EC 721 Advanced Digital Communications
Spring 2008

Mohamed Essam Khedr

Department of Electronics and Communications

Error correcting codes

http://webmail.aast.edu/~khedr

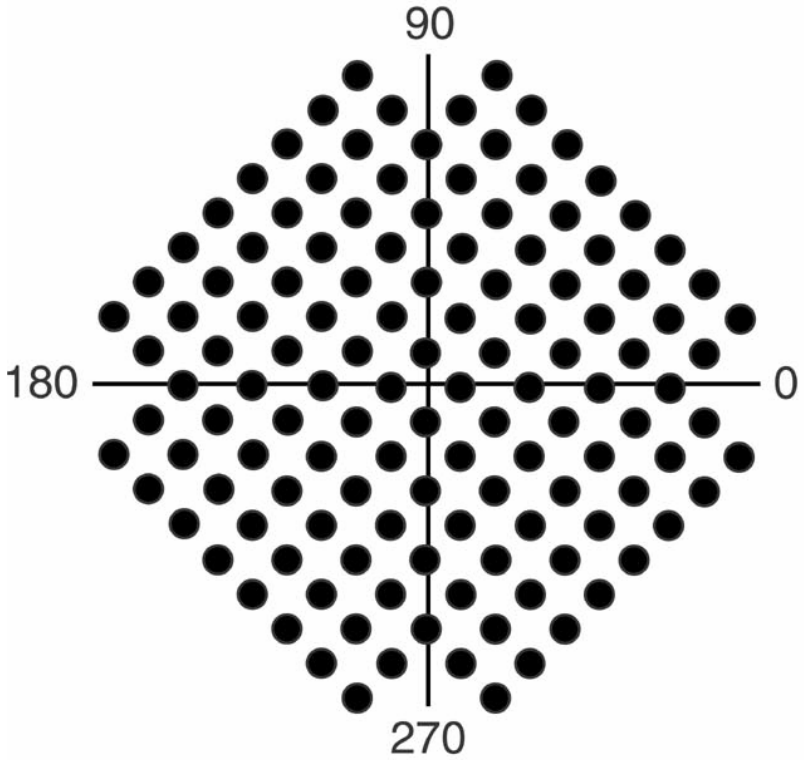
Syllabus

- Tentatively

Week 1	Overview, Probabilities, Random variables
Week 2	Random Process, Optimum Detection
Week 3	Digital Signal Representation
Week 4	Signal space and probability of error
Week 5	Probability of error of M-ary techniques
Week 6	Coding theory
Week 7	Linear block codes
Week 8	Convolutional Codes
Week 9	Convolutional Codes II
Week 10	
Week 11	
Week 12	
Week 13	
Week 14	
Week 15	

OOPS OOPS OOOOOOPS
Report and MATLAB ASSIGNMENT

- Soft and hard decisions
- Puncturing
- Interleaving
  - Block
  - Convolutional
- P_{ec} and P_{euc} for V.32
  6 data bits. 1 parity.
  14.4 kbps. modem



Linear block codes – cont'd

- Error detection capability is given by

$$e = d_{\min} - 1$$

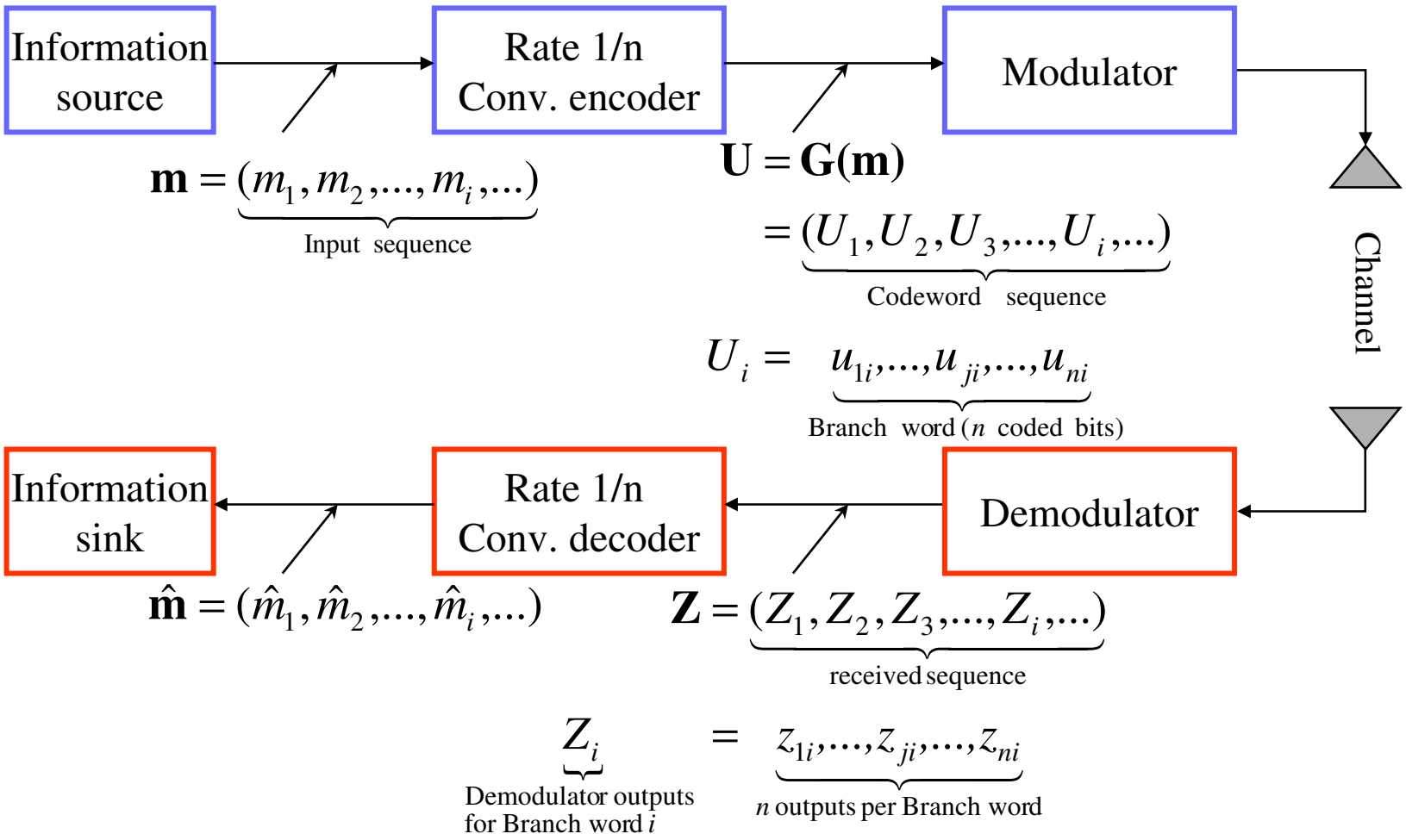
- Error correcting-capability **t** of a code, which is defined as the maximum number of guaranteed correctable errors per codeword, is

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

Convolutional codes-cont'd

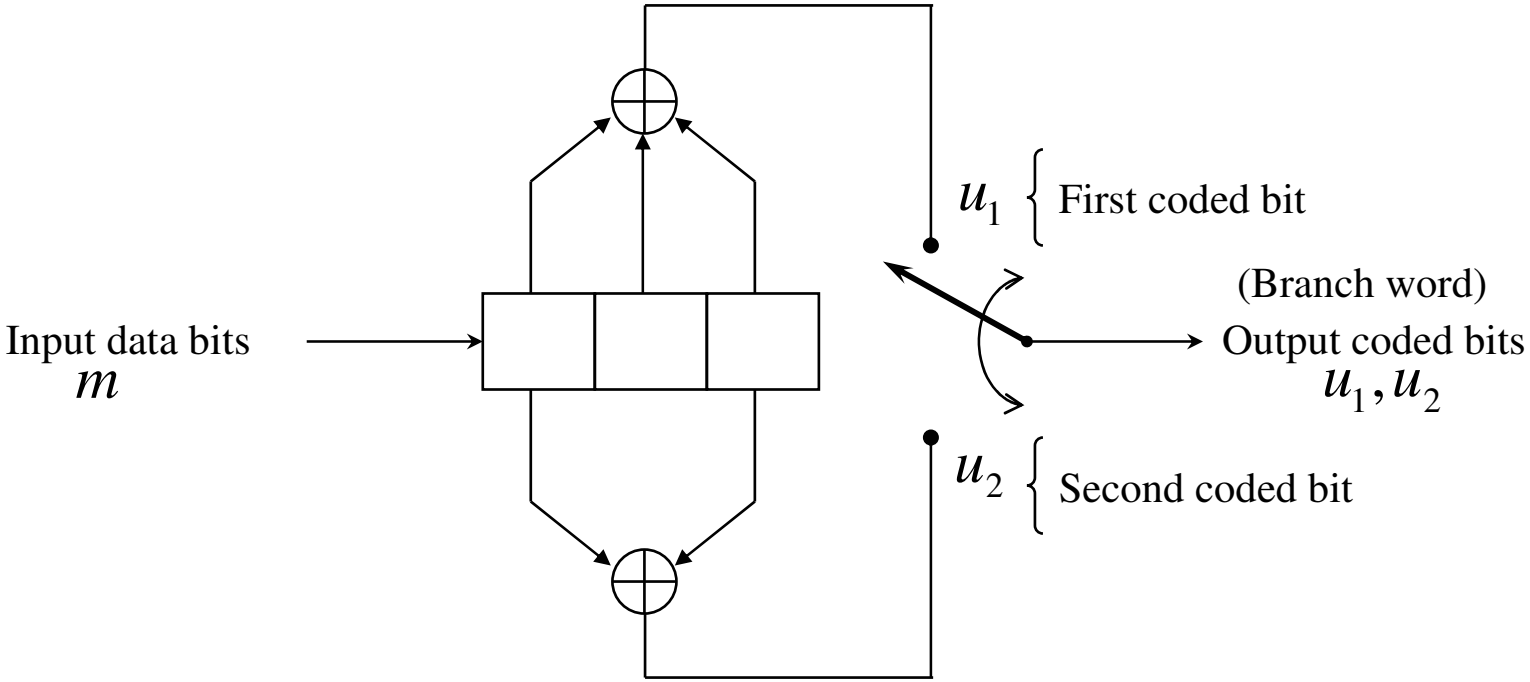
- A Convolutional code is specified by three parameters (n, k, K) or $(k/n, K)$ where
  - $R_c = k/n$ is the coding rate, determining the number of data bits per coded bit.
    - In practice, usually $k=1$ is chosen and we assume that from now on.
  - K is the constraint length of the encoder a where the encoder has K-1 memory elements.
    - There is different definitions in literatures for constraint length.

Block diagram of the DCS



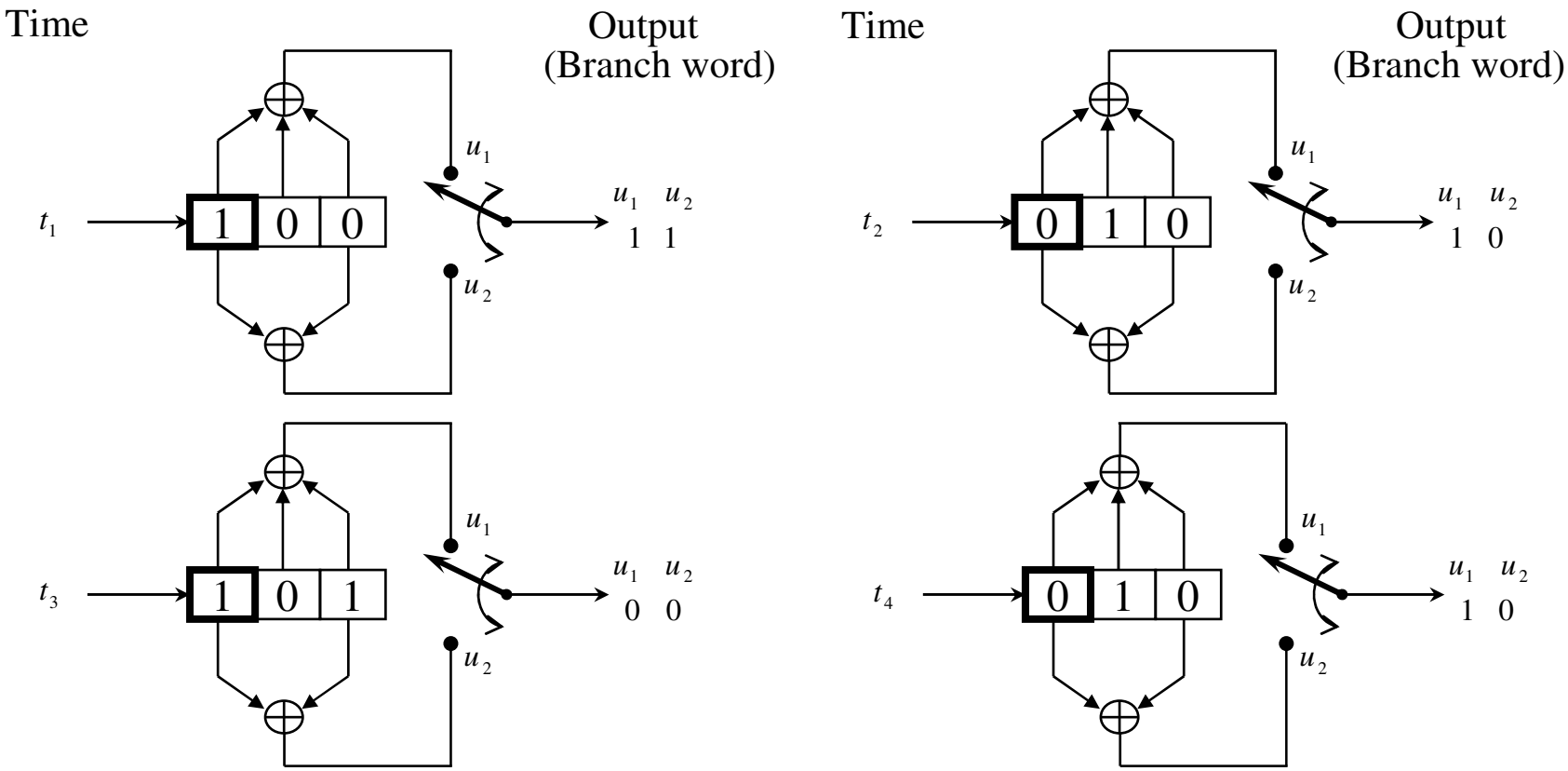
A Rate ½ Convolutional encoder

- Convolutional encoder (rate ½, K=3)
  - 3 shift-registers where the first one takes the incoming data bit and the rest, form the memory of the encoder.

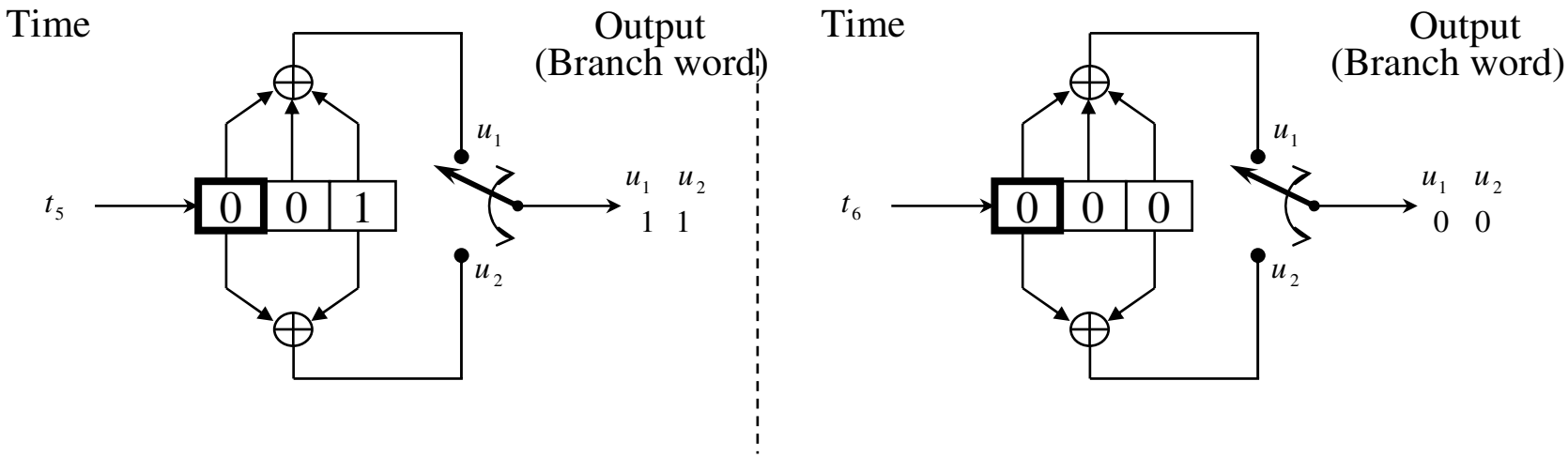


A Rate ½ Convolutional encoder

Message sequence: $\mathbf{m} = (101)$



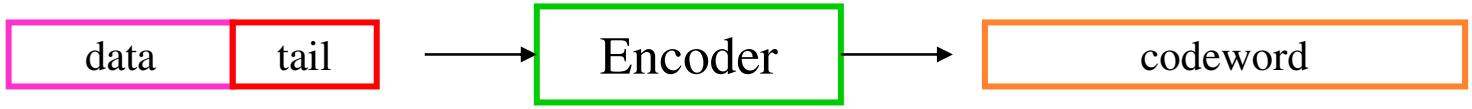
A Rate ½ Convolutional encoder



$\mathbf{m} = (101)$ → Encoder → $\mathbf{U} = (11\ \ 10\ \ 00\ \ 10\ \ 11)$

Effective code rate

- Initialize the memory before encoding the first bit (all-zero)
- Clear out the memory after encoding the last bit (all-zero)
  - Hence, a tail of zero-bits is appended to data bits.



- Effective code rate :
  - L is the number of data bits and $k=1$ is assumed:

$$R_{eff} = \frac{L}{n(L+K-1)} < R_c$$

Encoder representation – cont'd

- Impulse response representation:
  - The response of encoder to a single "one" bit that goes through it.
    - Example:

Register contents	Branch word u_1	u_2
100	1	1
010	1	0
001	1	1

Input sequence: 1 0 0

Output sequence: 11 10 11

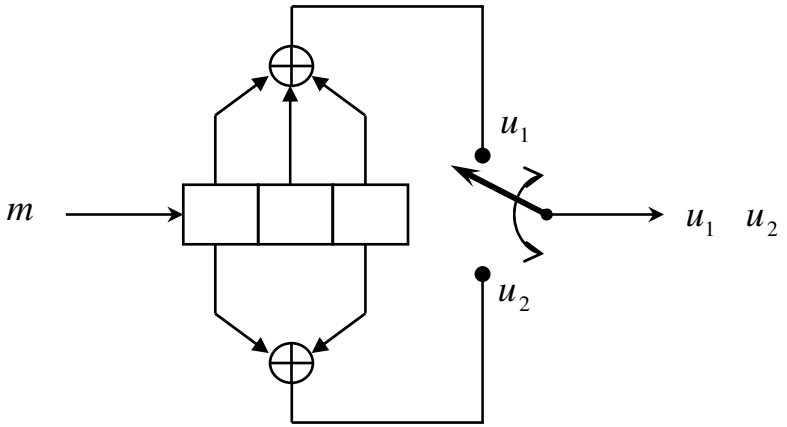
Input **m**	Output				
1	11	10	11		
0		00	00	00	
1			11	10	11
Modulo-2 sum:	11	10	00	10	11

Encoder representation

- Vector representation:
  - We define n binary vector with K elements (one vector for each modulo-2 adder). The i:th element in each vector, is "1" if the i:th stage in the shift register is connected to the corresponding modulo-2 adder, and "0" otherwise.
    - Example:

$$\mathbf{g}_1 = (111)$$

$$\mathbf{g}_2 = (101)$$



Encoder representation – cont'd

- Polynomial representation:
  - We define n generator polynomials, one for each modulo-2 adder. Each polynomial is of degree *K-1* or less and describes the connection of the shift registers to the corresponding modulo-2 adder.
    - Example:

$$\mathbf{g}_1(X) = g_0^{(1)} + g_1^{(1)}.X + g_2^{(1)}.X^2 = 1 + X + X^2$$

$$\mathbf{g}_2(X) = g_0^{(2)} + g_1^{(2)}.X + g_2^{(2)}.X^2 = 1 + X^2$$

The output sequence is found as follows:

$$\mathbf{U}(X) = \mathbf{m}(X)\mathbf{g}_1(X) \text{ interlaced with } \mathbf{m}(X)\mathbf{g}_2(X)$$

Encoder representation –cont'd

In more details:

$$\mathbf{m}(X)\mathbf{g}_1(X) = (1+X^2)(1+X+X^2) = 1+X+X^3+X^4$$

$$\mathbf{m}(X)\mathbf{g}_2(X) = (1+X^2)(1+X^2) = 1+X^4$$

$$\mathbf{m}(X)\mathbf{g}_1(X) = 1+X+0.X^2+X^3+X^4$$

$$\mathbf{m}(X)\mathbf{g}_2(X) = 1+0.X+0.X^2+0.X^3+X^4$$

$$\mathbf{U}(X) = (1,1)+(1,0)X+(0,0)X^2+(1,0)X^3+(1,1)X^4$$

$$\mathbf{U} = 11 \quad 10 \quad 00 \quad 10 \quad 11$$

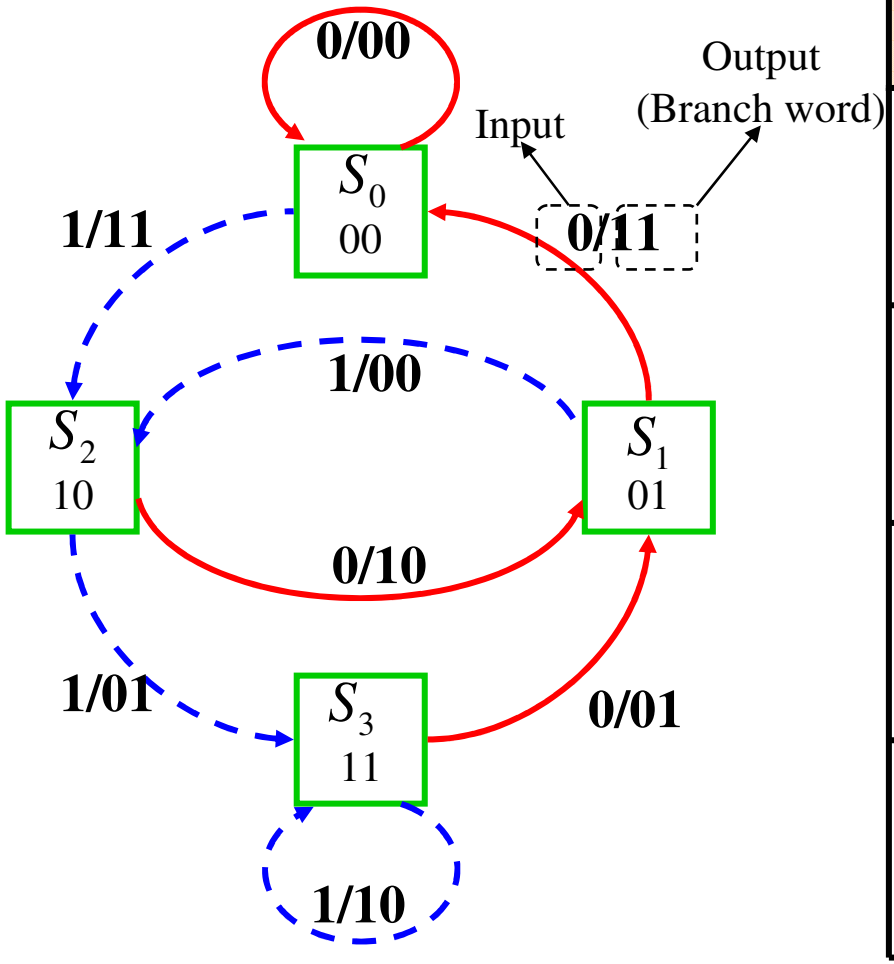
State diagram

- A finite-state machine only encounters a finite number of states.
- State of a machine: the smallest amount of information that, together with a current input to the machine, can predict the output of the machine.
- In a Convolutional encoder, the state is represented by the content of the memory.
- Hence, there are 2^{K-1} states.

State diagram – cont'd

- A state diagram is a way to represent the encoder.
- A state diagram contains all the states and all possible transitions between them.
- Only two transitions initiating from a state
- Only two transitions ending up in a state

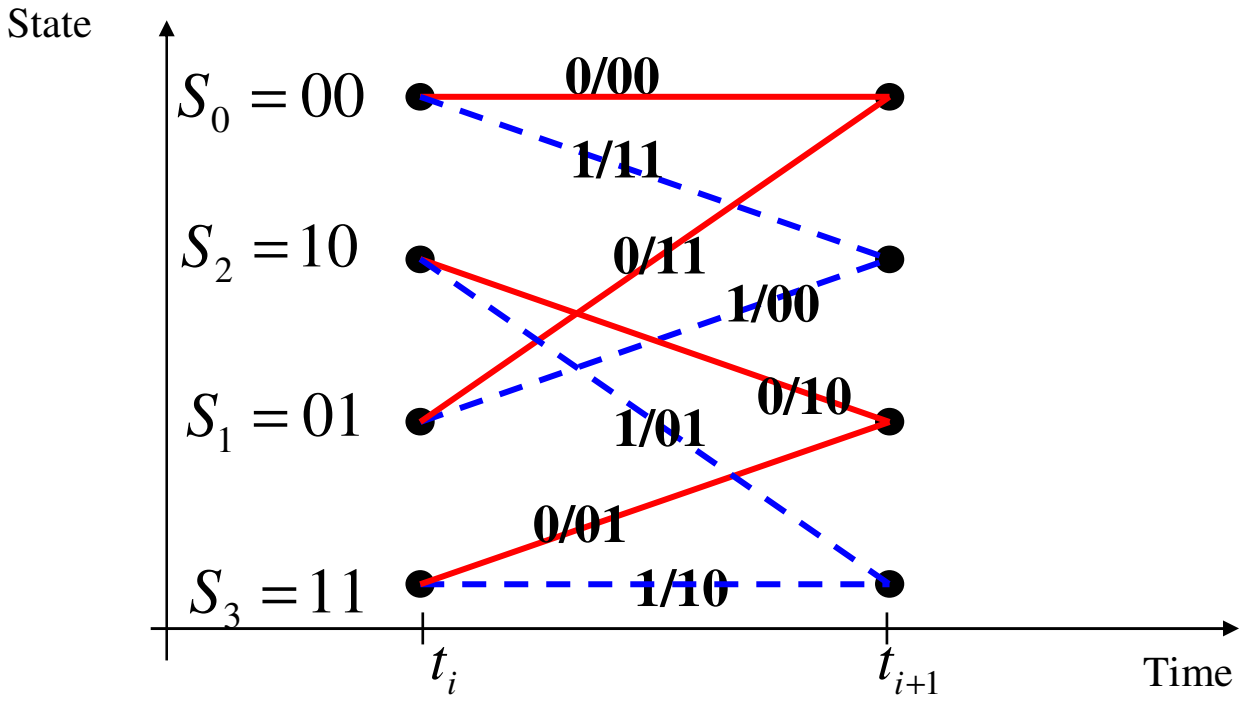
State diagram – cont'd



Current state	input	Next state	output
S_0 00	0	S_0	00
	1	S_2	11
S_1 01	0	S_0	11
	1	S_2	00
S_2 10	0	S_1	10
	1	S_3	01
S_3 11	0	S_1	01
	1	S_3	10

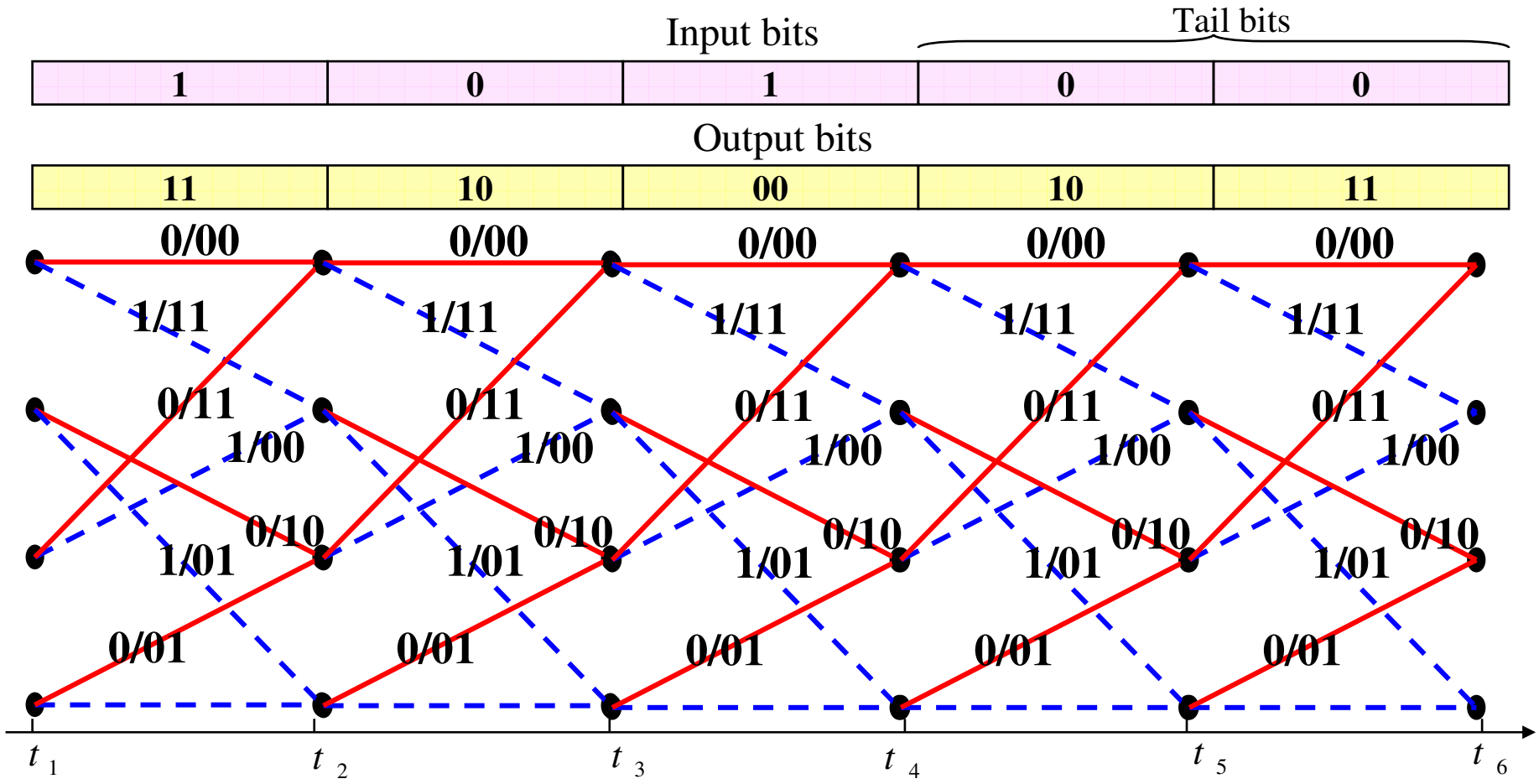
Trellis – cont'd

- Trellis diagram is an extension of the state diagram that shows the passage of time.
  - Example of a section of trellis for the rate ½ code

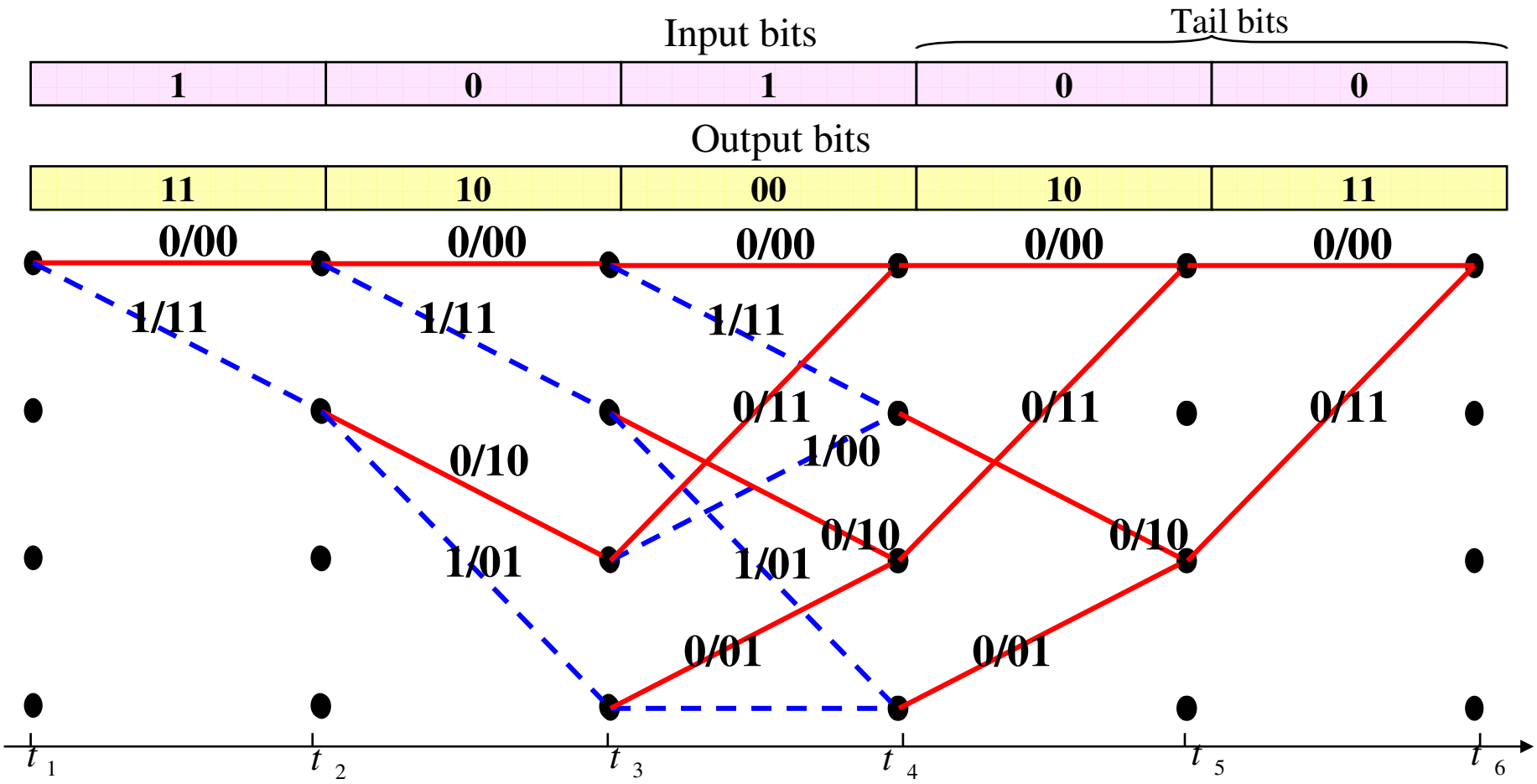


Trellis –cont'd

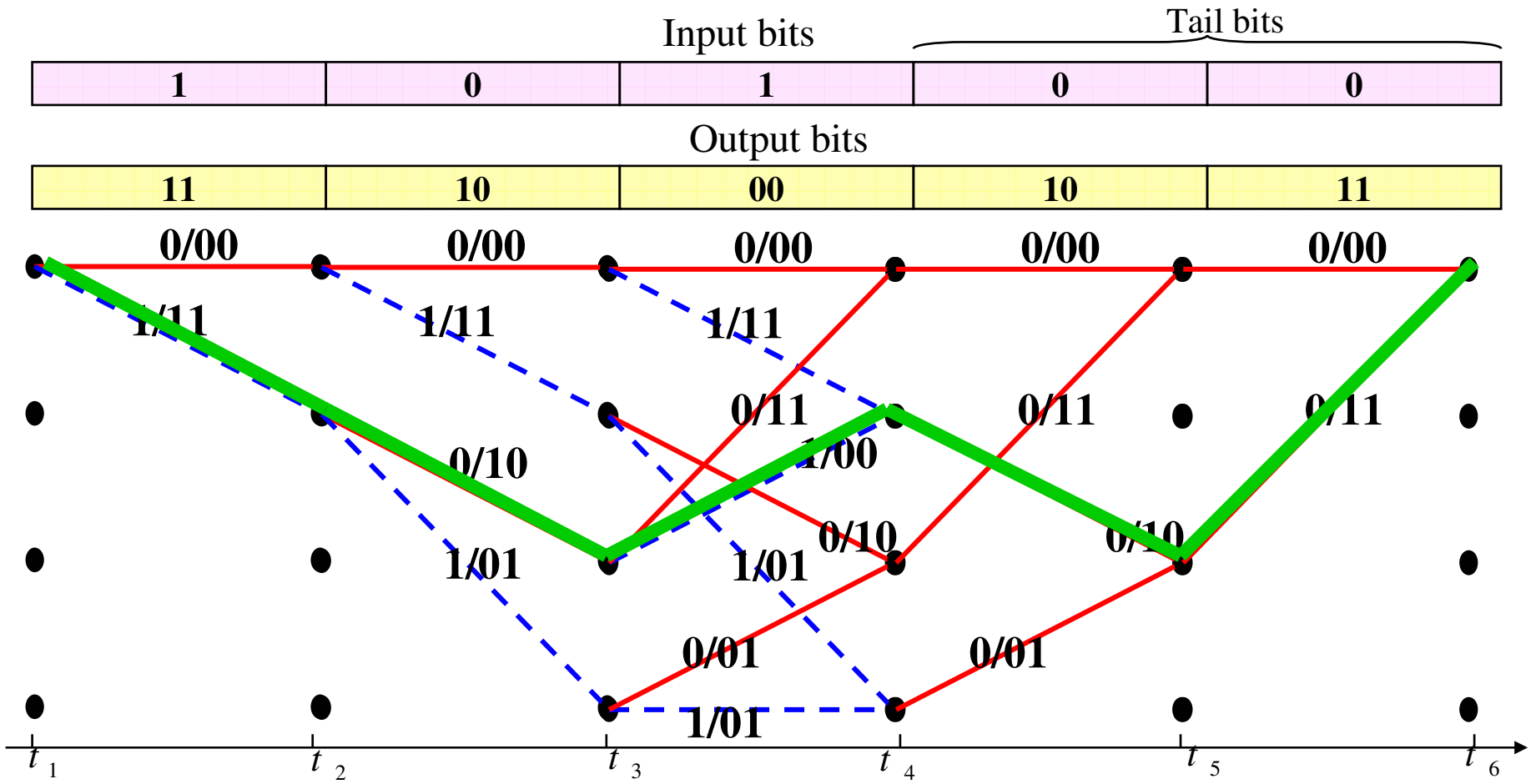
- A trellis diagram for the example code



Trellis – cont'd



Trellis of an example ½ Conv. code



Optimum decoding

- If the input sequence messages are equally likely, the optimum decoder which minimizes the probability of error is the *Maximum likelihood* decoder.

- ML decoder, selects a codeword among all the possible codewords which maximizes the likelihood function $p(\mathbf{Z} \mid \mathbf{U}^{(m')})$ where $\mathbf{Z}$ is the received sequence and $\mathbf{U}^{(m')}$ is one of the possible codewords:

2^L codewords to search!!!

➢ML decoding rule:

Choose $\mathbf{U}^{(m')}$ if $p(\mathbf{Z} \mid \mathbf{U}^{(m')}) = \max_{\text{over all } \mathbf{U}^{(m)}} p(\mathbf{Z} \mid \mathbf{U}^{(m)})$

The Viterbi algorithm

- The Viterbi algorithm performs Maximum likelihood decoding.
- It find a path through trellis with the largest metric (maximum correlation or minimum distance).
  - It processes the demodulator outputs in an iterative manner.
  - At each step in the trellis, it compares the metric of all paths entering each state, and keeps only the path with the largest metric, called the survivor, together with its metric.
  - It proceeds in the trellis by eliminating the least likely paths.
- It reduces the decoding complexity to $L2^{K-1}$!

The Viterbi algorithm - cont'd

- Viterbi algorithm:

A. Do the following set up:
   - For a data block of L bits, form the trellis. The trellis has $L+K-1$ sections or levels and starts at time t_1 and ends up at time t_{L+K} .
   - Label all the branches in the trellis with their corresponding branch metric.
   - For each state in the trellis at the time t_i which is denoted by $S(t_i) \in \{0,1,...,2^{K-1}\}$, define a parameter $\Gamma(S(t_i), t_i)$

B. Then, do the following:

The Viterbi algorithm - cont'd

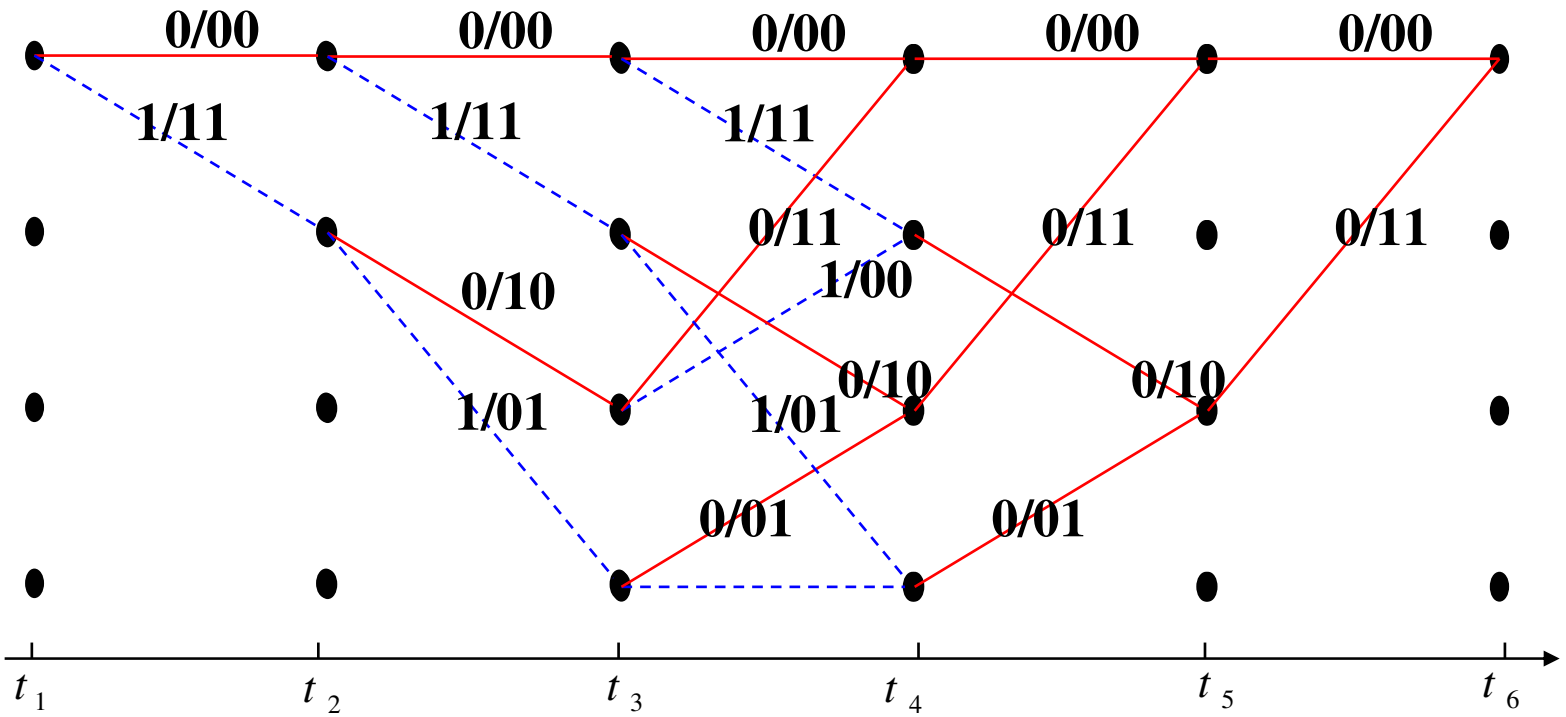
1. Set $\Gamma(0,t_1)=0$ and $i=2.$
2. At time t_i , compute the partial path metrics for all the paths entering each state.
3. Set $\Gamma\left(S(t_i),t_i\right)$ equal to the best partial path metric entering each state at time t_i.

   Keep the survivor path and delete the dead paths from the trellis.
4. If $i<L+K$, increase i by 1 and return to step 2.

C. Start at state zero at time t_{L+K} . Follow the surviving branches backwards through the trellis. The path thus defined is unique and correspond to the ML codeword.

Example of Hard decision Viterbi decoding

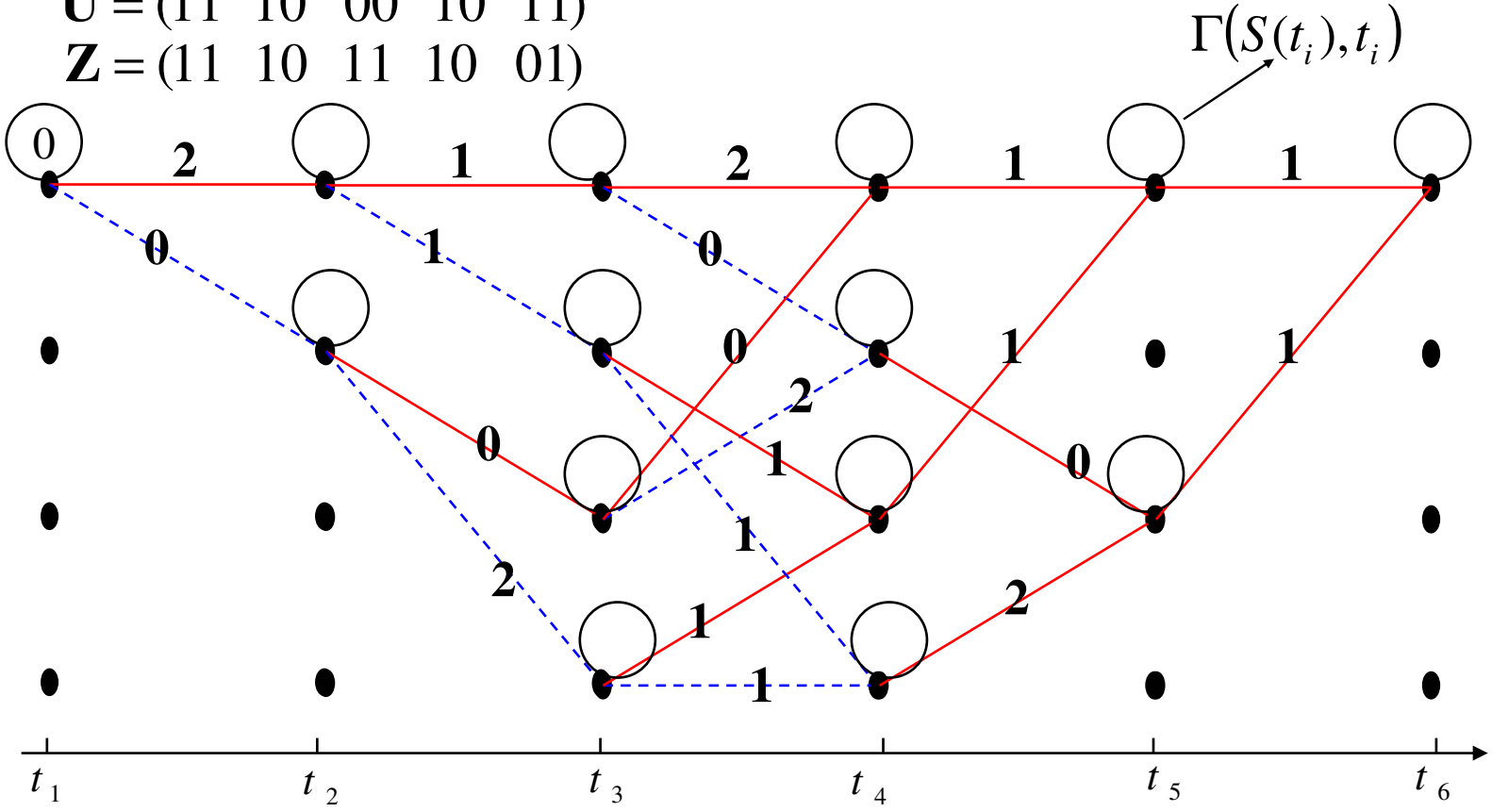
$$\mathbf{m} = (101)$$
$$\mathbf{U} = (11 \;\; 10 \;\; 00 \;\; 10 \;\; 11)$$
$$\mathbf{Z} = (11 \;\; 10 \;\; 11 \;\; 10 \;\; 01)$$



Example of Hard decision Viterbi decoding-cont'd

- Label al the branches with the branch metric (Hamming distance)

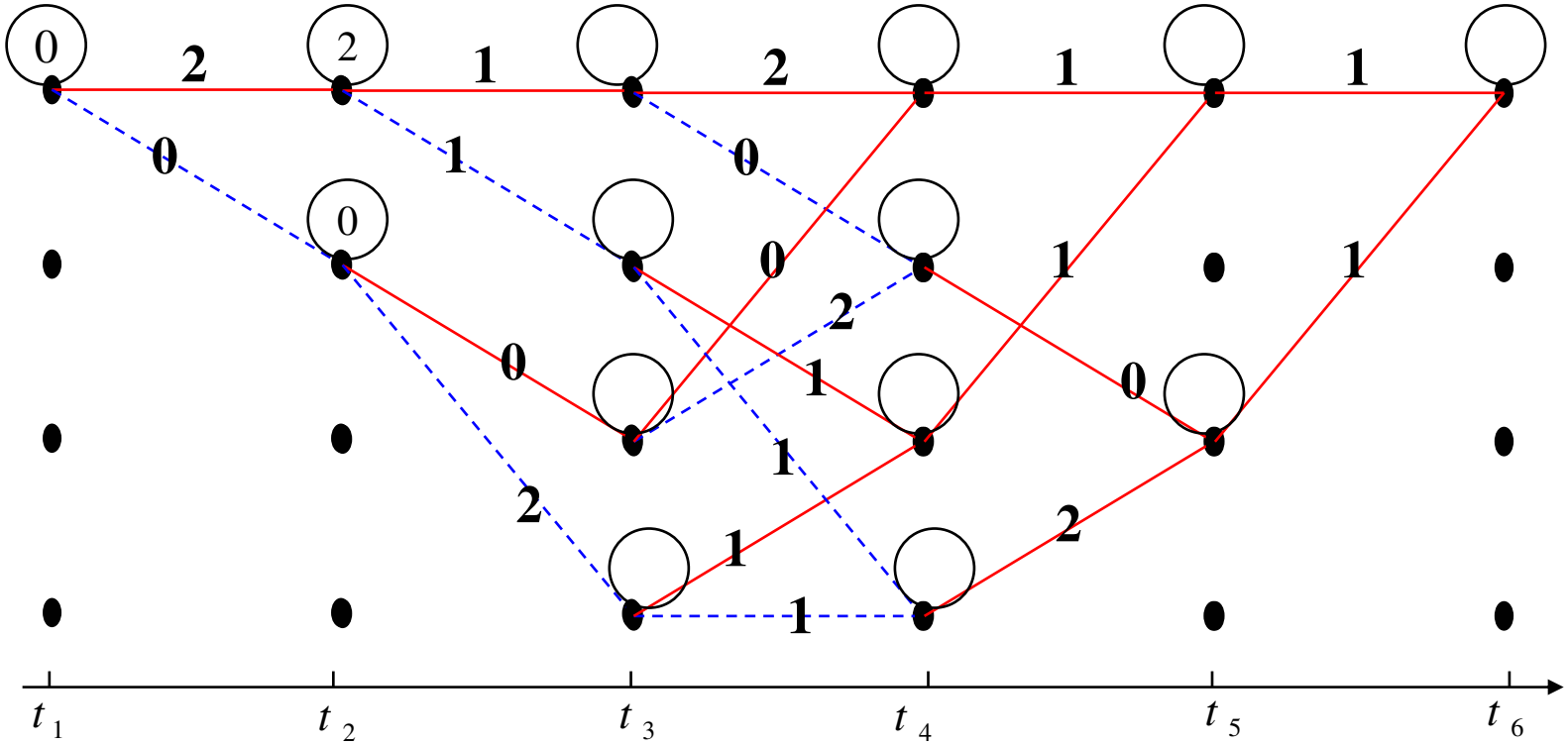
$$\mathbf{m} = (101)$$
$$\mathbf{U} = (11\ \ 10\ \ 00\ \ 10\ \ 11)$$
$$\mathbf{Z} = (11\ \ 10\ \ 11\ \ 10\ \ 01)$$



Example of Hard decision Viterbi decoding-cont'd

- i=2

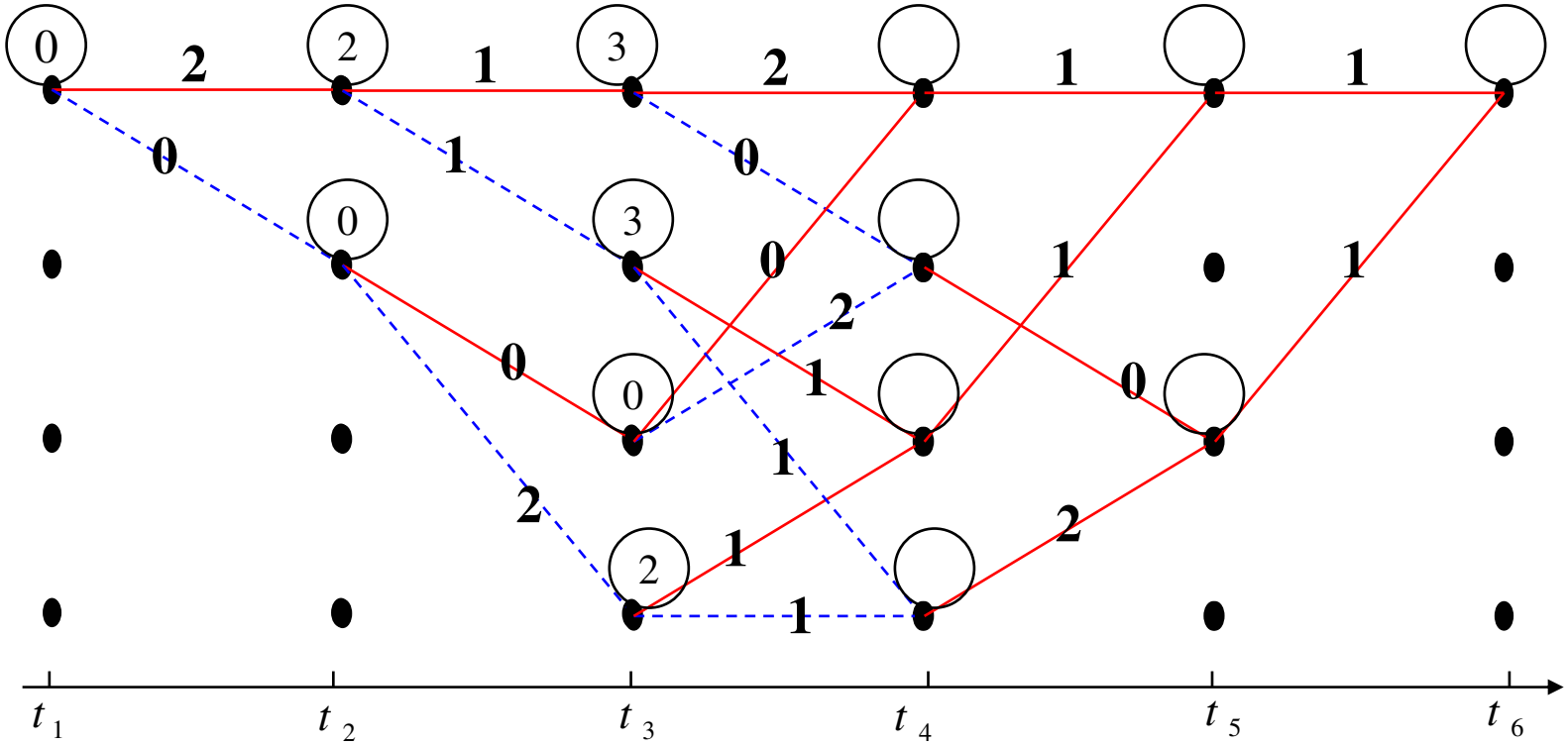
$$\mathbf{m} = (101)$$
$$\mathbf{U} = (11 \;\; 10 \;\; 00 \;\; 10 \;\; 11)$$
$$\mathbf{Z} = (11 \;\; 10 \;\; 11 \;\; 10 \;\; 01)$$



Example of Hard decision Viterbi decoding-cont'd

- i=3

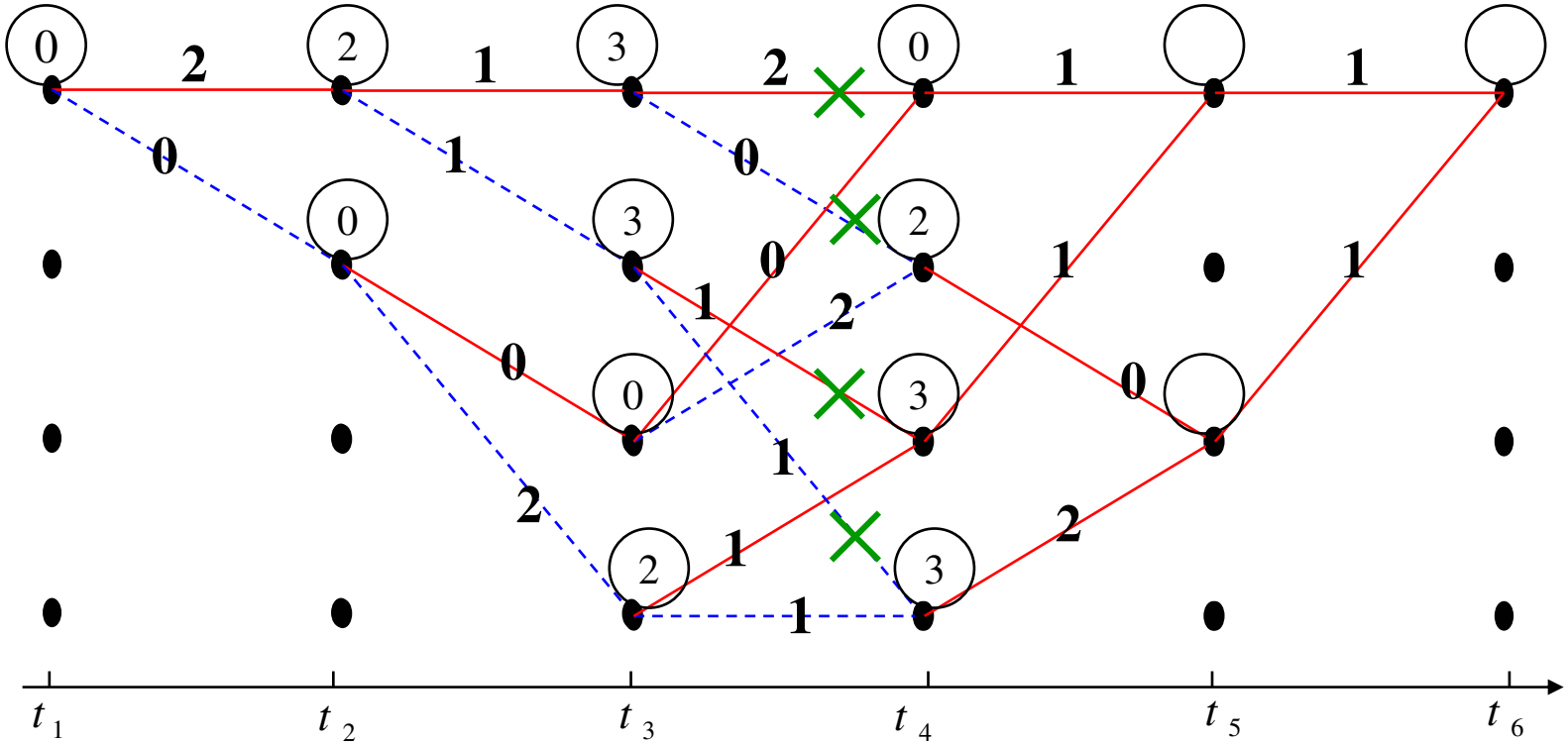
$$\mathbf{m} = (101)$$
$$\mathbf{U} = (11 \;\; 10 \;\; 00 \;\; 10 \;\; 11)$$
$$\mathbf{Z} = (11 \;\; 10 \;\; 11 \;\; 10 \;\; 01)$$



Example of Hard decision Viterbi decoding-cont'd

- i=4

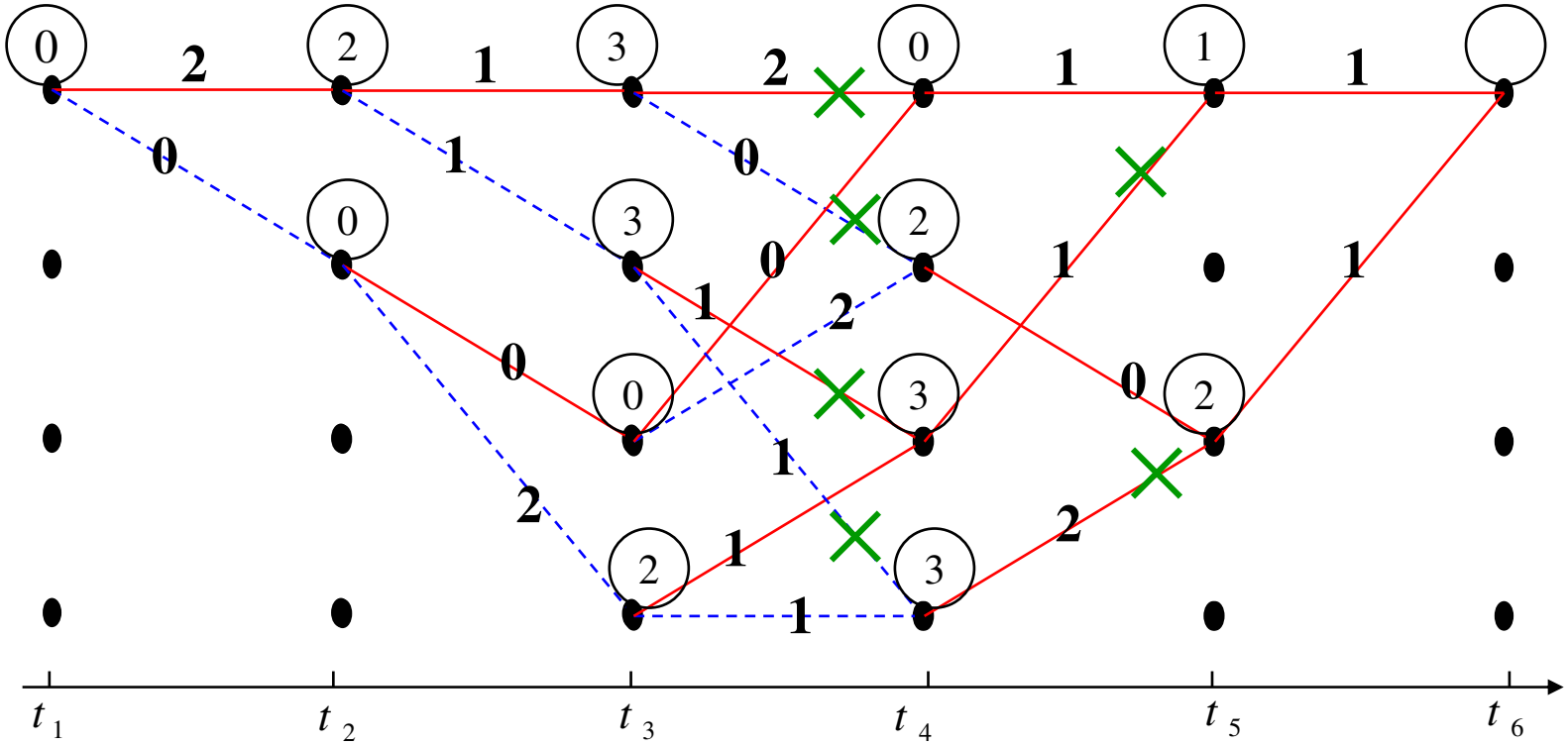
$$\mathbf{m} = (101)$$
$$\mathbf{U} = (11 \;\; 10 \;\; 00 \;\; 10 \;\; 11)$$
$$\mathbf{Z} = (11 \;\; 10 \;\; 11 \;\; 10 \;\; 01)$$



Example of Hard decision Viterbi decoding-cont'd

- i=5

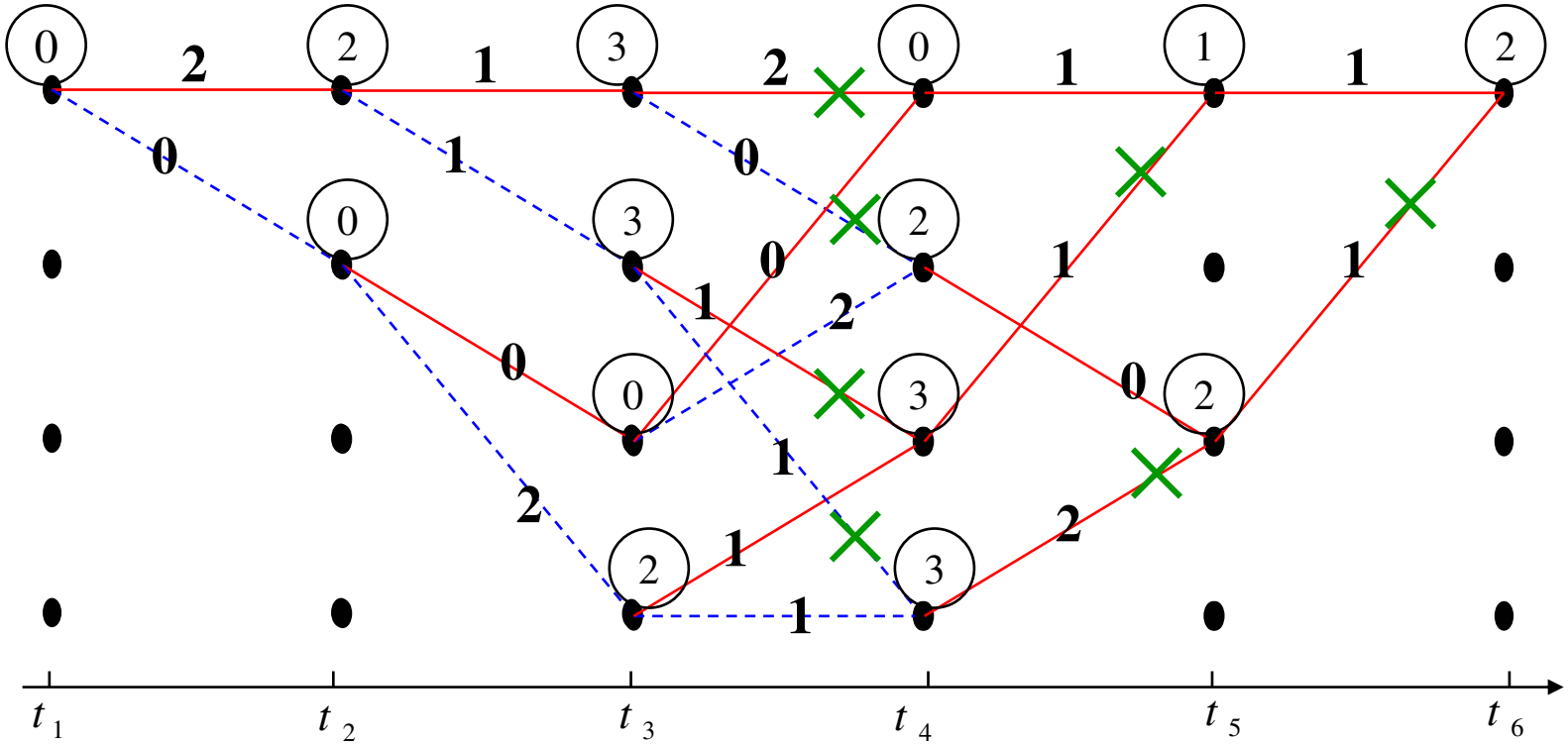
$$\mathbf{m} = (101)$$
$$\mathbf{U} = (11 \;\; 10 \;\; 00 \;\; 10 \;\; 11)$$
$$\mathbf{Z} = (11 \;\; 10 \;\; 11 \;\; 10 \;\; 01)$$



Example of Hard decision Viterbi decoding-cont'd

- i=6

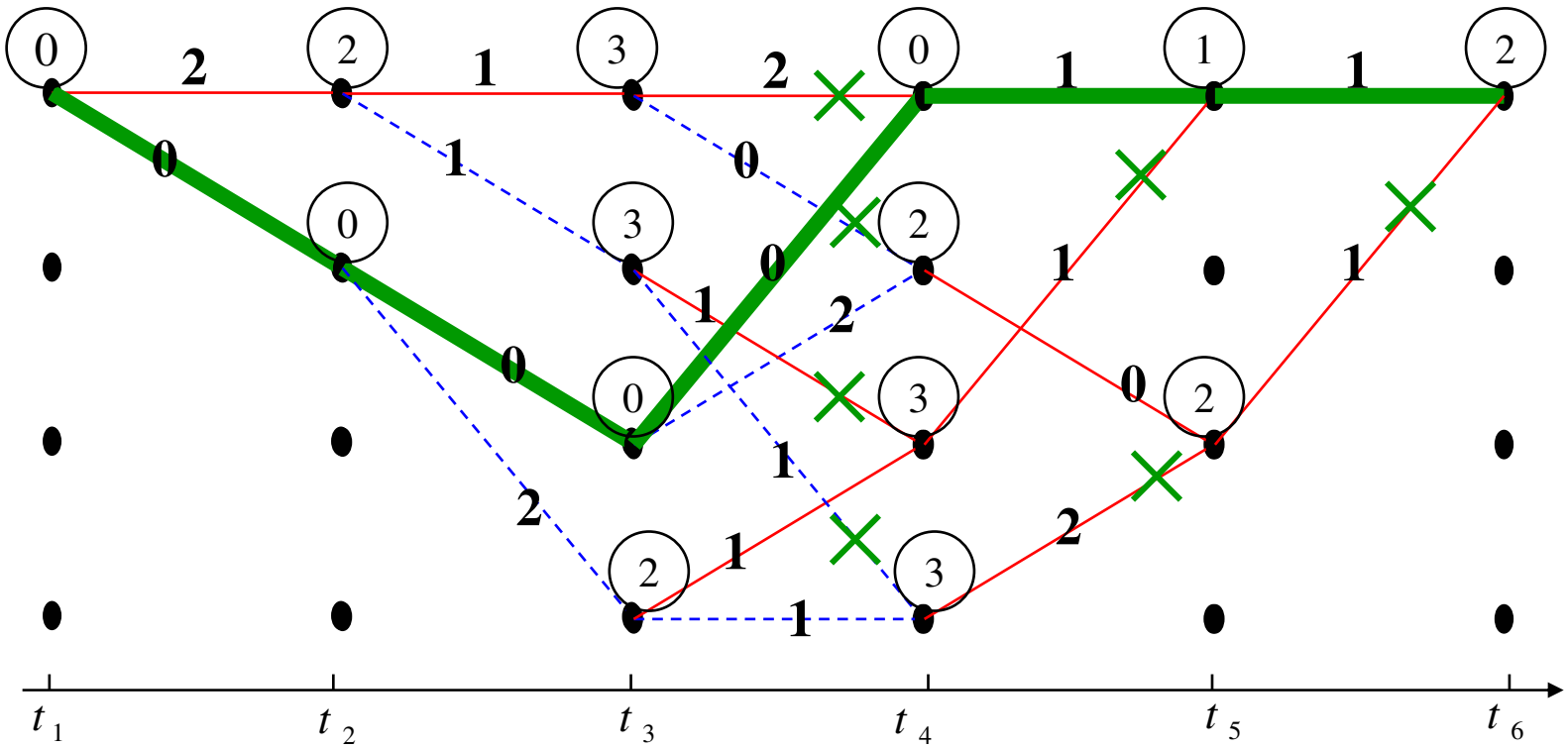
$$\mathbf{m} = (101)$$
$$\mathbf{U} = (11 \;\; 10 \;\; 00 \;\; 10 \;\; 11)$$
$$\mathbf{Z} = (11 \;\; 10 \;\; 11 \;\; 10 \;\; 01)$$



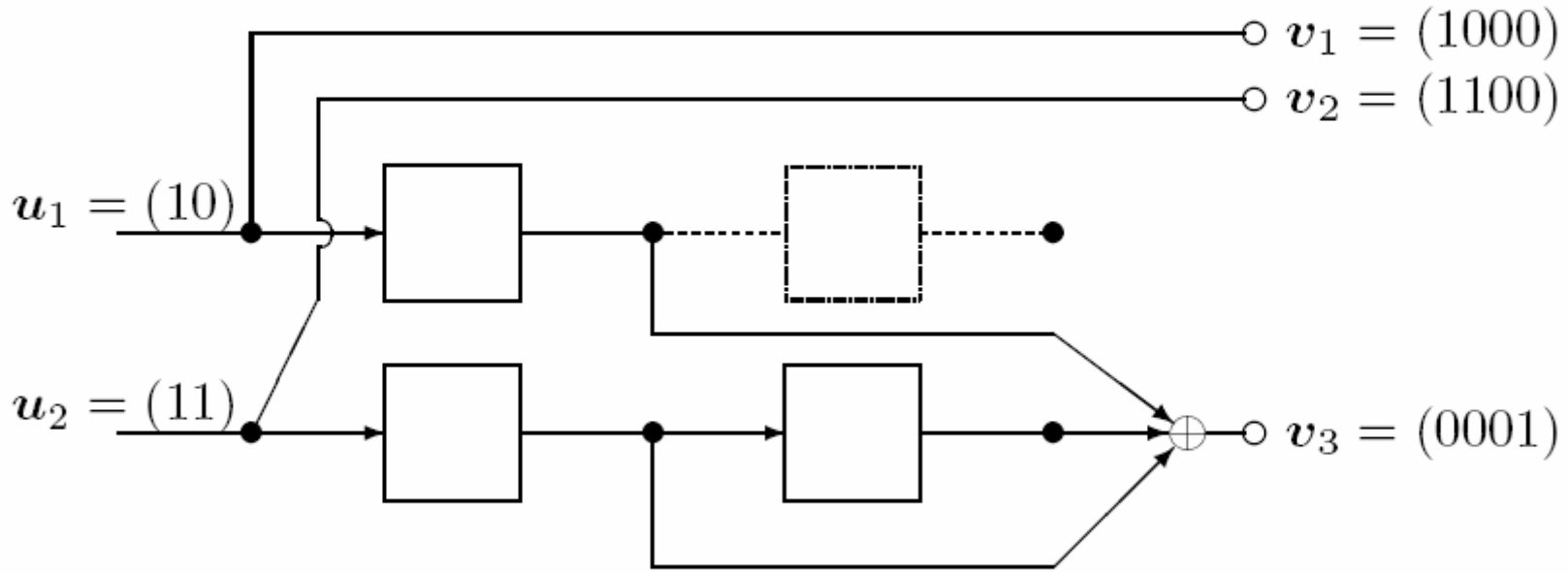
Example of Hard decision Viterbi decoding-cont'd

- Trace back and then:

$$\hat{\mathbf{m}} = (100)$$
$$\hat{\mathbf{U}} = (11 \;\; 10 \;\; 11 \;\; 00 \;\; 00)$$

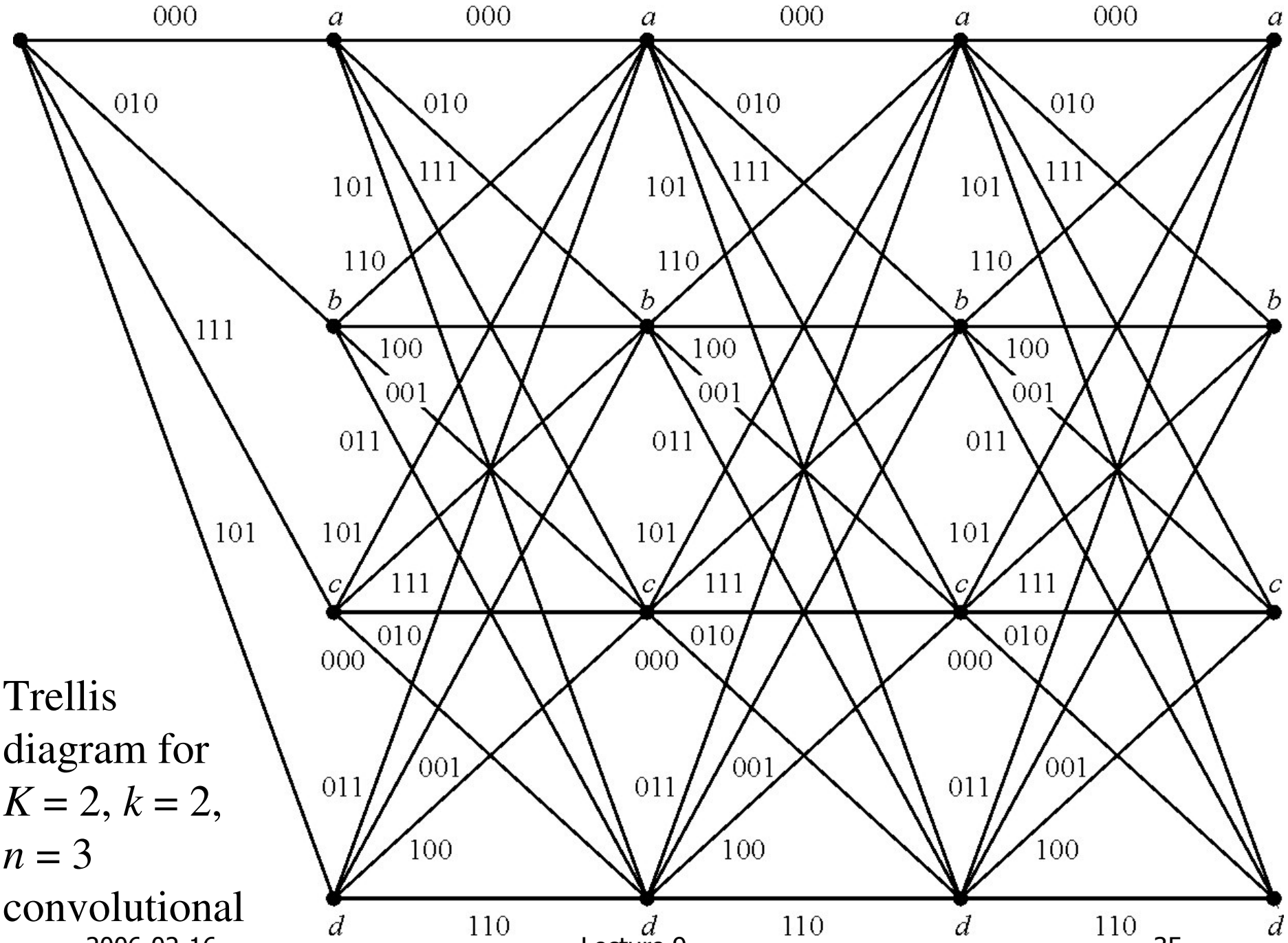


Encoder for the Binary $(3, 2, 2)$ Convolutional Code

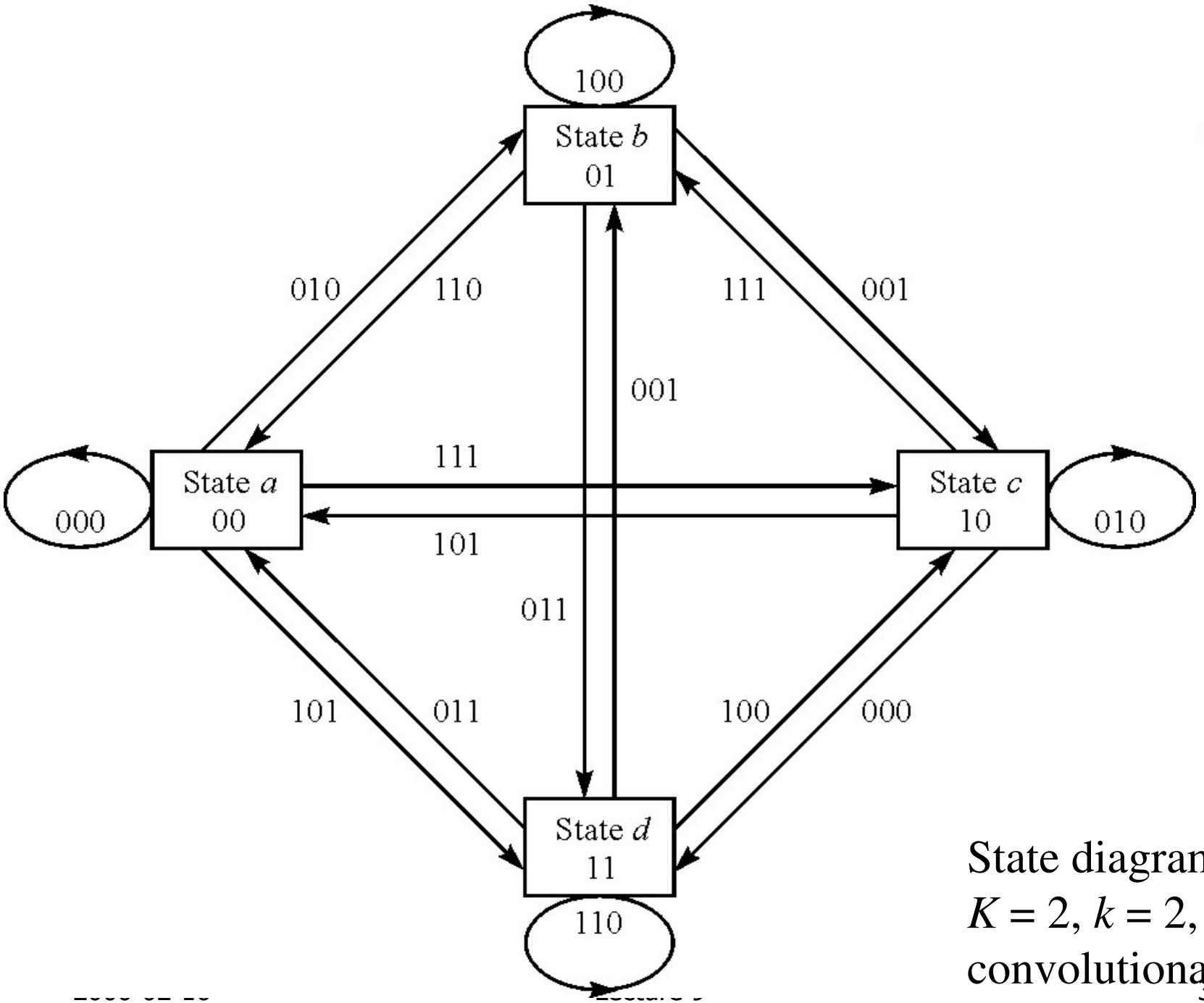


$\boldsymbol{u} = (11\ 01)$

$\boldsymbol{v} = (110\ 010\ 000\ 001)$



Trellis diagram for $K = 2$, $k = 2$, $n = 3$ convolutional code.

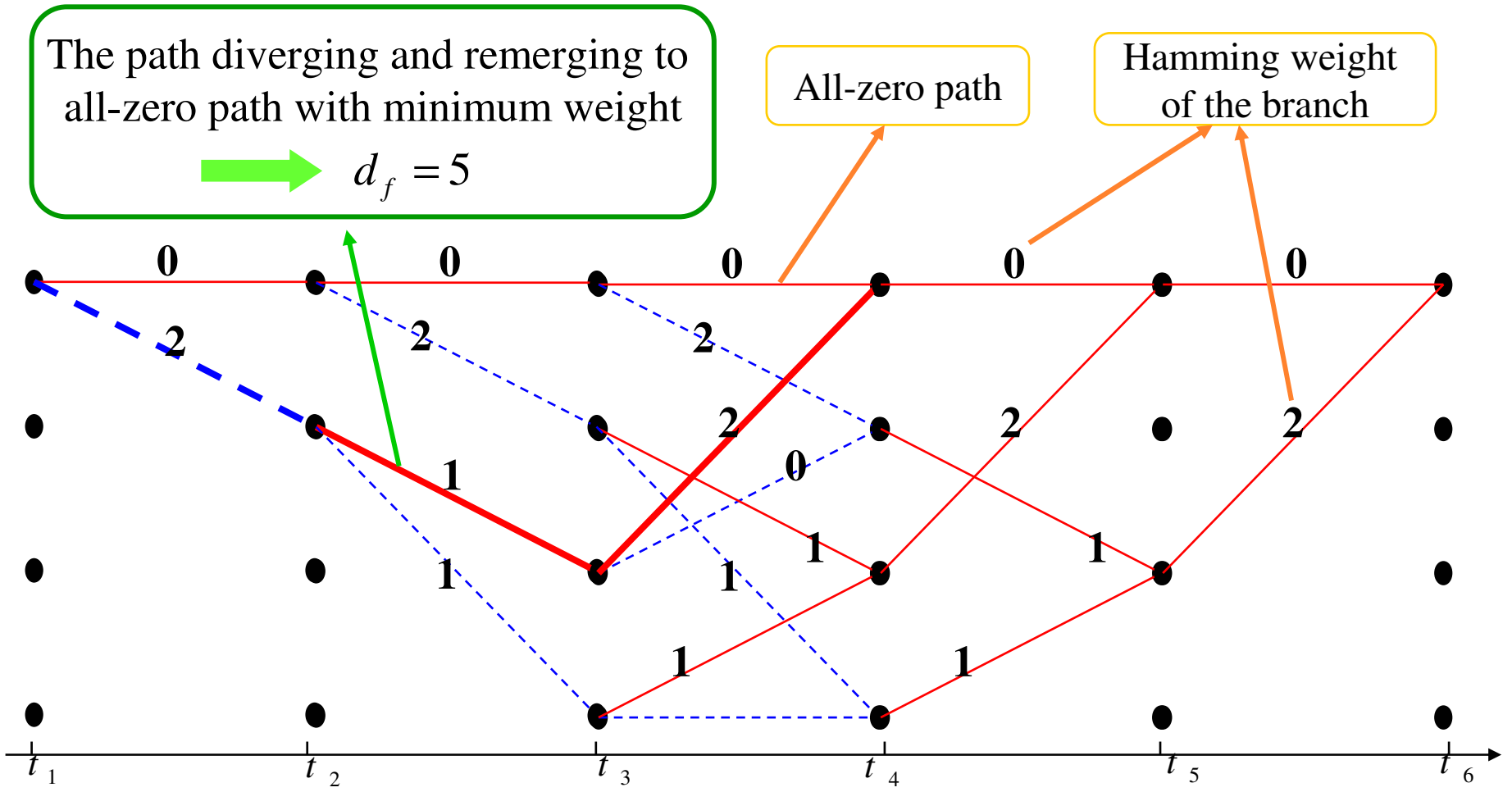


State diagram for $K = 2$, $k = 2$, $n = 3$ convolutional code.

Free distance of Convolutional codes

- Distance properties:
  - Since a Convolutional encoder generates codewords with various sizes (as opposite to the block codes), the following approach is used to find the minimum distance between all pairs of codewords:
    - Since the code is linear, the minimum distance of the code is the minimum distance between each of the codewords and the all-zero codeword.
    - This is the minimum distance in the set of all arbitrary long paths along the trellis that diverge and remerge to the all-zero path.
    - It is called the minimum free distance or the free distance of the code, denoted by d_{free} or d_f

Free distance ...



Transfer function of Convolutional codes

- Transfer function:
  - Transfer function of generating function is a tool which provides information about the weight distribution of the codewords.
    - The weight distribution specifies weights of different paths in the trellis (codewords) with their corresponding lengths and amount of information.

$$T(D,L,N)=\sum_{i=d_f}\sum_{j=K}\sum_{l=1} D^i L^j N^l$$

D, L, N : place holders

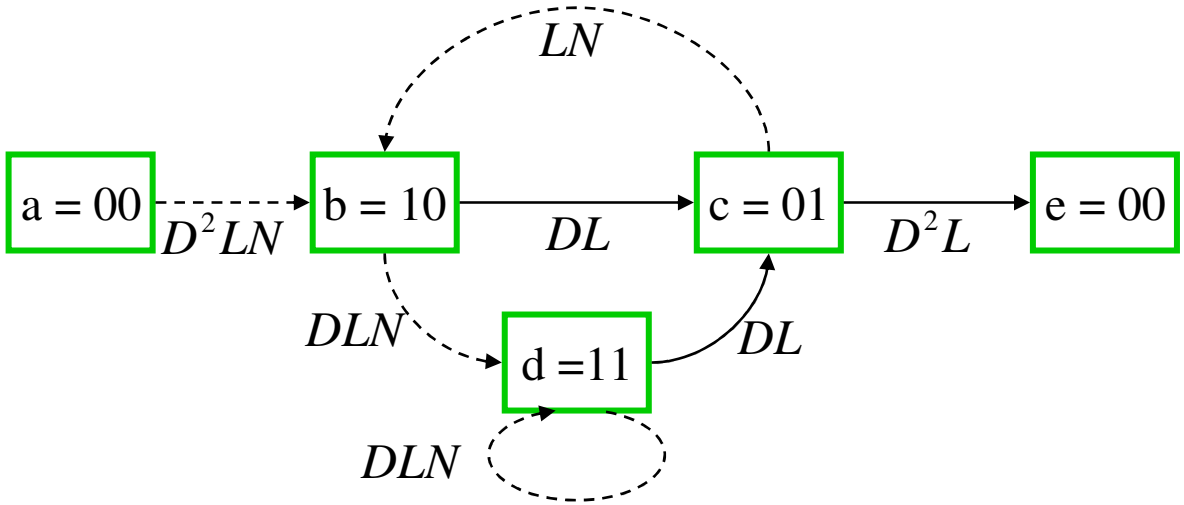
i : distance of the path from the all - zero path

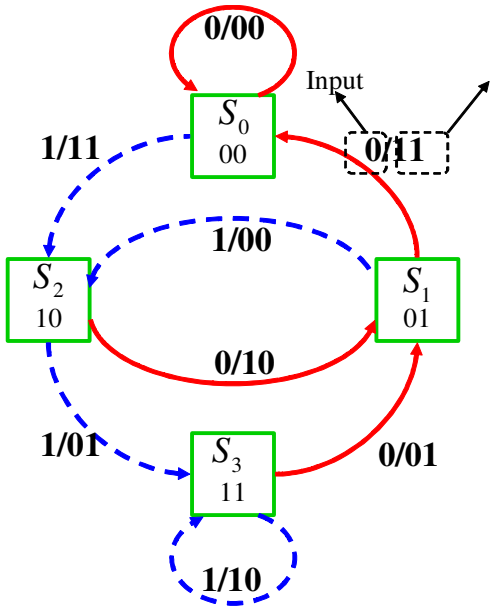
j : number of branches that the path takes until it remerges to the all - zero path

l : weight of the information bits corresponding to the path

Transfer function …

- Example of transfer function for the rate ½ Convolutonal code.
  1. Redraw the state diagram such that the zero state is split into two nodes, the starting and ending nodes.
  2. Label each branch by the corresponding $D^i L^j N^l$





Transfer function …

3. Write the state equations (X_a,…, X_e dummy variables)

$$\begin{cases} X_b = D^2LNX_a + LNX_c \\ X_c = DLX_b + DLX_d \\ X_d = DLNX_b + DLNX_d \\ X_e = D^2LX_c \end{cases}$$

4. Solve $T(D,L,N) = X_e / X_a$

$$T(D,L,N) = \frac{D^5L^3N}{1-DL(1+L)N} = D^5L^3N + D^6L^4N^2 + D^6L^5N^2 + \ldots.$$

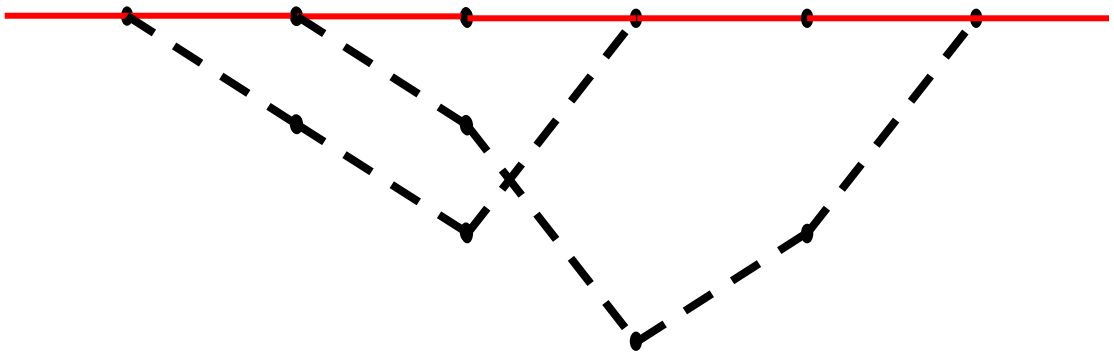
One path with weight 5, length 3 and data weight of 1

One path with weight 6, length 4 and data weight of 2

One path with weight 5, length 5 and data weight of 2

Performance bounds …

- Analysis is based on:
  - Assuming the all-zero codeword is transmitted
  - Evaluating the probability of an "**error event**" (usually using bounds such as union bound).
    - An "error event" occurs at a time instant in the trellis if a non-zero path leaves the all-zero path and remerges to it at a later time.



Performance bounds ...

- Bounds on bit error probability for memoryless channels:
  - Hard-decision decoding:

$$P_B \leq \left. \frac{dT(D,L,N)}{dN} \right|_{N=1,L=1,D=2\sqrt{p(1-p)}}$$

Performance bounds …

- Error correction capability of Convolutional codes, given by $t = \lfloor (d_f - 1)/2 \rfloor$, depends on
  - If the decoding is performed long enough (within 3 to 5 times of the constraint length)
  - How the errors are distributed (bursty or random)
- For a given code rate, increasing the constraint length, usually increases the free distance.
- For a given constraint length, decreasing the coding rate, usually increases the free distance.
- The coding gain is upper bounded

$$\text{coding gain} \leq 10 \log_{10}(R_c d_f)$$

Performance bounds ...

- Basic coding gain (dB) for soft-decision Viterbi decoding

Uncoded	Code rate		1/3		1/2	
E_b / N_0 (dB)	P_B	K	7	8	6	7
6.8	10^{-3}		4.2	4.4	3.5	3.8
9.6	10^{-5}		5.7	5.9	4.6	5.1
11.3	10^{-7}		6.2	6.5	5.3	5.8
Upper bound			7.0	7.3	6.0	7.0

Interleaving

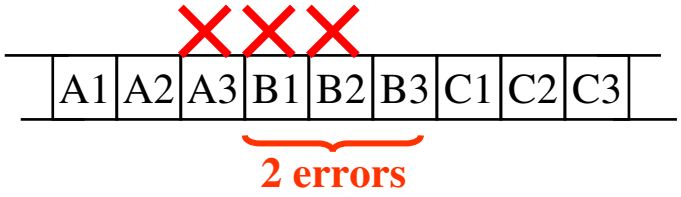
- Convolutional codes are suitable for memoryless channels with random error events.
- Some errors have bursty nature:
  - Statistical dependence among successive error events (time-correlation) due to the channel memory.
    - Like errors in multipath fading channels in wireless communications, errors due to the switching noise, …
- "Interleaving" makes the channel looks like as a memoryless channel at the decoder.

Interleaving …

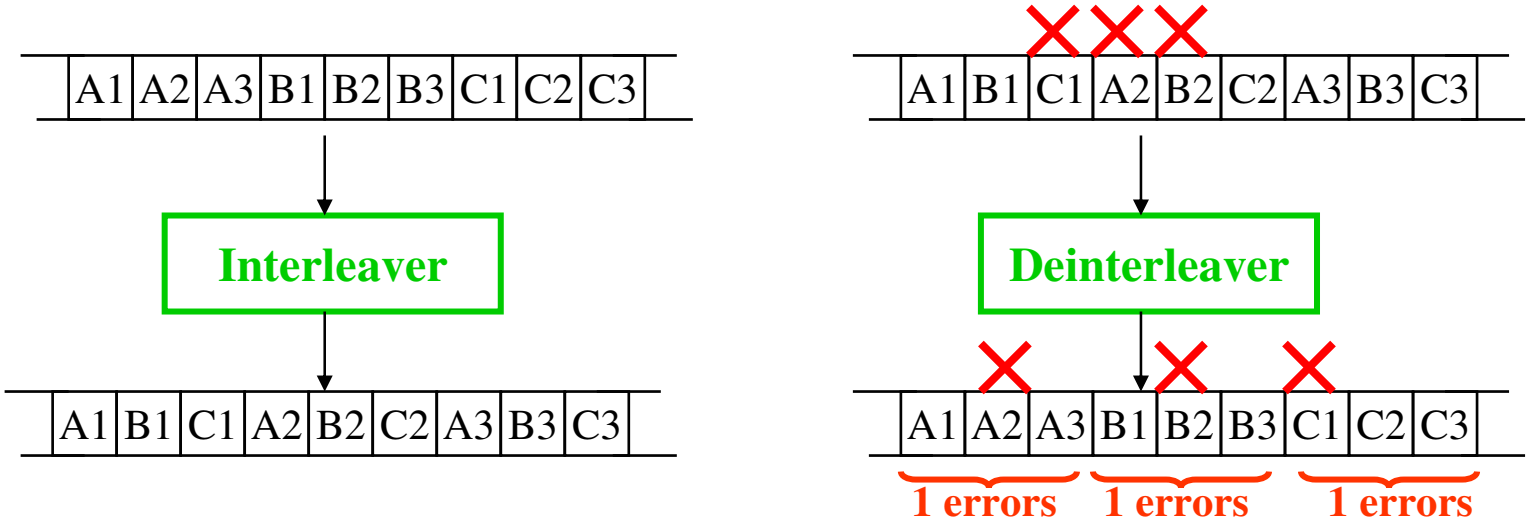
- Interleaving is done by spreading the coded symbols in time (interleaving) before transmission.
- The reverse in done at the receiver by deinterleaving the received sequence.
- "Interleaving" makes bursty errors look like random. Hence, Conv. codes can be used.
- Types of interleaving:
  - Block interleaving
  - Convolutional or cross interleaving

Interleaving …

- Consider a code with t=1 and 3 coded bits.
- A burst error of length 3 can not be corrected.



- Let us use a block interleaver 3X3



Concatenated codes

- A concatenated code uses two levels on coding, an inner code and an outer code (higher rate).
  - Popular concatenated codes: Convolutional codes with Viterbi decoding as the inner code and Reed-Solomon codes as the outer code
- The purpose is to reduce the overall complexity, yet achieving the required error performance.