CS311: Data Communication

Channel Coding - II

by

Dr. Manas Khatua
Assistant Professor
Dept. of CSE
IIT Jodhpur
E-mail: manaskhatua@iitj.ac.in
Web: http://home.iitj.ac.in/~manaskhatua
http://manaskhatua.github.io/

Coding Theory

- Coding theory is the study of the properties of codes and their respective fitness for specific applications.

- Codes are used for
  - Data compression
  - Error-detection and error-correction
  - Networking
  - Cryptography

- the purpose of coding is of designing efficient and reliable data transmission methods.

- There are four types of coding:
  - Source coding
  - **Channel coding**
  - Line coding
  - Cryptographic coding

Cont…

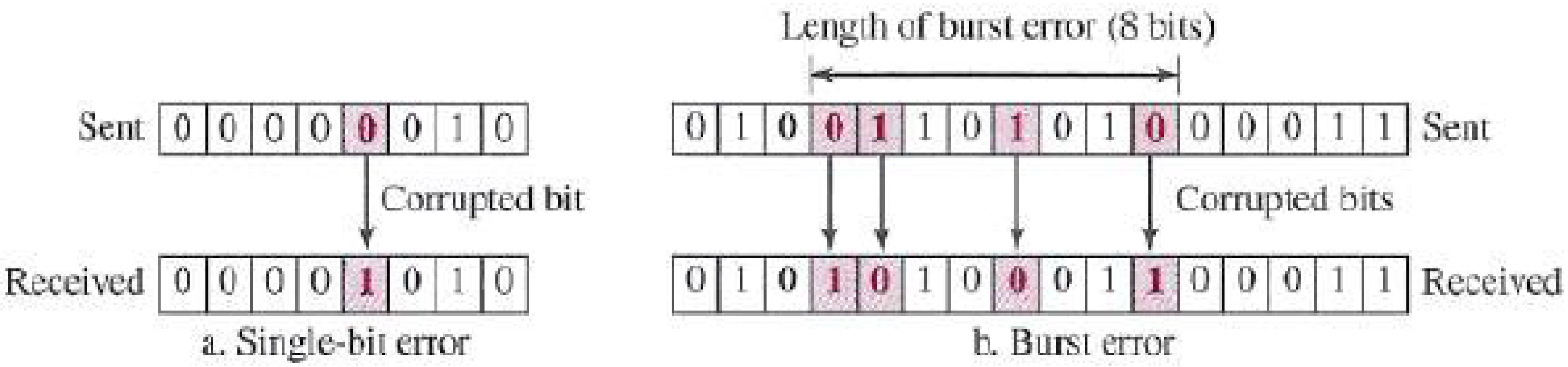
- Source coding
  - The aim of source coding is to take the source data and make it smaller in size.
  - e.g., Zip coding
- Channel coding
  - The purpose is to find codes which transmit quickly, contain many valid code words and can correct or at least detect many errors.
  - e.g., Reed-Solomon code, Turbo code, LDPC code, Cyclic code, Convolution code
- Line coding
  - is called digital baseband modulation technique
  - e.g., unipolar, polar, bipolar, and Manchester encoding
- Cryptographic coding
  - is the practice and study of techniques for secure communication in the presence of third parties
  - e.g., RSA Algorithm

Error Detection and Correction

- Objective:
  - System must guarantee that the data received are identical to the data transmitted

- Methods:
  1. If a frame is corrupted between the two nodes, it needs to be corrected
  2. Drop the frame and let the upper layer (e.g. Transport) protocol to handle it by retransmission

Types of Error



- Single bit error
- Burst error / multibit error

- Reason: noise in the channel

Detection and Correction

- Central idea: redundancy
  - put some extra bit with our data
  - Achieved by channel coding scheme
    - Linear block coding : the sum of any two codewords is also a code word
      - Cyclic codes (e.g., Hamming codes)
      - Repetition codes
      - Parity codes
      - Polynomial codes (e.g., BCH codes)
      - Reed–Solomon codes
      - Algebraic geometric codes
      - Reed–Muller codes
      - Perfect codes
    - **Convolution coding**: make every codeword symbol be the weighted sum of the various input message symbols

- Error Detection : looking to see if any error has occurred

- Error Correction: trying to recover the corruption
  - Need to know exact number of bits that are corrupted
  - Needs the position of those bits

Convolution Code

- It is a type of error-correcting code
- It generates parity symbols via the sliding application of a boolean polynomial function to a data stream
- The sliding application represents the 'convolution' of the encoder over the data
- Advantages over block code:
  - maximum-likelihood soft-decision decoded with reasonable complexity using time-invariant trellis
  - whereas, classic block codes are generally represented by a time-variant trellis and therefore are typically hard-decision decoded

Convolution Code

k number of input message symbols (as before)

n number of output codeword symbols (as before)

r base code rate = k/n

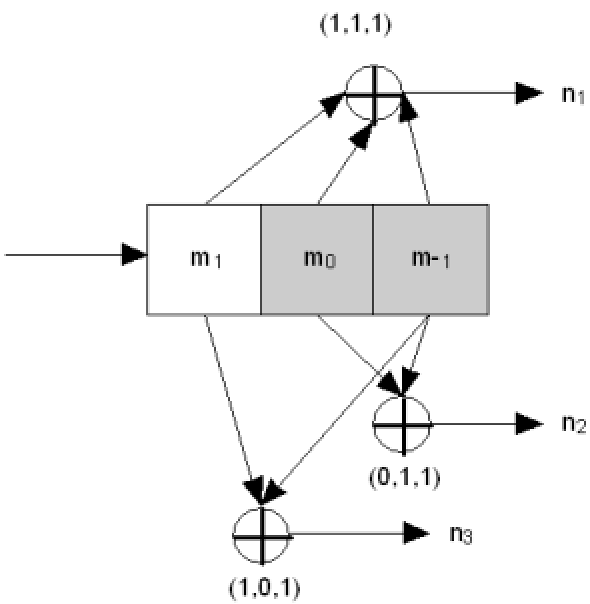
m memory, i.e., number of input symbol is stored

K depth of the encoder, i.e., max number of input symbols used by encoder to compute each output symbol.

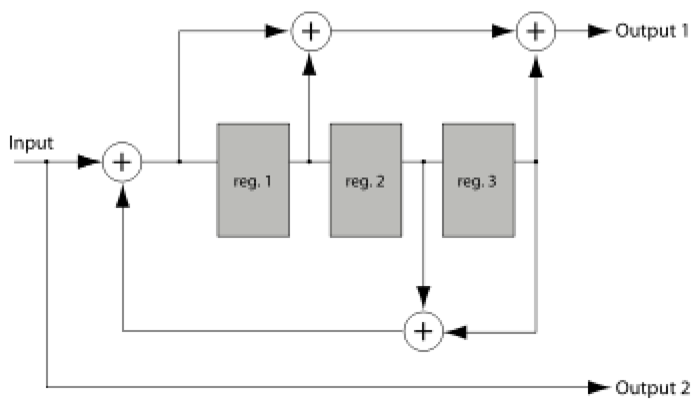
- the output is a function of the current input as well as the previous K-1 inputs.
- decoding time exponentially dependent on K.
- Convolutional codes are often characterized by [k,n,K]

Convolution Code

- Types:
  - Classic convolutional code
  - Recursive convolutional code (e.g. Turbo Code)



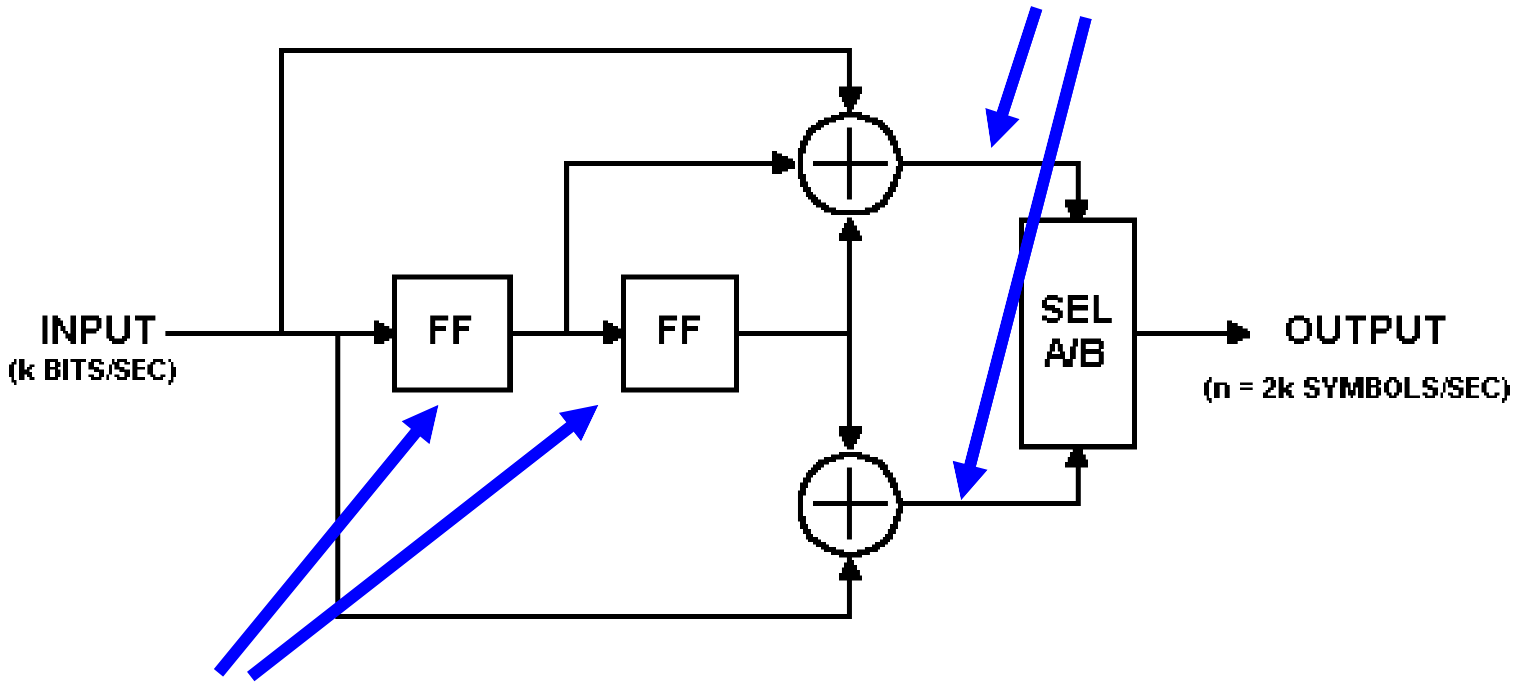
Rate 1/3 non-recursive, non-systematic convolutional encoder with depth 3



Rate 1/2 8-state recursive systematic convolutional encoder.

Convolution Encoder

output upper input followed by lower input

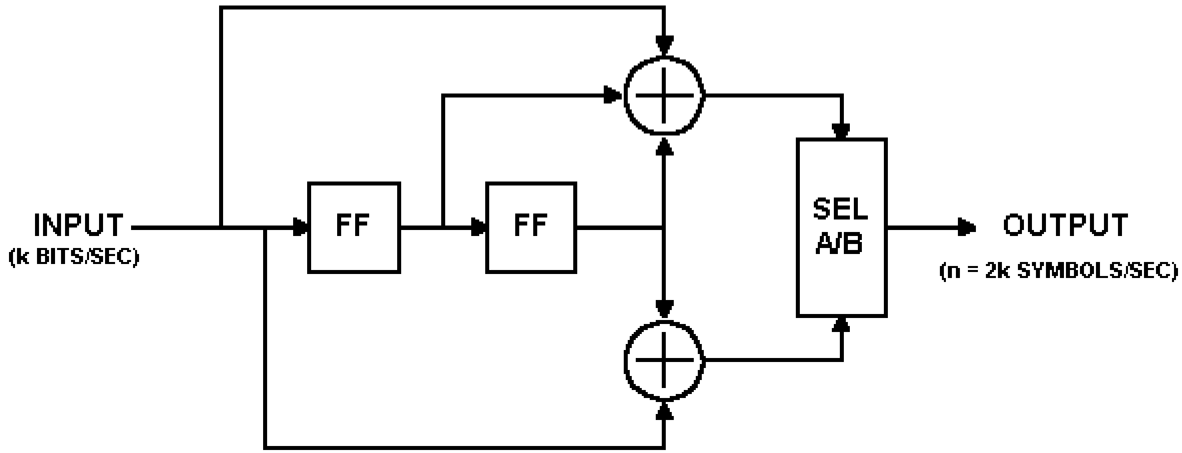


flip flop
(stores one bit)

Input symbol k = 15
Output symbol n = 30
Code rate r = 1/2

Two-stage register m = 2
Constraint length or depth K = m+1 = 3

Example



Both flip flops set to 0 initially.

Input : **0** 1 **0** 1 **1** 1 **0** 0 **1** 0 **1** 0 **0** 0 **1**

Output : **00** 11 **10** 00 **01** 10 **01** 11 **11** 10 **00** 10 **11** 00 **11**

Flush encoder by clocking m = 2 times with 0 inputs.

Input : **1** 0 **1** 1 **0** 1 **1** 1

Output : **11** 10 **00** 01 **01** 00 **01** 10

State Transition & Output Tables

Flip-Flop State transition table

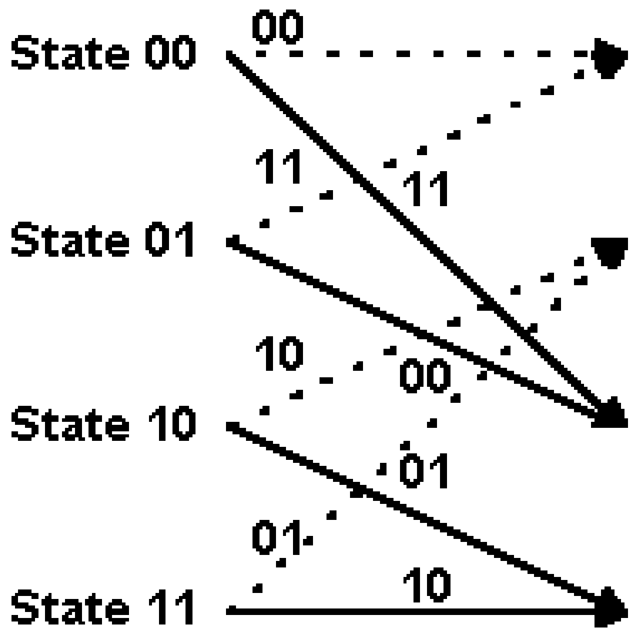
	Next State, if	
Current State	**Input = 0:**	**Input = 1:**
00	00	10
01	00	10
10	01	11
11	01	11

Output table

	Output Symbols, if	
Current State	**Input = 0:**	**Input = 1:**
00	00	11
01	11	00
10	10	01
11	01	10

2^m rows

State Transitions Pattern

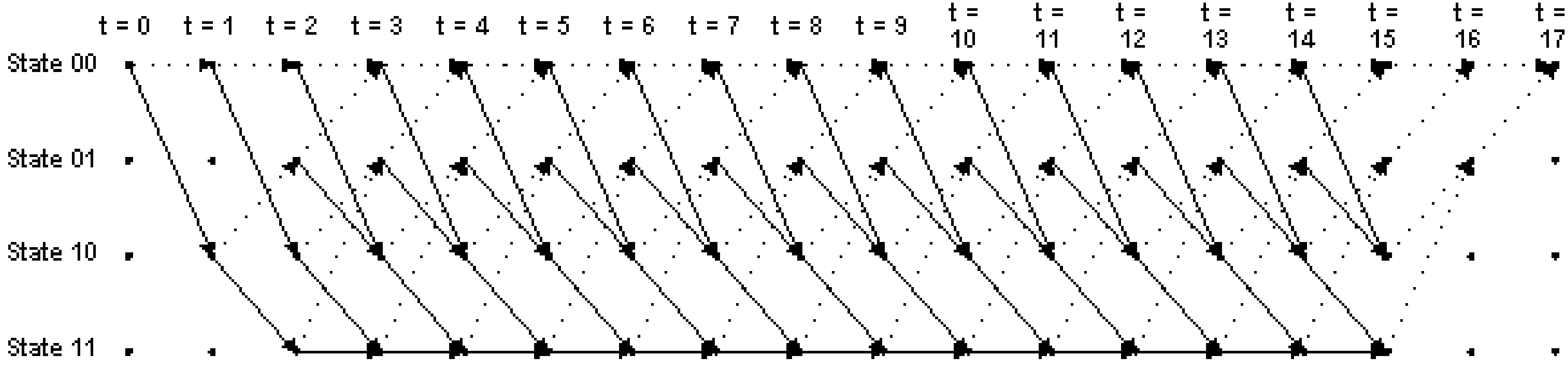


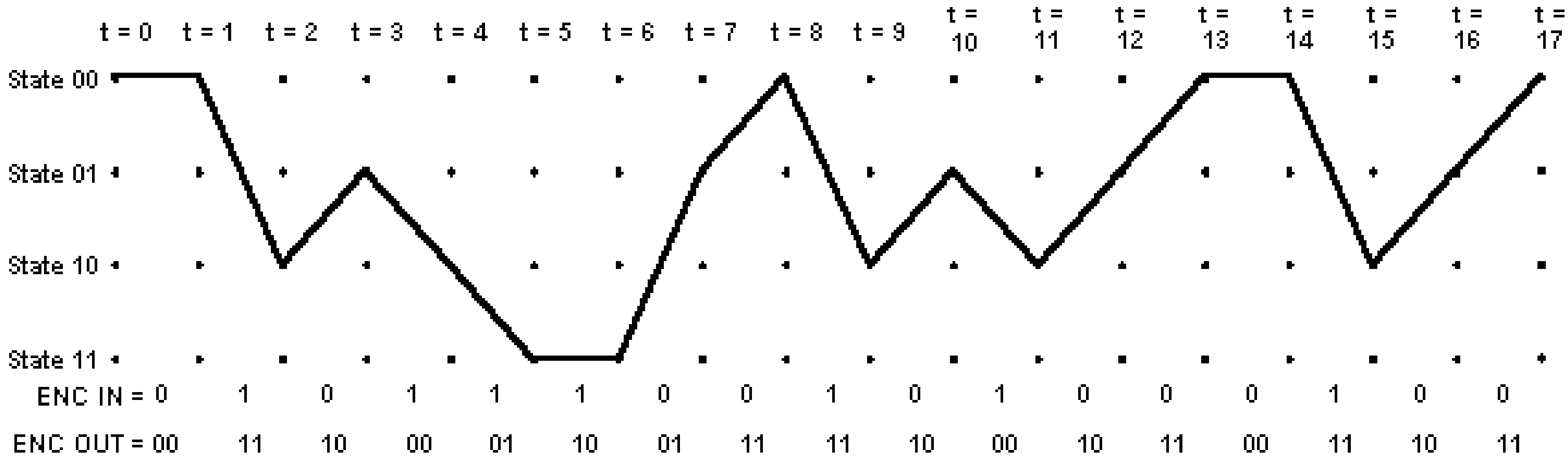
input symbol is 1

input symbol is 0

arcs labeled with output symbols

Trellis

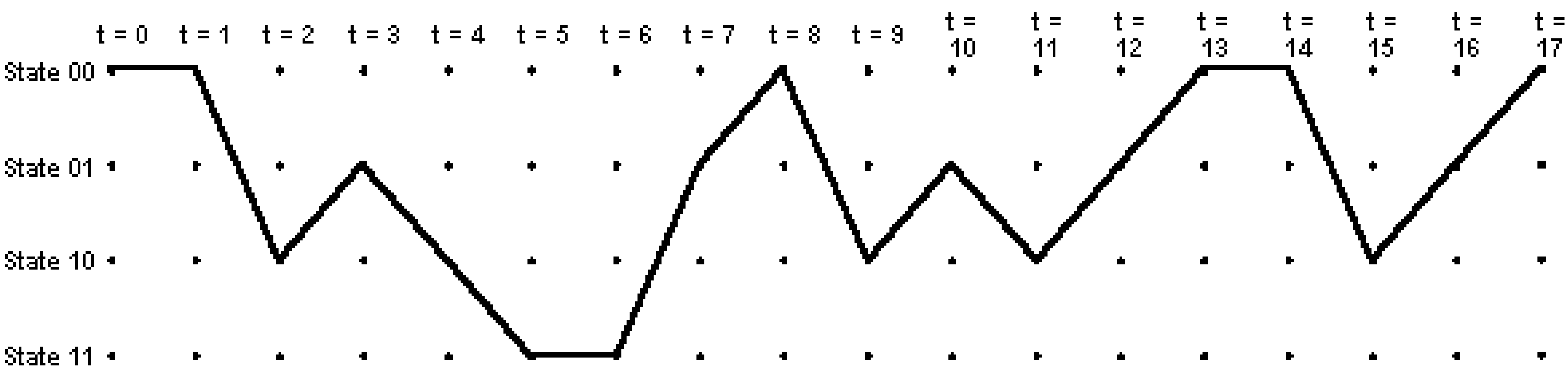

t = 0 t = 1 t = 2 t = 3 t = 4 t = 5 t = 6 t = 7 t = 8 t = 9 t = 10 t = 11 t = 12 t = 13 t = 14 t = 15 t = 16 t = 17
State 00
State 01
State 10
State 11


t = 0 t = 1 t = 2 t = 3 t = 4 t = 5 t = 6 t = 7 t = 8 t = 9 t = 10 t = 11 t = 12 t = 13 t = 14 t = 15 t = 16 t = 17
State 00
State 01
State 10
State 11
ENC IN = 0 1 0 1 1 1 0 0 1 0 1 0 0 0 1 0 0
ENC OUT = 00 11 10 00 01 10 01 11 11 10 00 10 11 00 11 10 11

Oh no! Errors in received bits!

	Next State, if	
Current State	**Input = 0:**	**Input = 1:**
00	00	10
01	00	10
10	01	11
11	01	11

	Output Symbols, if	
Current State	**Input = 0:**	**Input = 1:**
00	00	11
01	11	00
10	10	01
11	01	10



ENC IN = 0	1	0	1	1	1	0	0	1	0	1	0	0	0	1	0	0
ENC OUT = 00	11	10	00	01	10	01	11	11	10	00	10	11	00	11	10	11
RECEIVED =00	11	11	00	01	10	01	11	11	10	00	00	11	00	11	10	11
ERRORS =		X									X					

Decoding

- A message **m** is encoded into the code sequence **c**.
- Each code sequence represents a path in the trellis diagram.

- <u>Minimum Distance Decoding</u>
  - Upon receiving the received sequence **r**,
    search for the path that is closest (in Hamming distance) to **r** .
  - Hamming distance between the received sequence and the possible output sequences

- Solution: Viterbi Algorithm

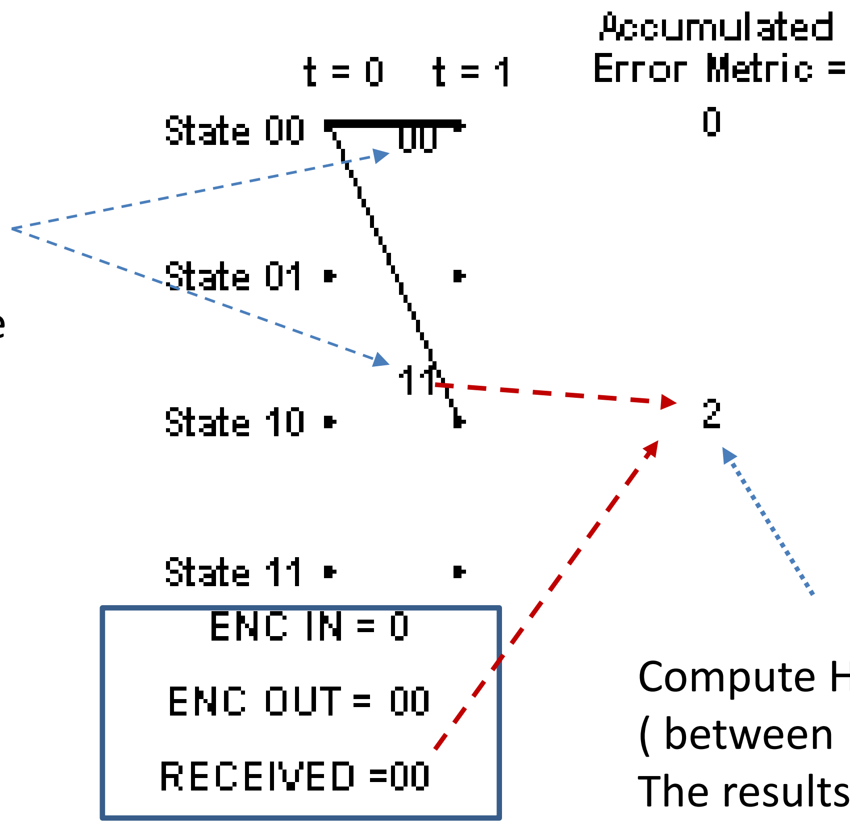
Reference:

A Tutorial on Convolutional Coding with Viterbi Decoding, by Chip Fleming, http://home.netcom.com/~chip.f/viterbi/tutorial.html

Viterbi Decoding - Accumulated Error Metric

- Suppose we receive the above encoded message with a couple of bit errors:
- At t = 1, we received 00_2.
- The only possible channel symbol pairs we could have received are 00_2 and 11_2.

Possible Output Sequence



Trying to find the input sequence whose corresponding output matches the received output as closely as possible.

Compute Hamming distance
(between received bits and possible output)
The results can only be zero, one, or two

Cont…

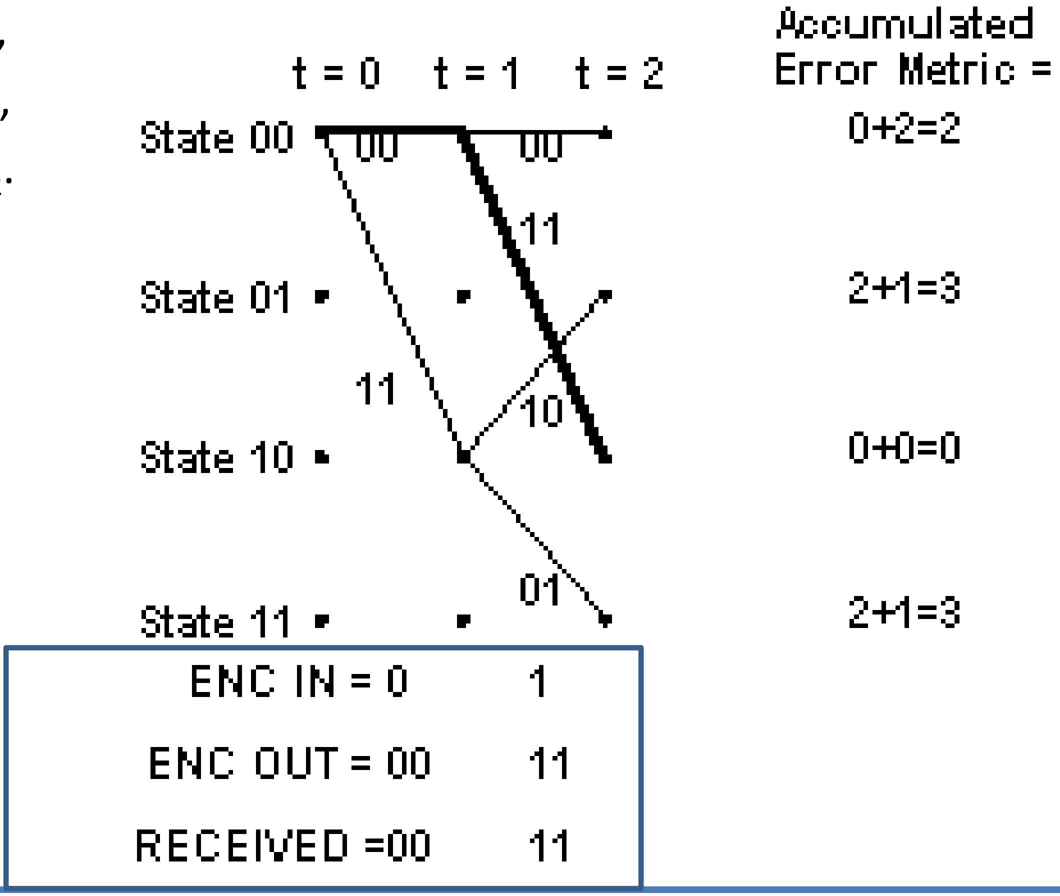
- At t = 2, we received 11_2 channel symbol pair.
- The possible channel symbol pairs we could have received in going from t = 1 to t = 2 are:

00_2 going from State 00_2 to State 00_2,

11_2 going from State 00_2 to State 10_2,

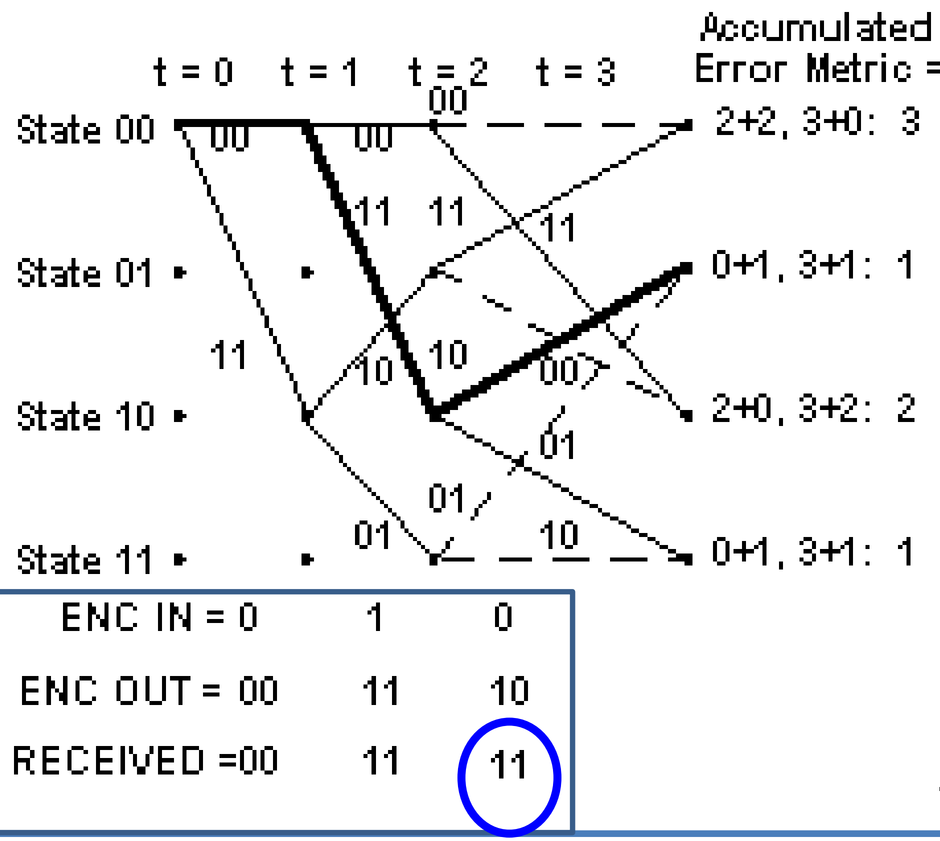
10_2 going from State 10_2 to State 01_2,

01_2 going from State 10_2 to State 11_2.



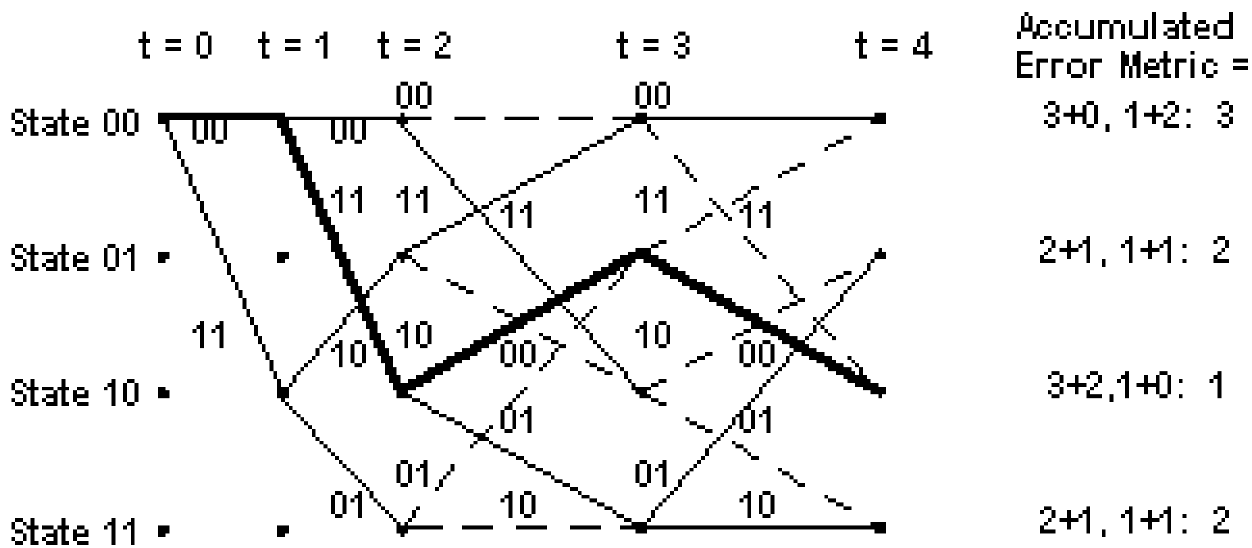
Cont…

- At t = 3, we received 11_2 channel symbol pair.
- If the two accumulated error metrics are equal, some people use a fair coin toss to choose the surviving predecessor state.



Cont…

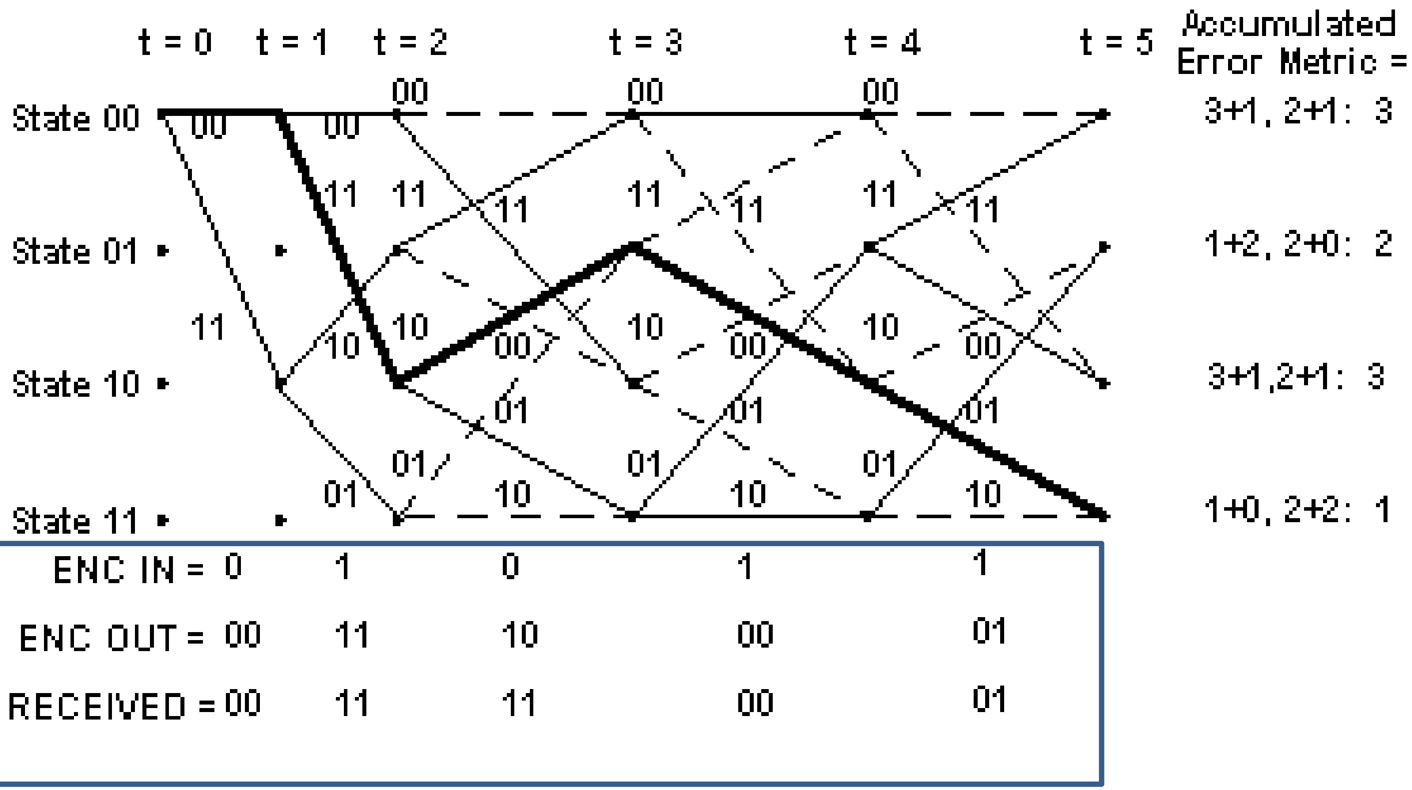
- At t = 4, we received 00_2 channel symbol pair.



ENC IN =	0	1	0	1
ENC OUT =	00	11	10	00
RECEIVED =	00	11	11	00

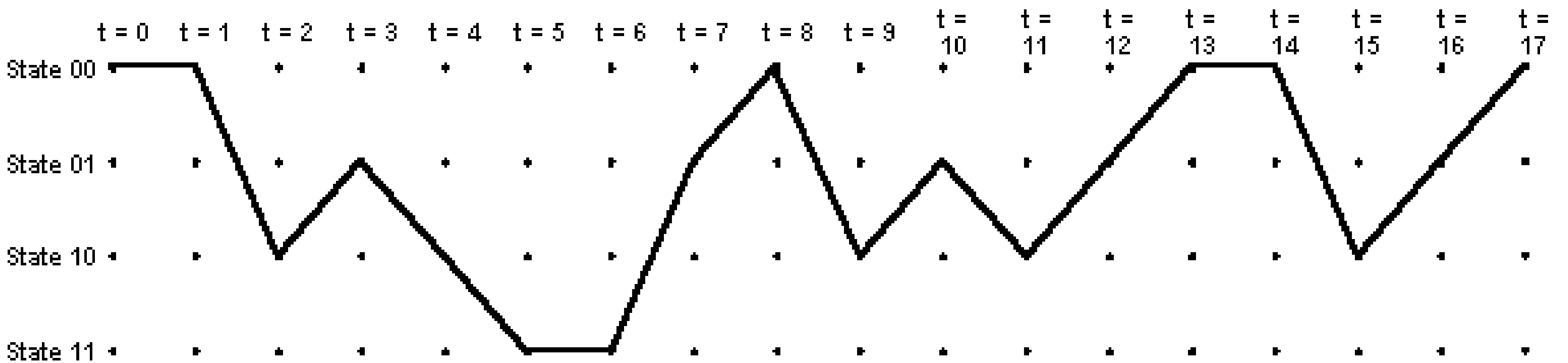
Cont…

- At t = 5, we received 01_2 channel symbol pair.



Cont…

- At t = 17, we received 11_2 channel symbol pair.



ENC IN = 0	1	0	1	1	1	0	0	1	0	1	0	0	0	1	0	0
ENC OUT = 00	11	10	00	01	10	01	11	11	10	00	10	11	00	11	10	11
RECEIVED =00	11	11	00	01	10	01	11	11	10	00	00	11	00	11	10	11
ERRORS =		X									X					

Accumulated Error Metric over Time

t =	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
State 00_2		0	2	3	3	3	3	4	1	3	4	3	3	2	2	4	5	2
State 01_2			3	1	2	2	3	1	4	4	1	4	2	3	4	4	2	
State 10_2		2	0	2	1	3	3	4	3	1	4	1	4	3	3	2		
State 11_2			3	1	2	1	1	3	4	4	3	4	2	3	4	4		

Note: Last two inputs known to be zero.

Consider the lowest value in each slot.

Surviving Predecessor States

t =	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
State 00_2	00	00	00	01	00	01	01	00	01	00	00	01	00	01	00	00	00	01
State 01_2	00	00	10	10	11	11	10	11	11	10	10	11	10	11	10	10	10	00
State 10_2	00	00	00	00	01	01	01	00	01	00	00	01	01	00	01	00	00	00
State 11_2	00	00	10	10	11	10	11	10	11	10	10	11	10	11	10	10	00	00

00 00 10 01 10 11 11 01 00 10 01 10 01 00 00 10 01

States Selected when Tracing Back

t =	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
	00	00	10	01	10	11	11	01	00	10	01	10	01	00	00	10	01	00

Thanks!

Figure and slide materials are taken from the following sources:

1. W. Stallings, (2010), Data and Computer Communications
2. NPTL lecture on Data Communication, by Prof. A. K. Pal, IIT Kharagpur
3. B. A. Forouzan, (2013), Data Communication and Networking