã hóa Turbo*- Vũ Đình Thành, Lê Ngọc Phúc -Trường Đại Học Bách Khoa, ĐHQ-HCM (Bài nhận ngày 30 tháng 04 năm 2006, hoàn chỉnh sửa chữa ngày 26 tháng 02 năm 2007)
16. Đặng Văn Chuyết, Nguyễn Tuấn Anh. *Cơ sở lý thuyết truyền tin*. NXB Giáo Dục, 1998.
17. Nguyễn Văn Quang *Nghiên cứu sử dụng mã hóa Turbo vào hệ thống di động thế hệ thứ 3 để chống nhiễu và sửa sai* luận án Thạc sĩ ngành Điện tử - Viễn thông - Điều khiển Trường Đại học Giao Thông Vận Tải - Cơ sở II
18. Phạm Hồng Liên, Chung Thị Ngọc Hạnh; *Nghiên cứu ứng dụng mã TURBO vào hệ thống thông tin di động thế hệ 3* MT- 2000; Tạp chí phát triển khoa học công nghệ Đại học quốc gia Tp.Hồ Chí Minh; 07-2001
19. Tác giả: Nguyễn Hiếu Minh. Nguyễn Văn Hậu. Sách : *Cơ Sở Lý Thuyết Truyền Tin* Nhà xuất bản: Nhà xuất bản Khoa Học Kỹ Thuật

20. Website trực tuyến :

http://en.wikipedia.org

http://tudiencongnghe.com

http://www.ebook.edu.vn

http://books.google.com

http:/.thuvien-ebook.com

KẾT LUẬN

Như vậy, Mã Turbo (hay mã lốc) là một kỹ thuật mã hóa sửa sai (FEC). Turbo Codes thuộc họ mã lưới (mã hóa theo Trellis) và được xây dựng dựa trên 1 mã chập (Convolution Codes).

Sở dĩ gọi là mã Turbo (lốc xoáy) vì cấu trúc giải mã được thực hiện theo giải thuật vòng lặp (iteration) và sau mỗi vòng lặp, tỷ số tín hiệu trên nhiễu SNR sẽ được tăng dần.Mã Turbo là bộ mã có chất lượng tốt nhất so với các bộ mã đã biết từ trước tới nay với độ phức tạp của bộ mã hóa cũng như bộ giải mã chấp nhận được.

Luận văn với đề tài *" Nghiên cứu mã Turbo"* đã nêu được vai trò của mã kênh trong hệ thông tin số, lý thuyết của Shannon và các bộ mã thường được sử dụng để nâng cao chất lượng của hệ thống như mã chập, mã kề. Các chương cũng đã phân tích được cấu trúc và nguyên lý của mã Turbo thông qua quá trình mã hóa và giải mã lặp. Đề tài đã nêu lên hai thuật toán giải mã Turbo đó là thuật toán MAP và thuật toán SOVA. Nội dung của đề tài nêu lên các ứng dụng và hạn chế của mã hóa Turbo trong hệ thống thông tin di động CDMA 2000.

Và cuối cùng là chương trình mô phỏng bằng Matlab về ứng dụng của mã hóa Turbo trong hệ thông thông tin di động CDMA.