Digital Communication Systems ECS 452

Asst. Prof. Dr. Prapun Suksompong

prapun@siit.tu.ac.th

5.2 Binary Convolutional Codes

Binary Convolutional Codes

- Introduced by Elias in 1955
  - There, it is referred to as convolutional parity-check symbols codes.
  - Peter Elias received
    - Claude E. Shannon Award in 1977
    - IEEE Richard W. Hamming Medal in 2002
      - for "fundamental and pioneering contributions to information theory and its applications
- The encoder **has memory**.
  - In other words, the encoder is a **sequential circuit** or a **finite-state machine**.
    - Easily implemented by shift register(s).
    - The **state** of the encoder is defined as the contents of its memory.

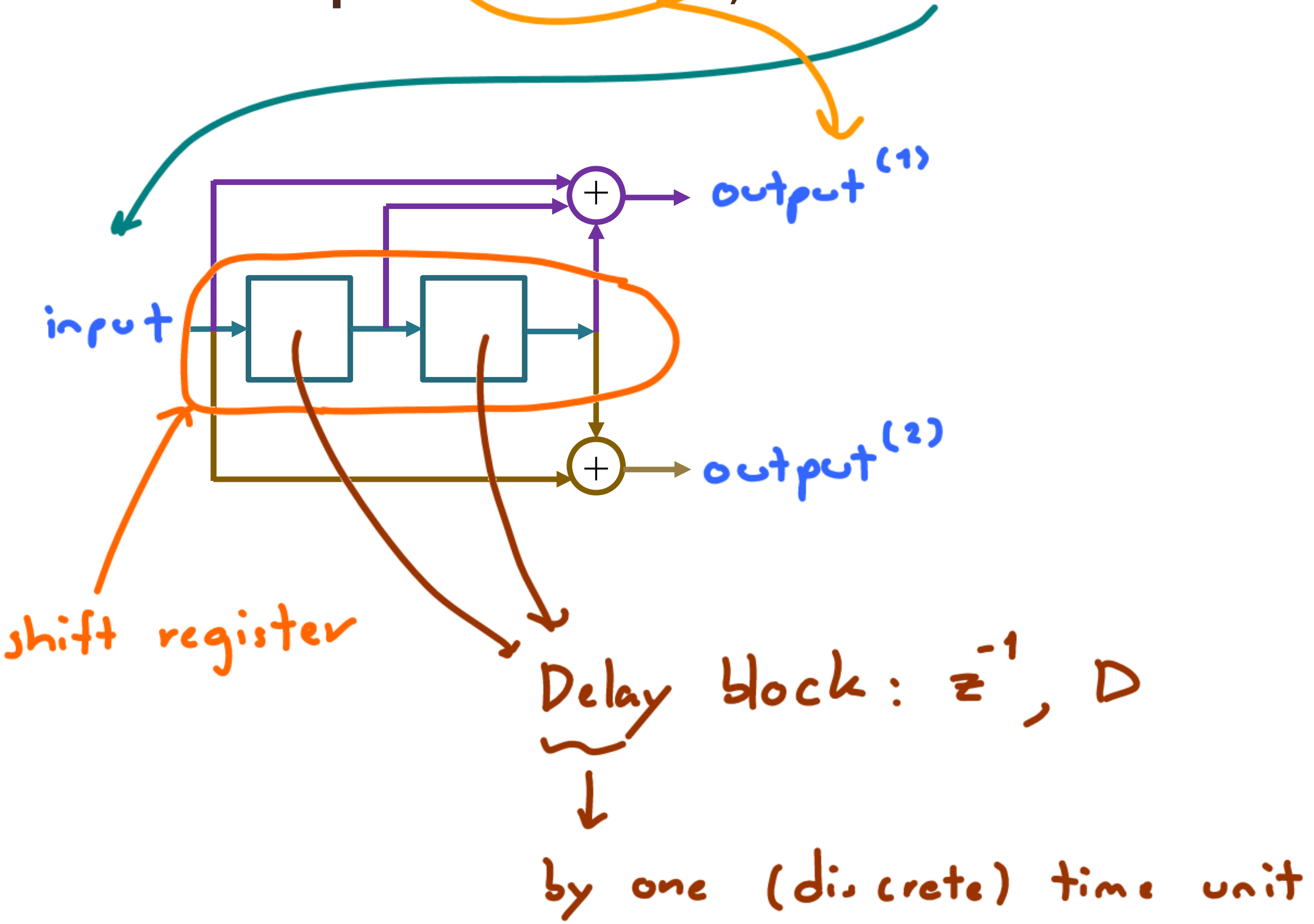
Binary Convolutional Codes

- The encoding is done on a **continuous** running basis rather than by blocks of *k* data digits.
  - So, we use the terms **bit streams** or **sequences** for the input and output of the encoder.
  - In theory, these sequences have infinite duration.
  - In practice, the state of the convolutional code is periodically forced to a known state and therefore code sequences are produced in a block-wise manner.

Binary Convolutional Codes

- In general, a **rate-$\frac{k}{n}$ convolutional encoder** has
  - k shift registers, one per input information bit, and
  - n output coded bits that are given by linear combinations (over the binary field, of the contents of the registers and the input information bits.
- k and n are usually small.
- For simplicity of exposition, and for practical purposes, only **rate-$\frac{1}{n}$** binary convolutional codes are considered here.
  - $k = 1$.
  - These are the most widely used binary codes.

Example: $n = 2$, $k = 1$



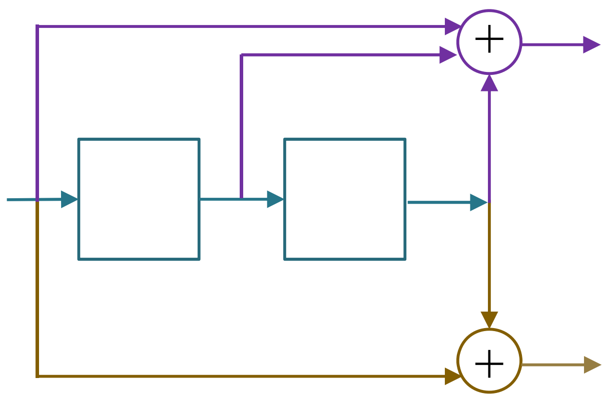
Graphical Representations

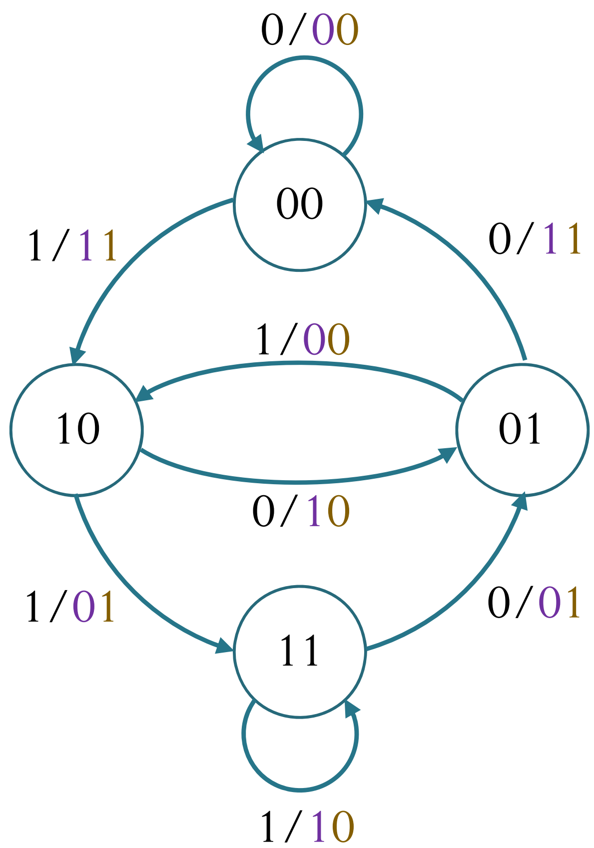
- Three different but related graphical representations have been devised for the study of convolutional encoding:

1. the state diagram
2. the code tree
3. the trellis diagram

State (transition) Diagram

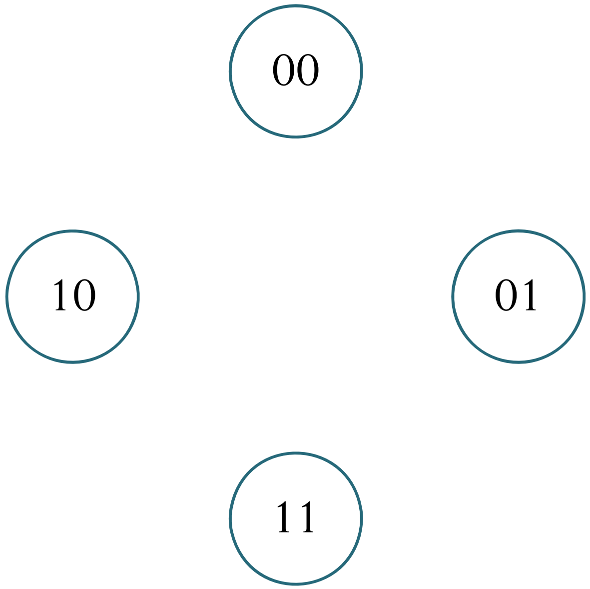
- The encoder behavior can be seen from the perspective of a finite state machine with its state (transition) diagram.

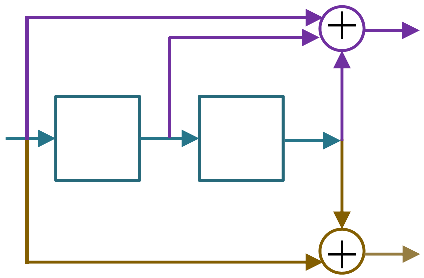




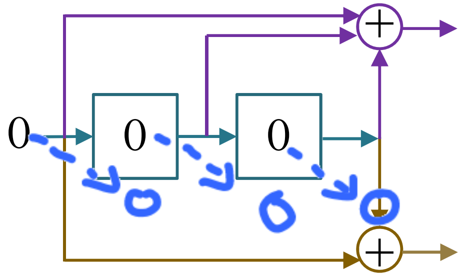
A four-state directed graph that uniquely represents the input-output relation of the encoder.

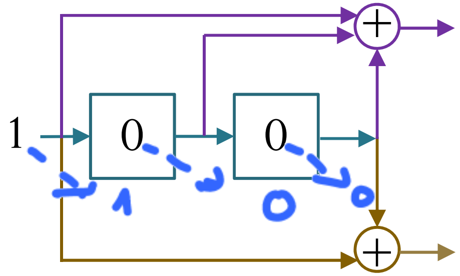
State Diagram

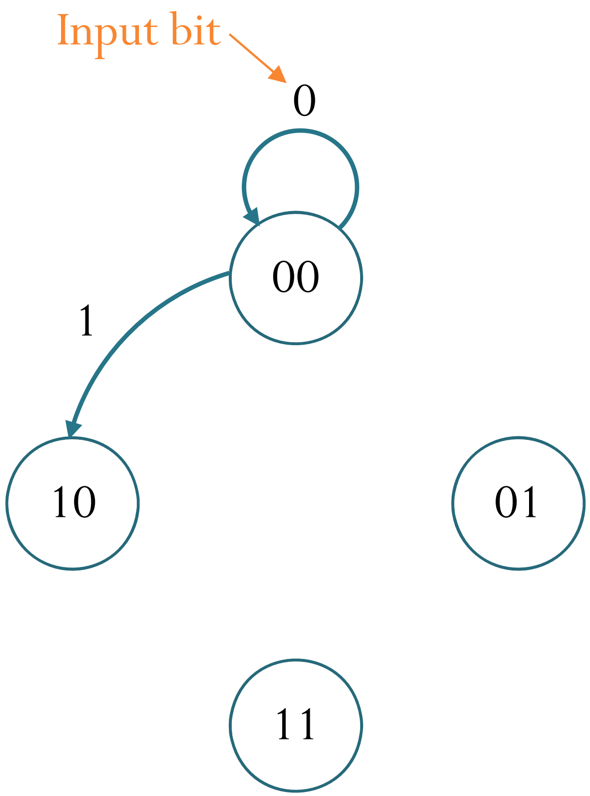


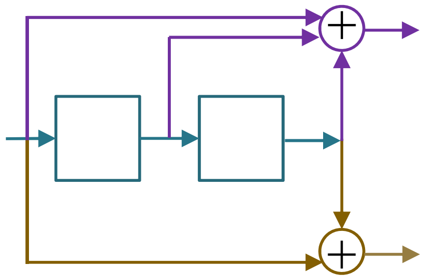


State Diagram

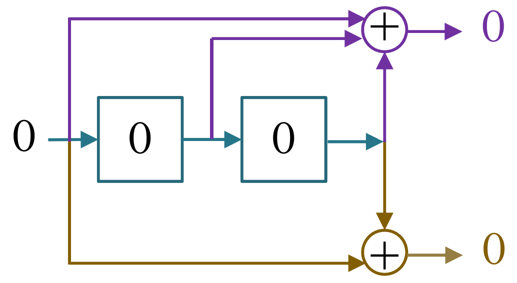


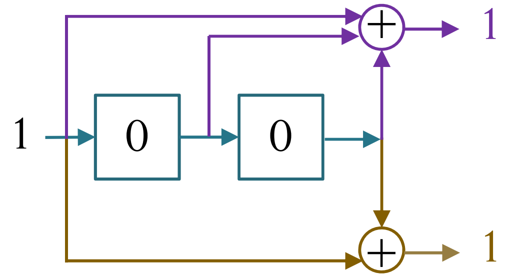


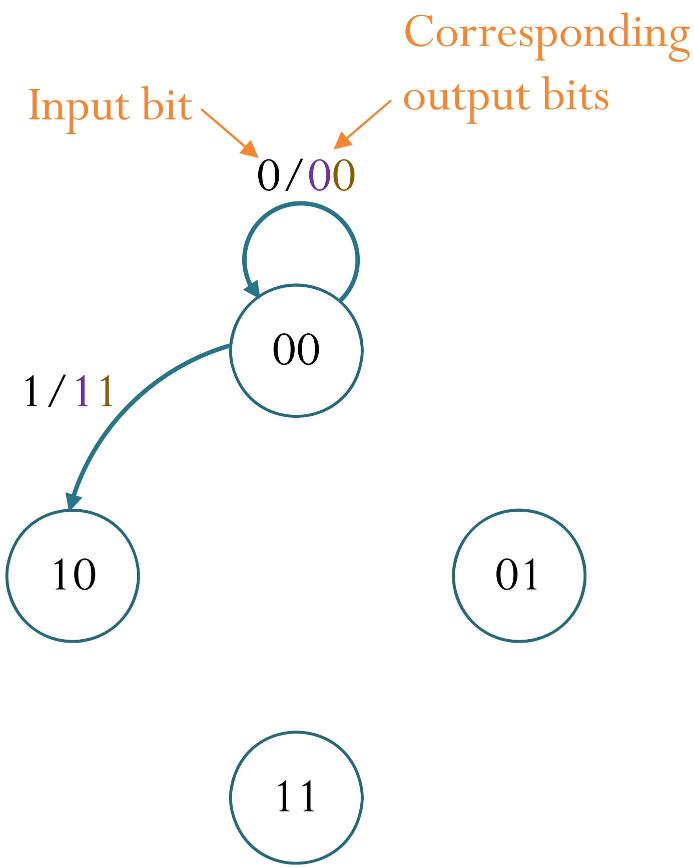


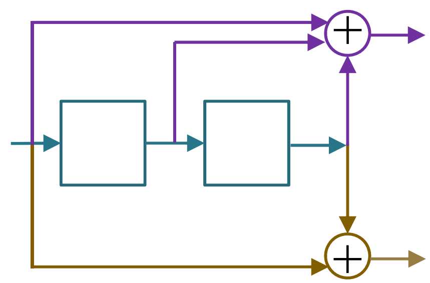


State Diagram

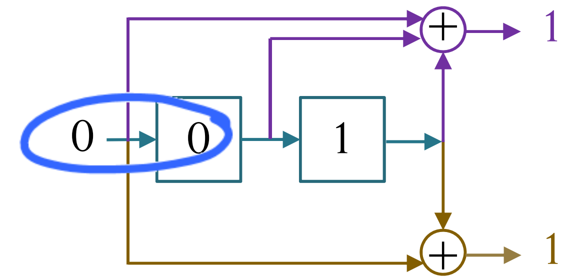


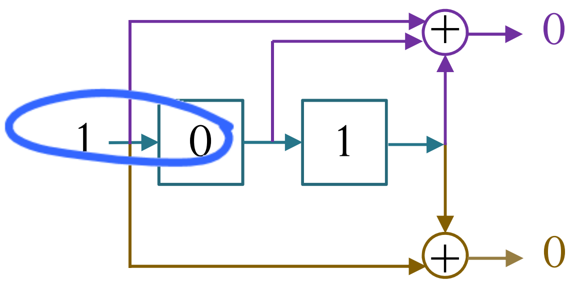


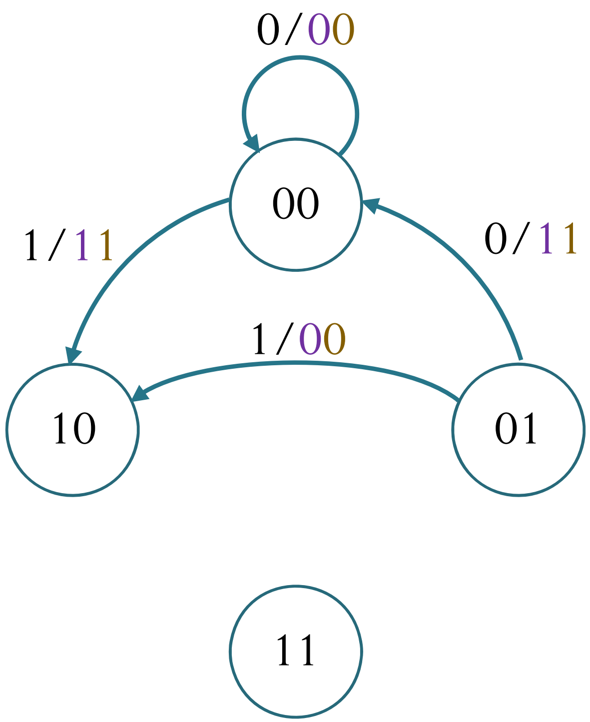


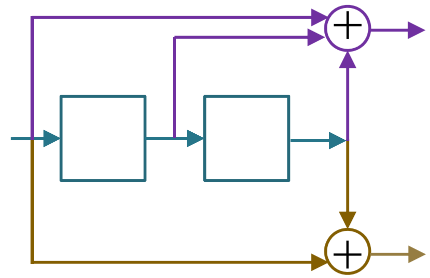


State Diagram

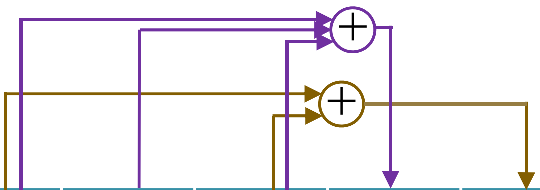




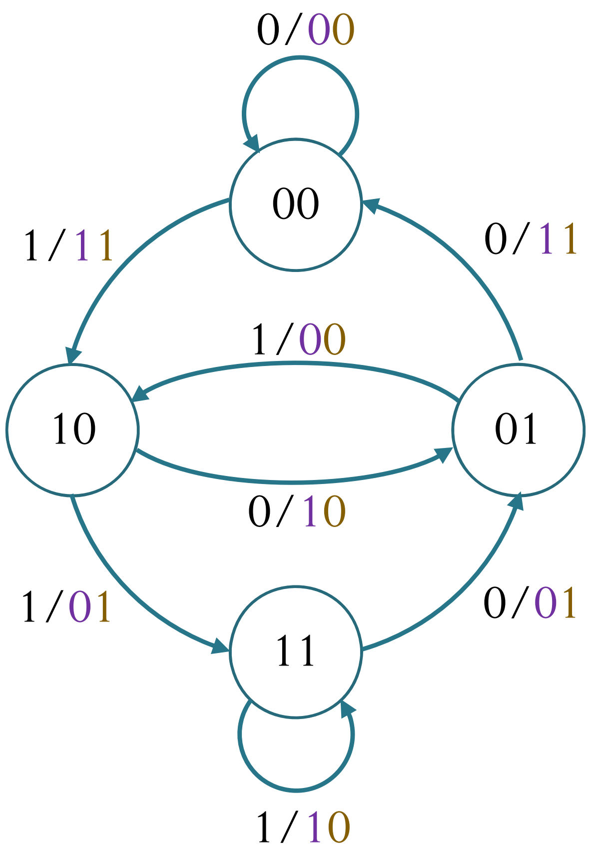


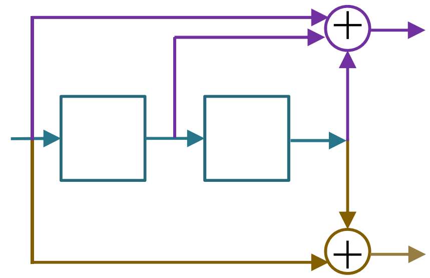


State Diagram



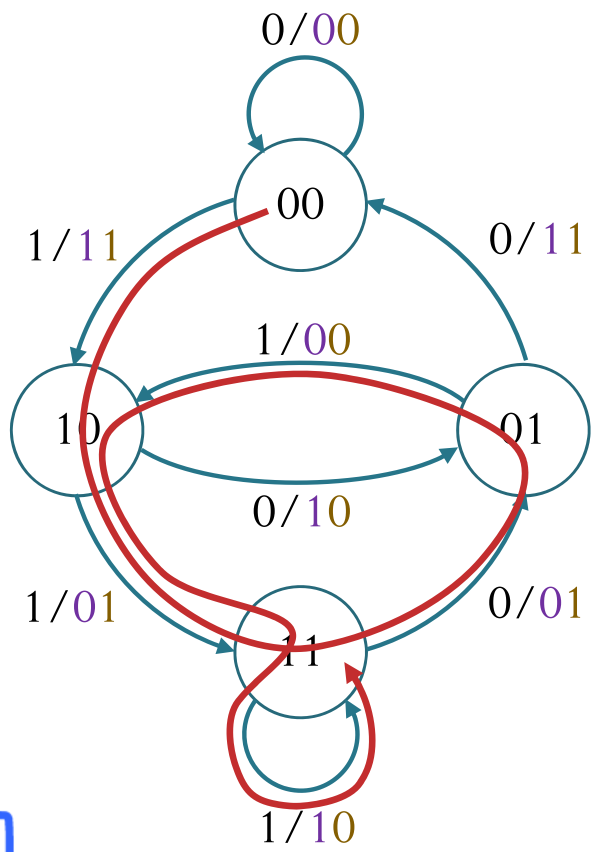
b	s_0	s_1	$x^{(1)}$	$x^{(2)}$
0	0	0	0	0
1	0	0	1	1
0	0	1	1	1
1	0	1	0	0
0	1	0	1	0
1	1	0	0	1
0	1	1	0	1
1	1	1	1	0





"tracing"

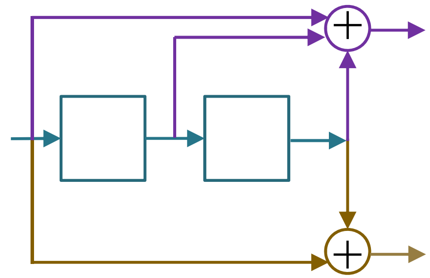
State Diagram



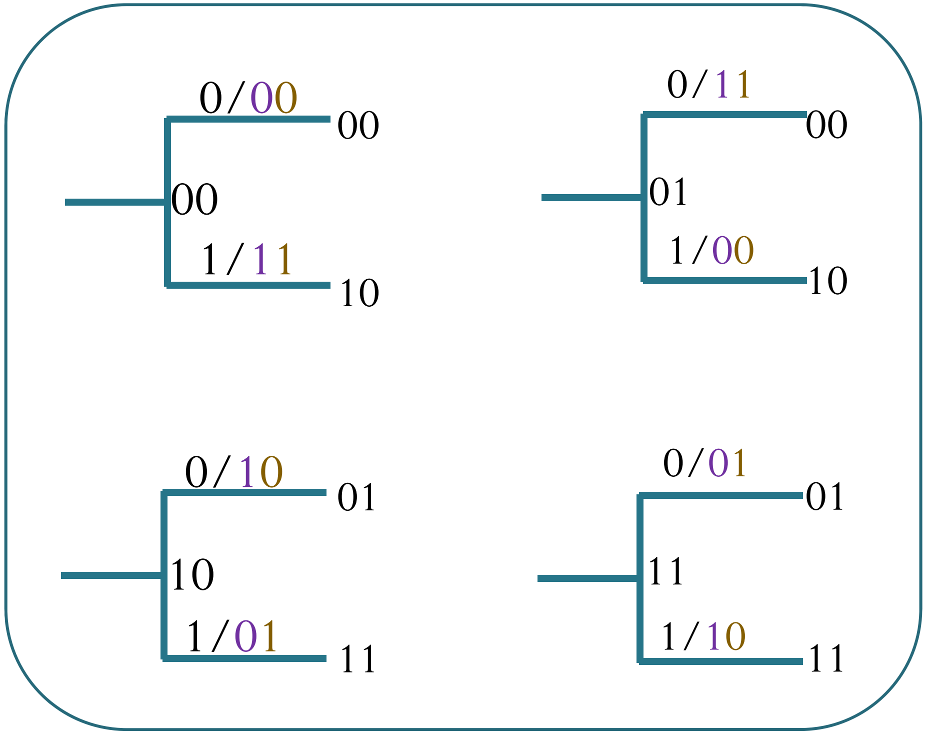
b = [1 1 0 1 1 1]

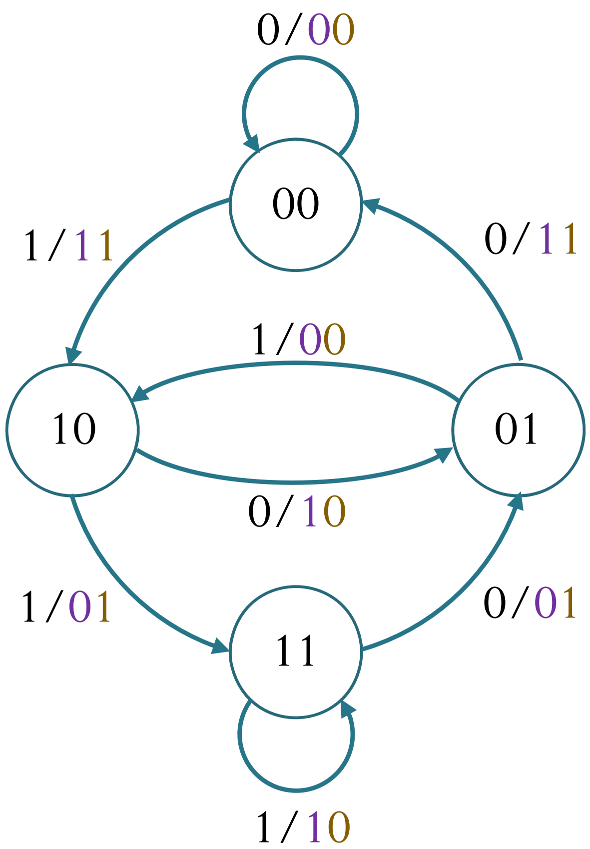
Input	1	1	0	1	1	1
Output	11	01	01	00	01	10

x = [1 1 0 1 0 1 0 0 0 1 1 0]

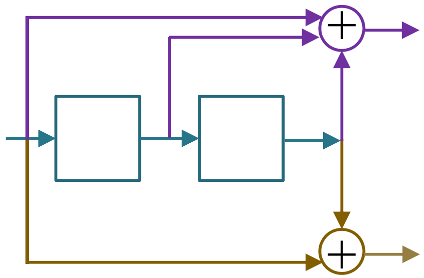


Parts for Code Tree





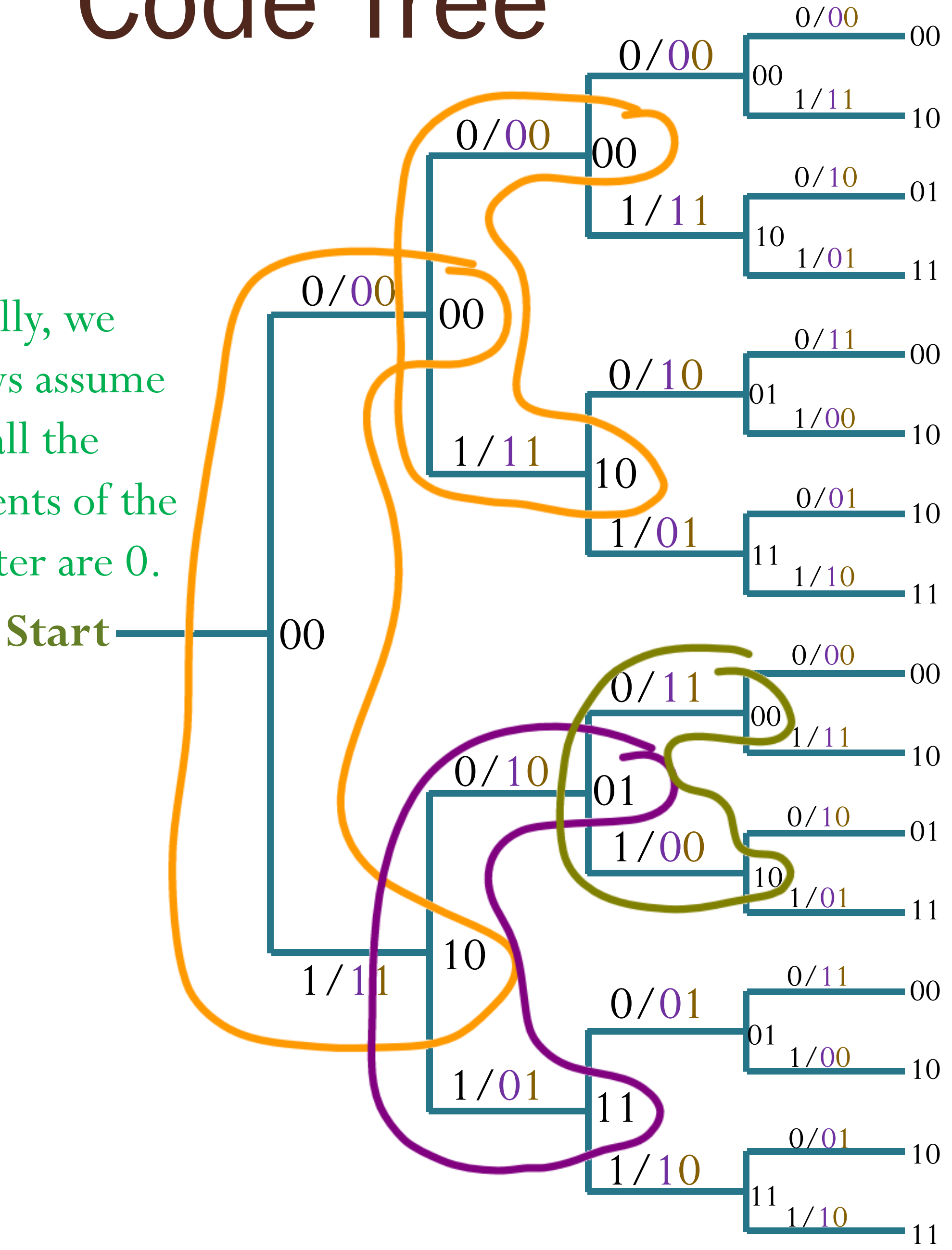
Two branches initiate from each node,
the upper one for 0 and
the lower one for 1.

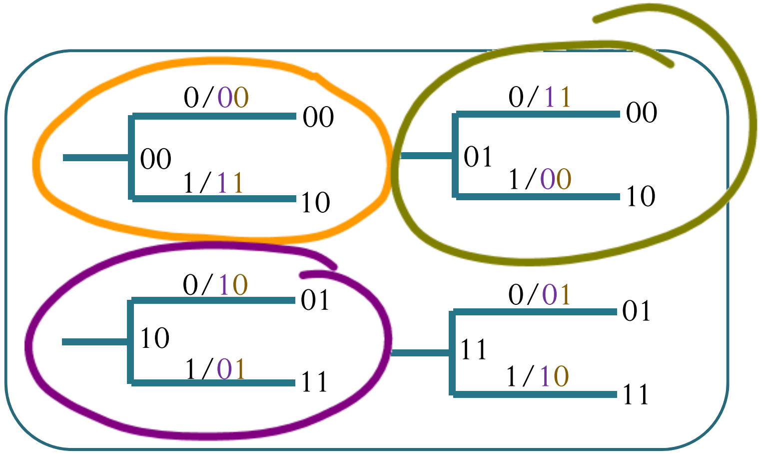


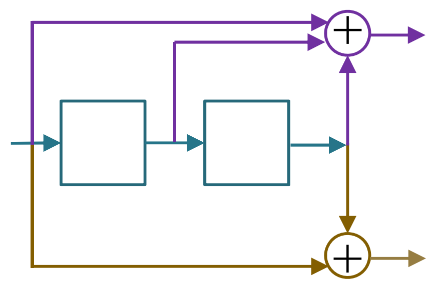
Show the coded output for any possible sequence of data digits.

Code Tree

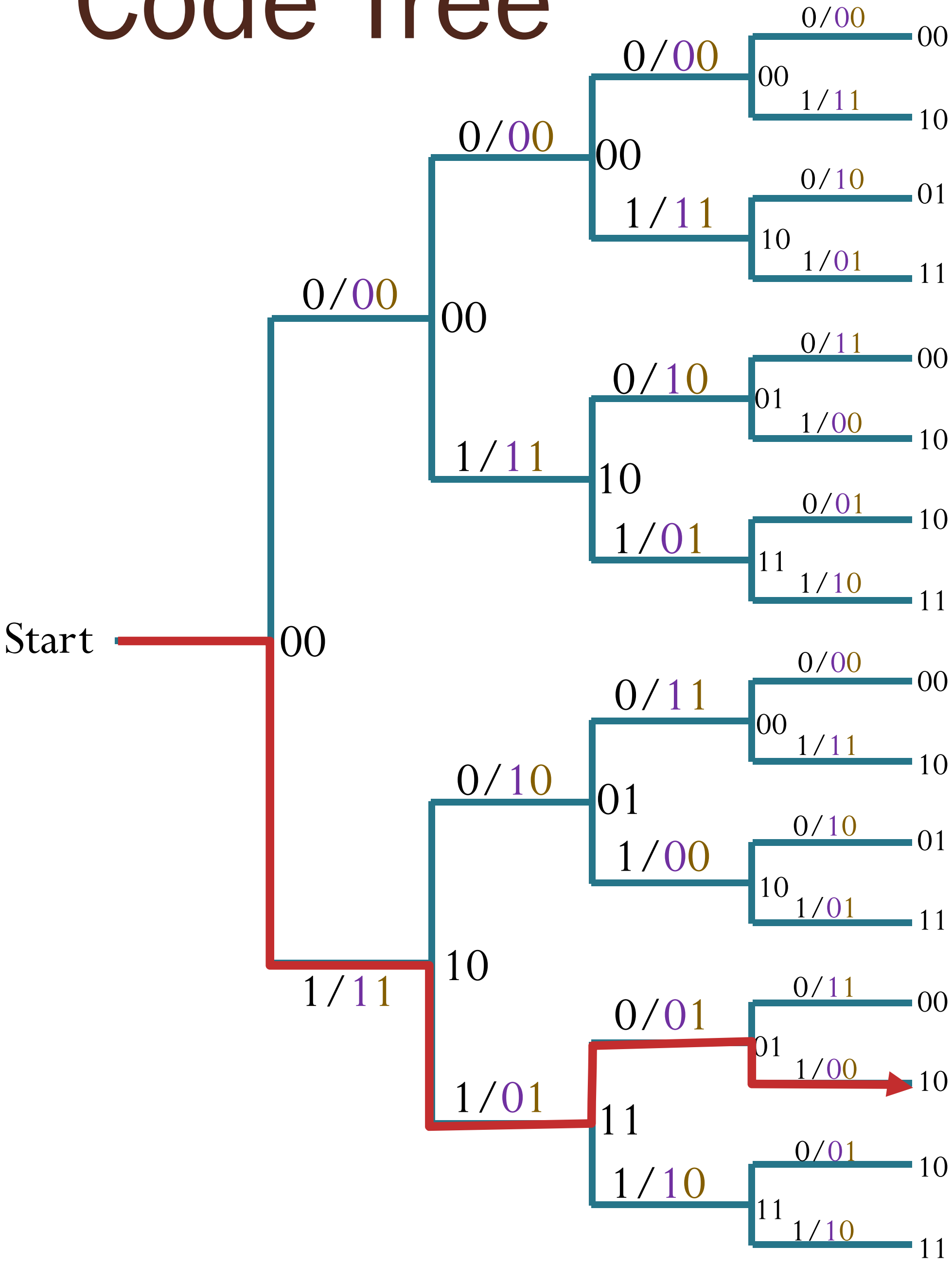
Initially, we always assume that all the contents of the register are 0.



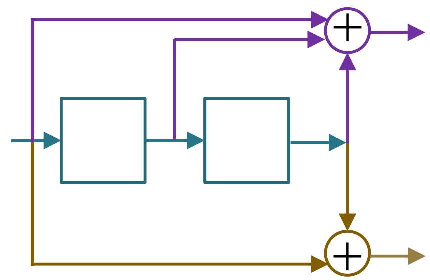




Code Tree

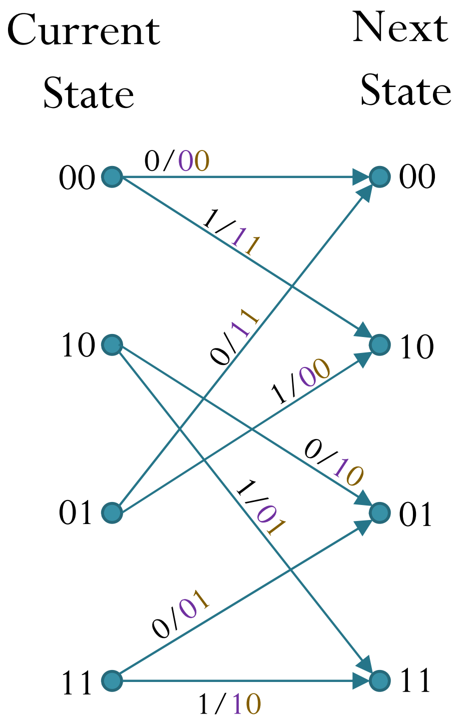


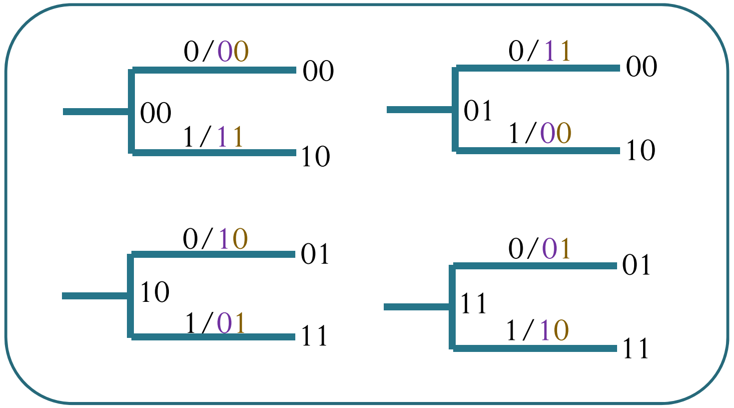
Input	1	1	0	1
Output	11	01	01	00

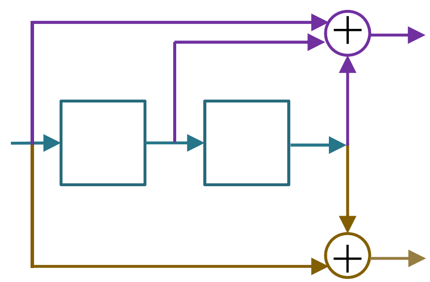


Code Trellis

[Carlson & Crilly, 2009, p. 620]

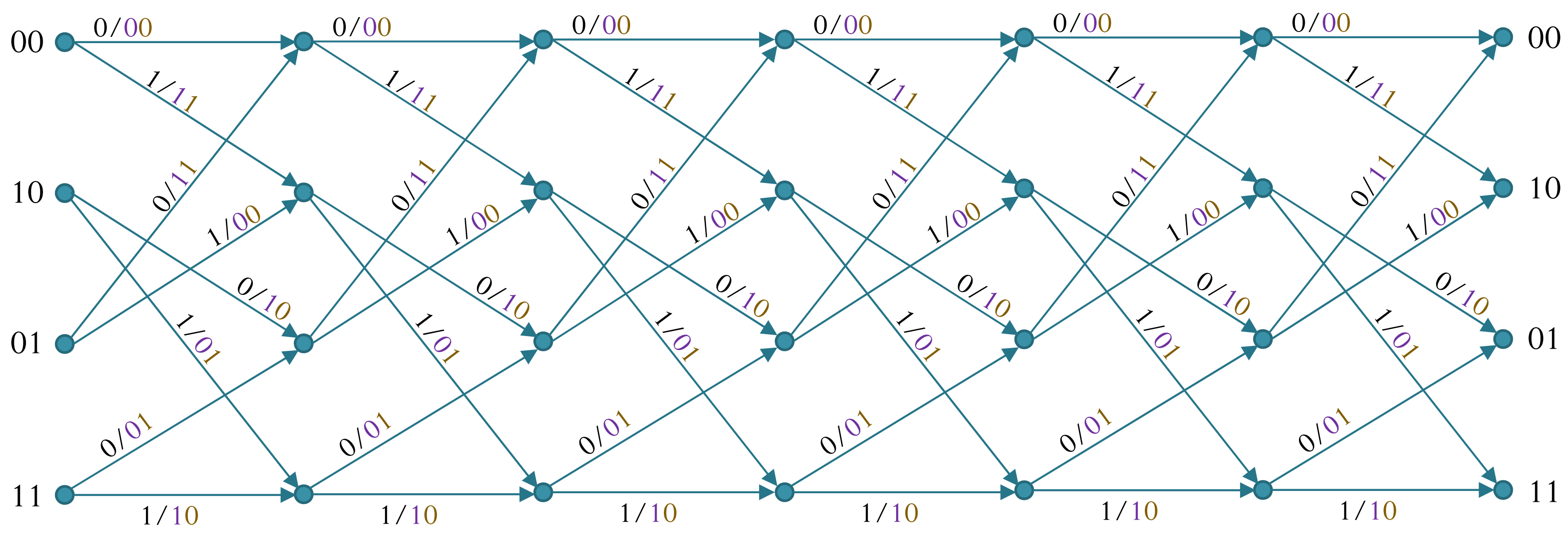


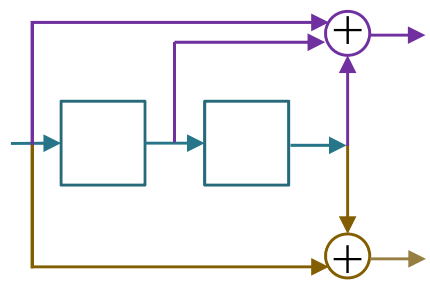




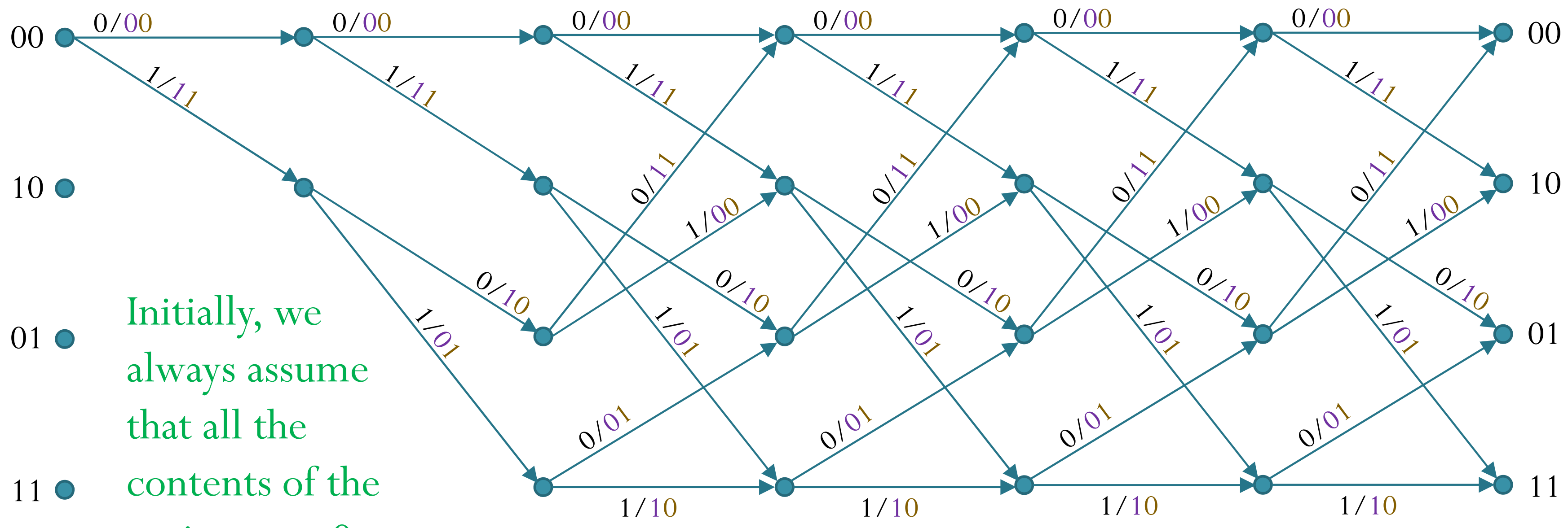
Trellis Diagram

Another useful way of representing the code tree.



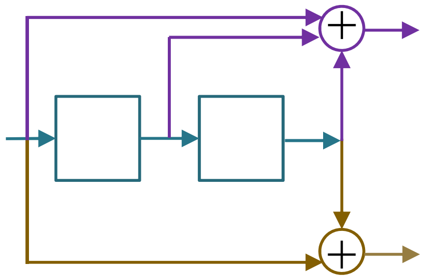


Trellis Diagram

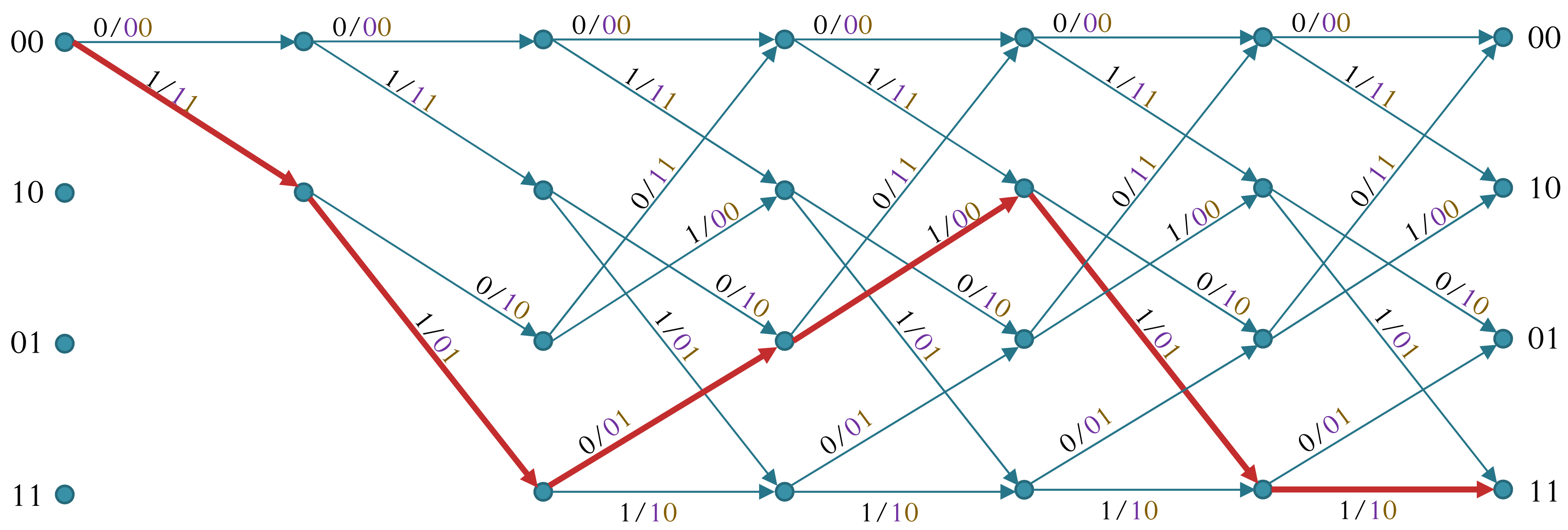


Initially, we always assume that all the contents of the register are 0.

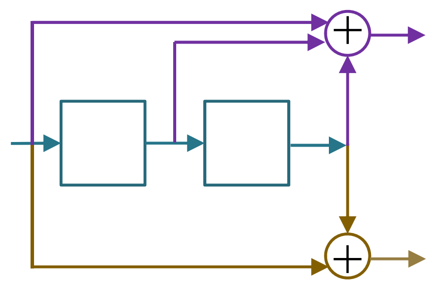
Each path that traverses through the trellis represents a valid codeword.



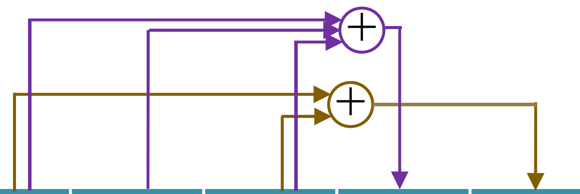
Trellis Diagram



Input	1	1	0	1	1	1
Output	11	01	01	00	01	10

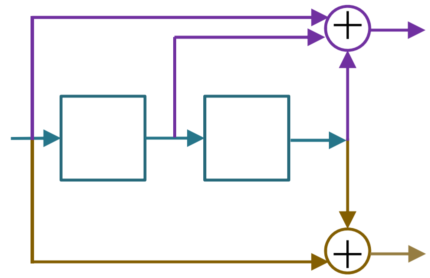


Directly Finding the Output



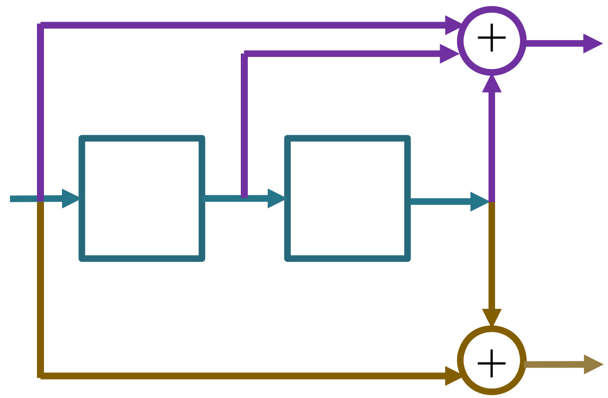
b	s_0	s_1	$x^{(1)}$	$x^{(2)}$
1	0	0	1	1
1	1	0	0	1
0	1	1	0	1
1	0	1	0	0
1	1	0	0	1
1	1	1	1	0

Input	1	1	0	1	1	1
Output	11	01	01	00	01	10



Direct Minimum Distance Decoding

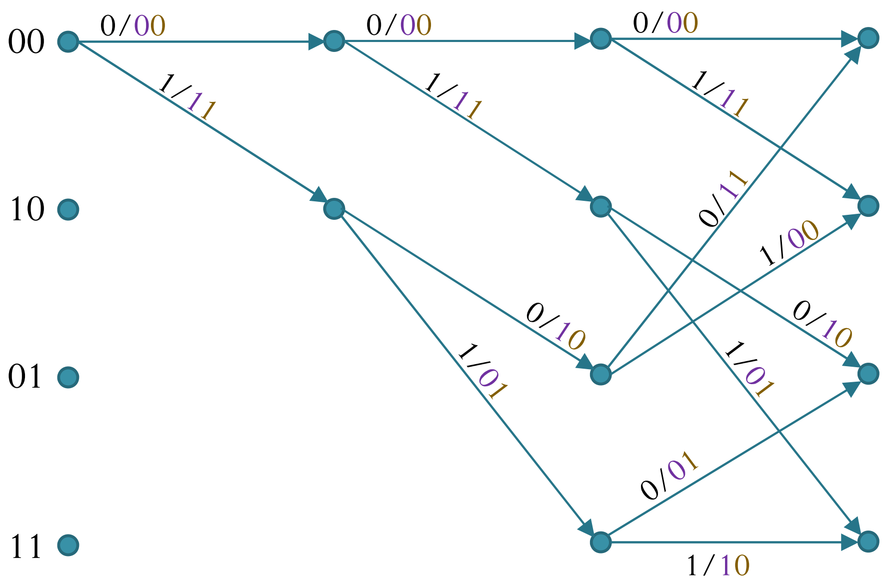
- Suppose $\underline{\boldsymbol{y}} = [11\ 01\ 11]$.
- Find $\underline{\hat{\mathbf{b}}}$.
  - Find the message $\underline{\hat{\mathbf{b}}}$ which corresponds to the (valid) codeword $\underline{\hat{\mathbf{x}}}$ with minimum (Hamming) distance from $\underline{\boldsymbol{y}}$.



Direct Minimum Distance Decoding

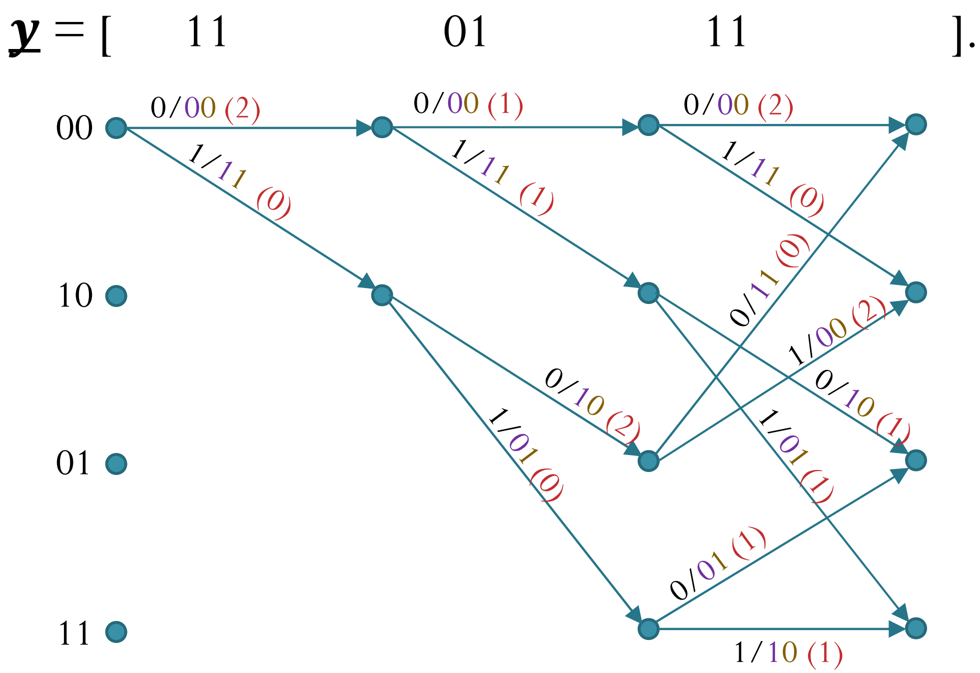
- Suppose $\underline{\boldsymbol{y}} = [11\ 01\ 11]$.
- Find $\hat{\underline{\mathbf{b}}}$.
  - Find the message $\hat{\underline{\mathbf{b}}}$ which corresponds to the (valid) codeword $\hat{\underline{\mathbf{x}}}$ with minimum (Hamming) distance from $\underline{\boldsymbol{y}}$.

$\underline{\boldsymbol{y}} = [\quad 11 \quad 01 \quad 11 \quad].$



Direct Minimum Distance Decoding

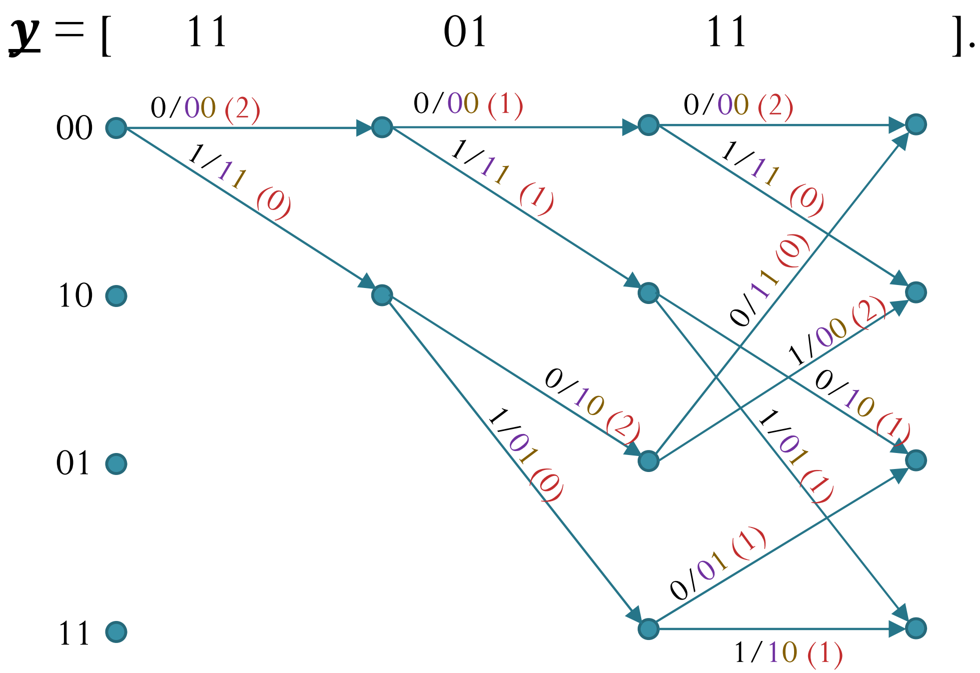
- Suppose $\underline{\boldsymbol{y}}$ = [11 01 11].
- Find $\hat{\underline{\mathbf{b}}}$.
  - Find the message $\hat{\underline{\mathbf{b}}}$ which corresponds to the (valid) codeword $\hat{\underline{\mathbf{x}}}$ with minimum (Hamming) distance from $\underline{\boldsymbol{y}}$.



The number in parentheses on each branch is the branch metric, obtained by counting the differences between the encoded bits and the corresponding bits in $\underline{\boldsymbol{y}}$.

Direct Minimum Distance Decoding

- Suppose $\underline{y}$ = [11 01 11].
- Find $\hat{\underline{\mathbf{b}}}$.
  - Find the message $\hat{\underline{\mathbf{b}}}$ which corresponds to the (valid) codeword $\hat{\underline{\mathbf{x}}}$ with minimum (Hamming) distance from $\underline{y}$.



$\underline{b}$	$d(\underline{x},\underline{y})$
000	2+1+2 = 5
001	2+1+0 = 3
010	2+1+1 = 4
011	2+1+1 = 4
100	0+2+0 = 2
101	0+2+2 = 4
110	0+0+1 = 1
111	0+0+1 = 1

Viterbi decoding

- Developed by Andrew J. Viterbi
  - Also co-founded Qualcomm Inc.
- Published in the paper "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm", IEEE Transactions on Information Theory, Volume IT-13, pages 260-269, in April, **1967**.

https://en.wikipedia.org/wiki/Andrew_Viterbi

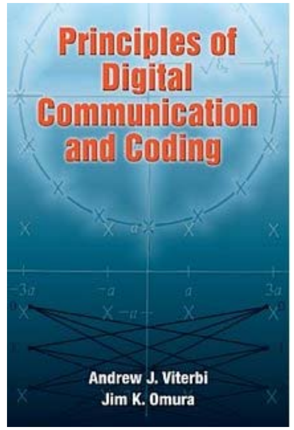
Viterbi and His Decoding Algorithm

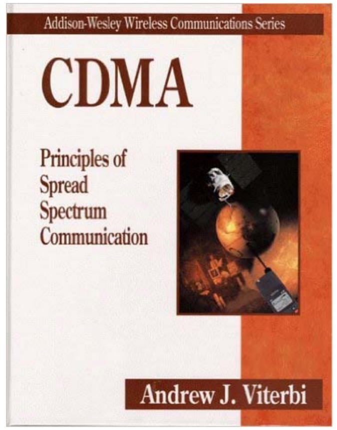
[http://viterbi.usc.edu/about/viterbi/viterbi_video.htm]

Andrew J. Viterbi

- 1991: Claude E. Shannon Award
- 1952-1957: MIT BS & MS
  - Studied electronics and communications theory under such renowned scholars as Norbert Wiener, Claude Shannon, Bruno Rossi and Roberto Fano.
- 1962: Earned one of the first doctorates in electrical engineering granted at the University of Southern California (USC)
  - Ph.D. dissertation: error correcting codes
- 2004: USC Viterbi School of Engineering

  named in recognition of his $52 million gift





USC University of Southern California

USC Viterbi School of Engineering

Andrew J. Viterbi

[http://viterbi.usc.edu/about/viterbi/viterbi_video.htm]

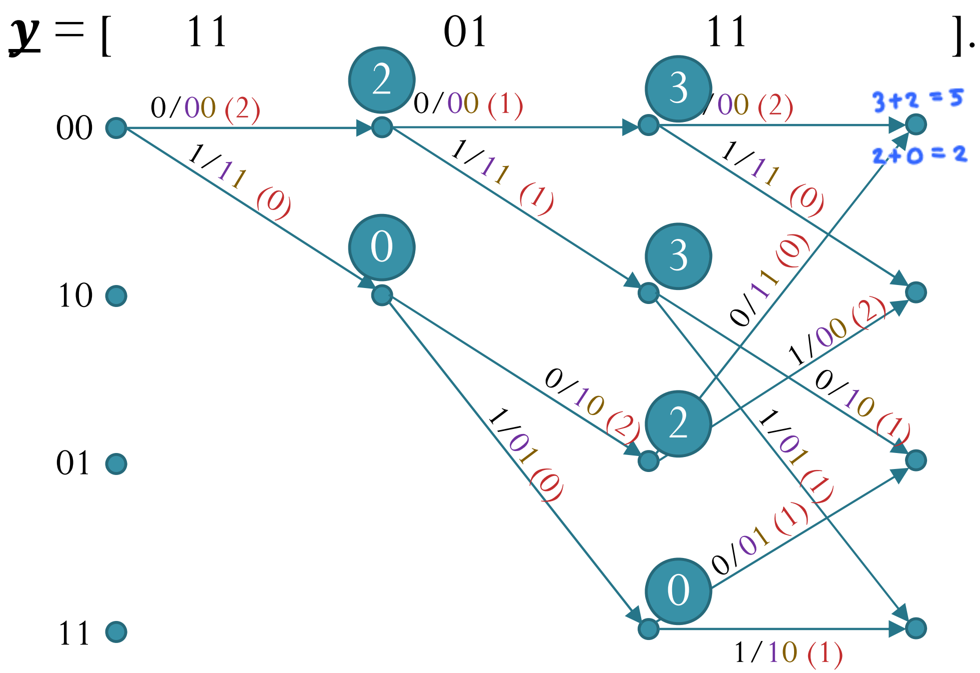
Andrew J. Viterbi

- Cofounded Qualcomm
- Helped to develop the CDMA standard for cellular networks.
- 1998 Golden Jubilee Award for Technological Innovation
  - To commemorate the 50th Anniversary of Information Theory
  - Given to the authors of discoveries, advances and inventions that have had a profound impact in the technology of information transmission, processing and compression.

1. Norman Abramson: For the invention of the first random-access communication protocol.
2. Elwyn Berlekamp: For the invention of a computationally efficient algebraic decoding algorithm.
3. Claude Berrou, Alain Glavieux and Punya Thitimajshima: For the invention of turbo codes.
4. Ingrid Daubechies: For the invention of wavelet-based methods for signal processing.
5. Whitfield Diffie and Martin Hellman: For the invention of public-key cryptography.
6. Peter **Elias**: For the invention of convolutional codes.
7. G. David Forney, Jr: For the invention of concatenated codes and a generalized minimum-distance decoding algorithm.
8. Robert M. Gray: For the invention and development of training mode vector quantization.
9. David **Huffman**: For the invention of the Huffman minimum-length lossless data-compression code.
10. Kees A. Schouhamer Immink: For the invention of constrained codes for commercial recording systems.
11. Abraham Lempel and Jacob Ziv: For the invention of the Lempel-Ziv universal data compression algorithm.
12. Robert W. Lucky: For the invention of pioneering adaptive equalization methods.
13. Dwight O. North: For the invention of the matched filter.
14. Irving S. Reed: For the co-invention of the Reed-Solomon error correction codes.
15. Jorma Rissanen: For the invention of arithmetic coding.
16. Gottfried Ungerboeck: For the invention of trellis coded modulation.
17. Andrew J. **Viterbi**: For the invention of the Viterbi algorithm.

Viterbi Decoding

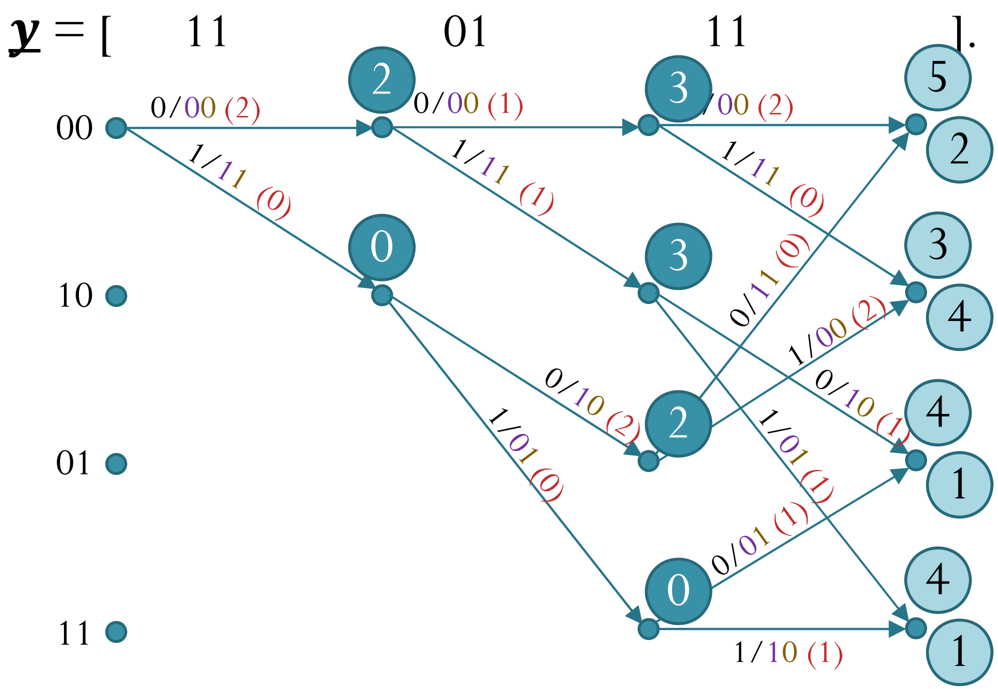
- Suppose $\underline{\boldsymbol{y}} = [11\ 01\ 11]$.
- Find $\hat{\underline{\mathbf{b}}}$.
  - Find the message $\hat{\underline{\mathbf{b}}}$ which corresponds to the (valid) codeword $\hat{\underline{\mathbf{x}}}$ with minimum (Hamming) distance from $\underline{\boldsymbol{y}}$.



Each **circled number** at a node is the running (cumulative) path metric, obtained by summing branch metrics (distance) up to that node.

Viterbi Decoding

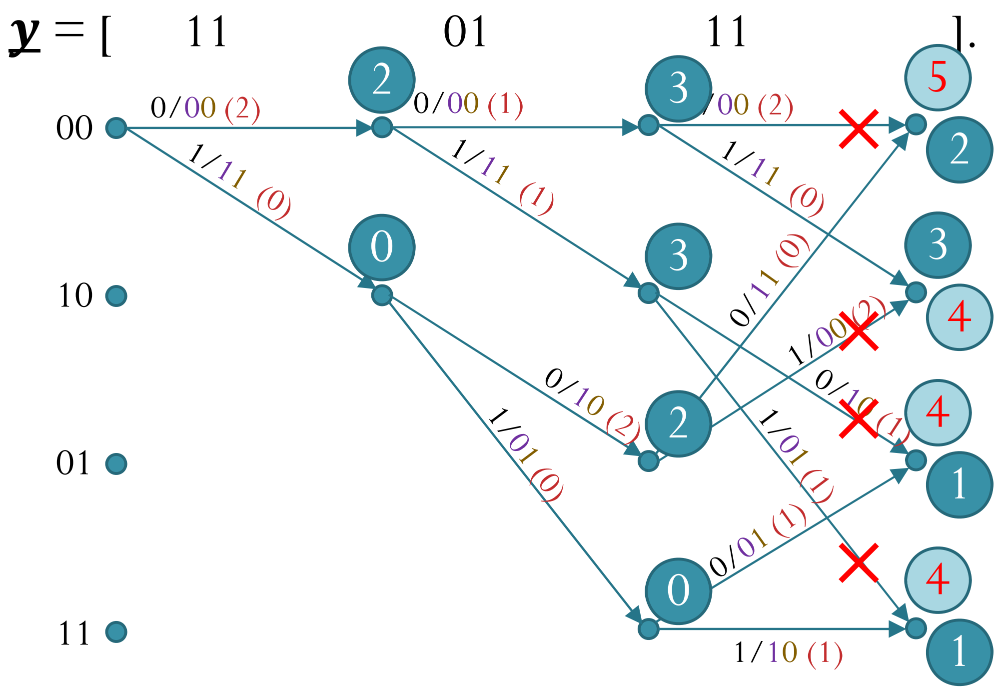
- Suppose $\underline{\boldsymbol{y}}$ = [11 01 11].
- Find $\hat{\underline{\mathbf{b}}}$.
  - Find the message $\hat{\underline{\mathbf{b}}}$ which corresponds to the (valid) codeword $\hat{\underline{\mathbf{x}}}$ with minimum (Hamming) distance from $\underline{\boldsymbol{y}}$.



- For the last column of nodes, each of the nodes has two branches going into it.
- So, there are two possible cumulative distance values.

Viterbi Decoding

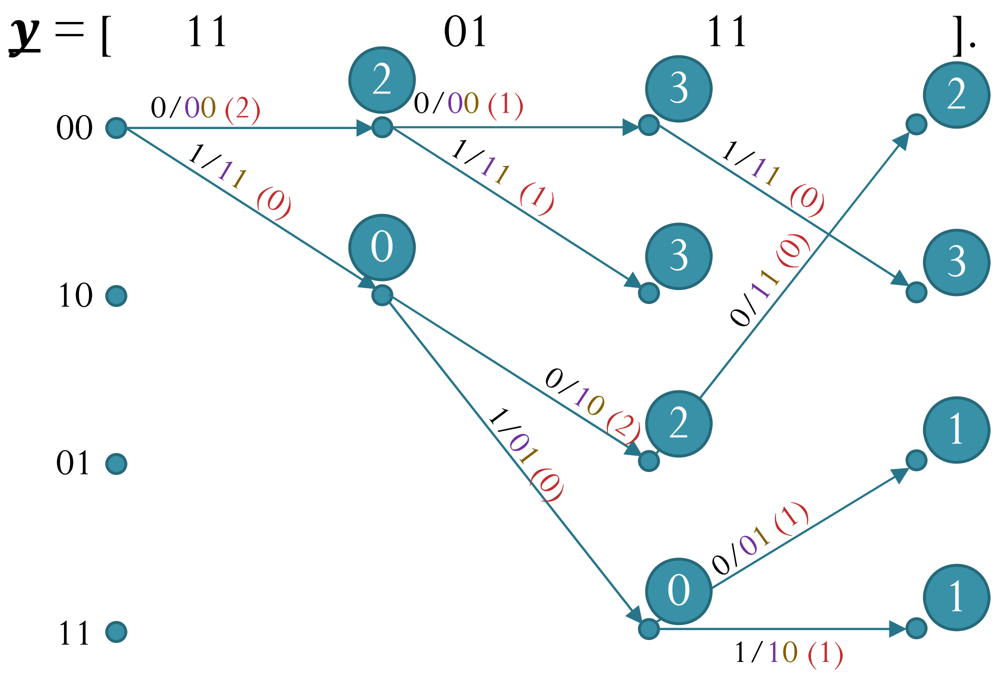
- Suppose $\underline{\mathbf{y}}$ = [11 01 11].
- Find $\hat{\underline{\mathbf{b}}}$.
  - Find the message $\hat{\underline{\mathbf{b}}}$ which corresponds to the (valid) codeword $\hat{\underline{\mathbf{x}}}$ with minimum (Hamming) distance from $\underline{\mathbf{y}}$.



- For the last column of nodes, each of the nodes has two branches going into it.
- So, there are two possible cumulative distance values.
- We **discard the larger-metric path** because, regardless of what happens subsequently, this path will have a larger Hamming distance from $\underline{\mathbf{y}}$.

Viterbi Decoding

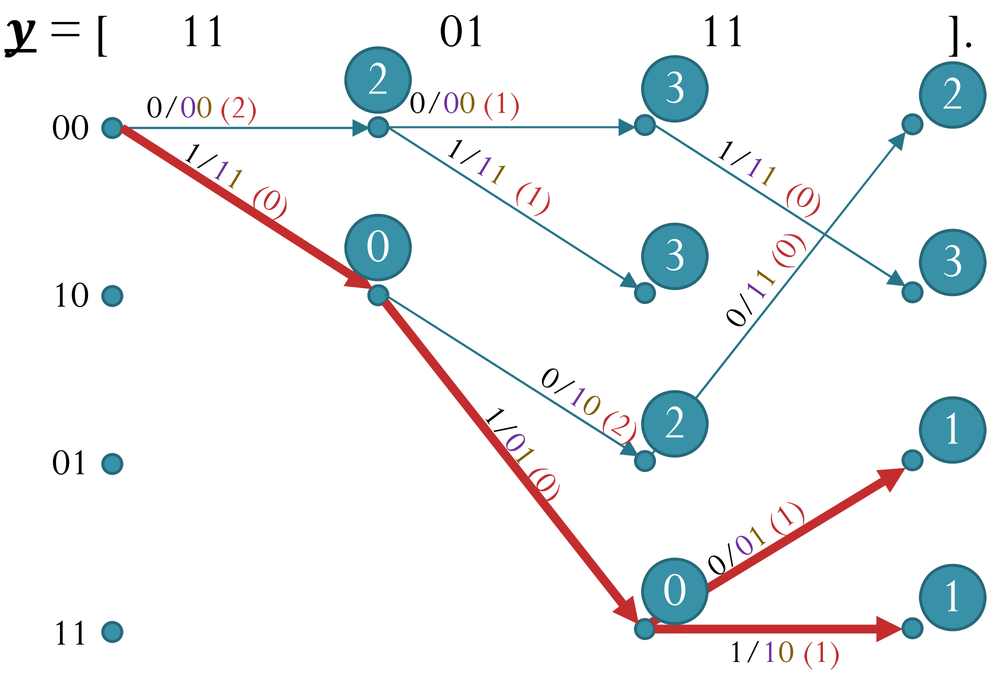
- Suppose $\underline{y}$ = [11 01 11].
- Find $\hat{\underline{\mathbf{b}}}$.
  - Find the message $\hat{\underline{\mathbf{b}}}$ which corresponds to the (valid) codeword $\hat{\underline{\mathbf{x}}}$ with minimum (Hamming) distance from $\underline{y}$.



- For the last column of nodes, each of the nodes has two branches going into it.
- So, there are two possible cumulative distance values.
- We **discard the larger-metric path** because, regardless of what happens subsequently, this path will have a larger Hamming distance from $\underline{y}$.

Viterbi Decoding

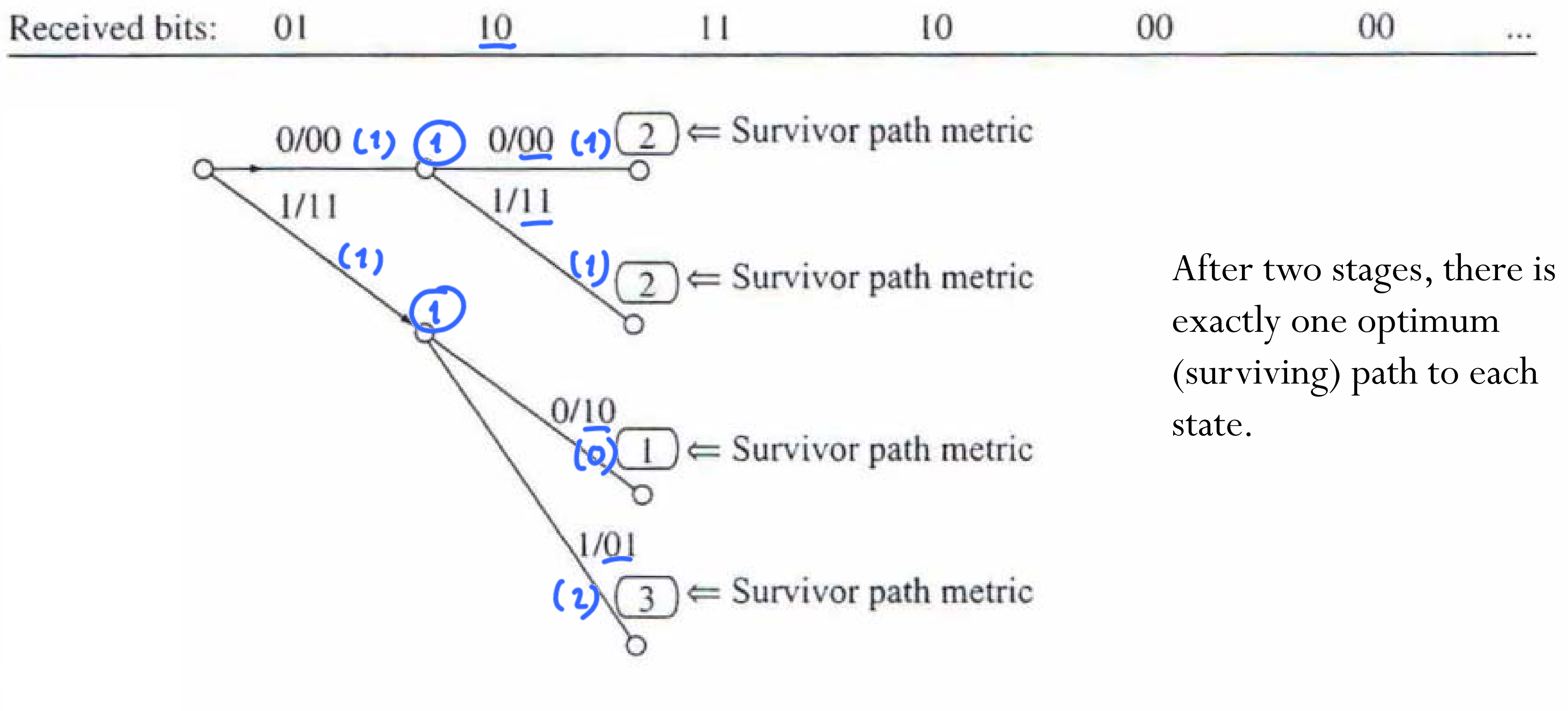
- Suppose $\underline{\boldsymbol{y}}$ = [11 01 11].
- Find $\hat{\underline{\mathbf{b}}}$.
  - Find the message $\hat{\underline{\mathbf{b}}}$ which corresponds to the (valid) codeword $\hat{\underline{\mathbf{x}}}$ with minimum (Hamming) distance from $\underline{\boldsymbol{y}}$.



- So, the codewords which are nearest to $\underline{\boldsymbol{y}}$ is [11 01 01] or [11 01 10].
- The corresponding messages are [110] or [111], respectively.

Ex. Viterbi Decoding

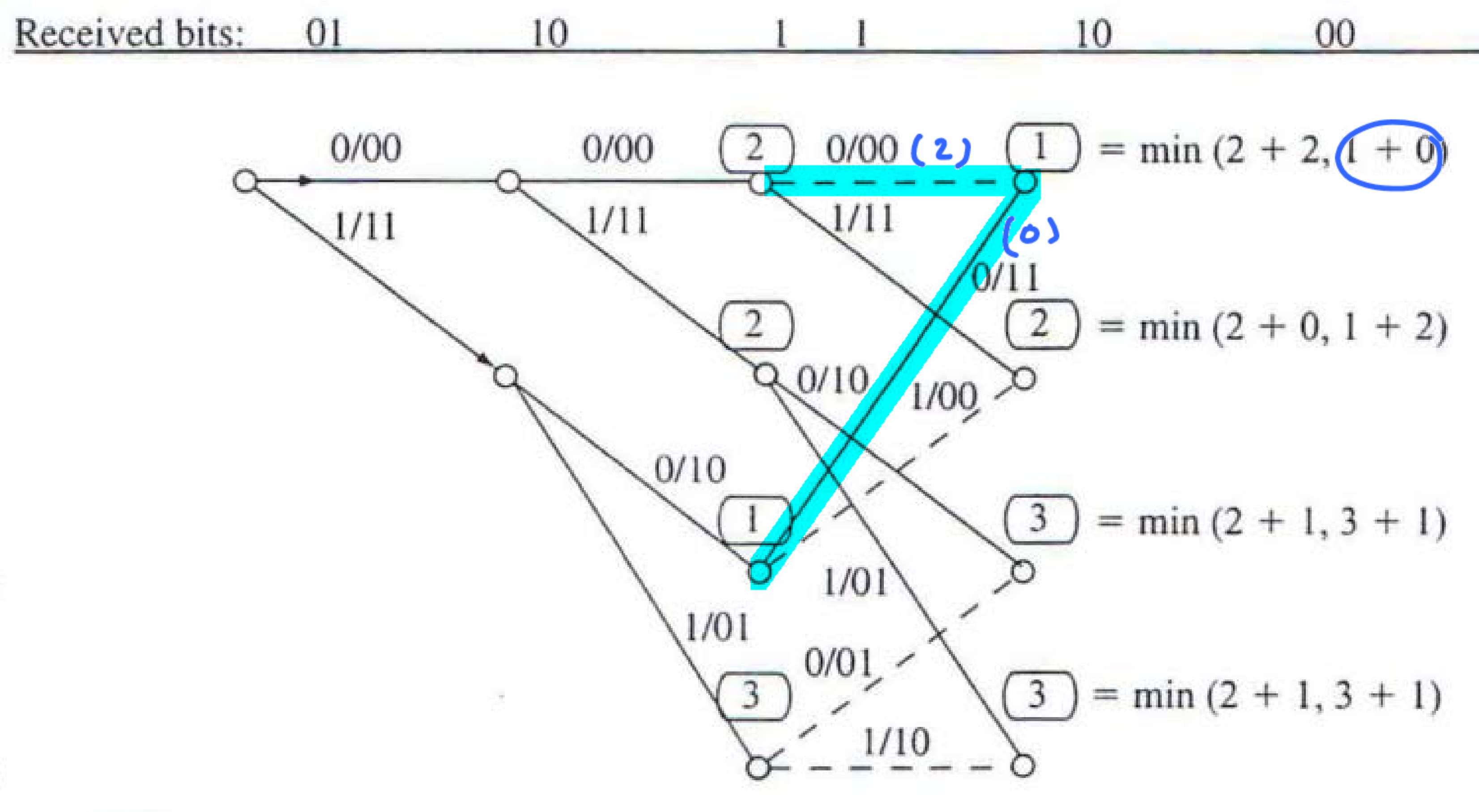
- Suppose $\underline{y}$ = [01 10 11 10 00 00].



After two stages, there is exactly one optimum (surviving) path to each state.

Ex. Viterbi Decoding

- Suppose $\underline{\boldsymbol{y}}$ = [01 10 11 10 00 00].

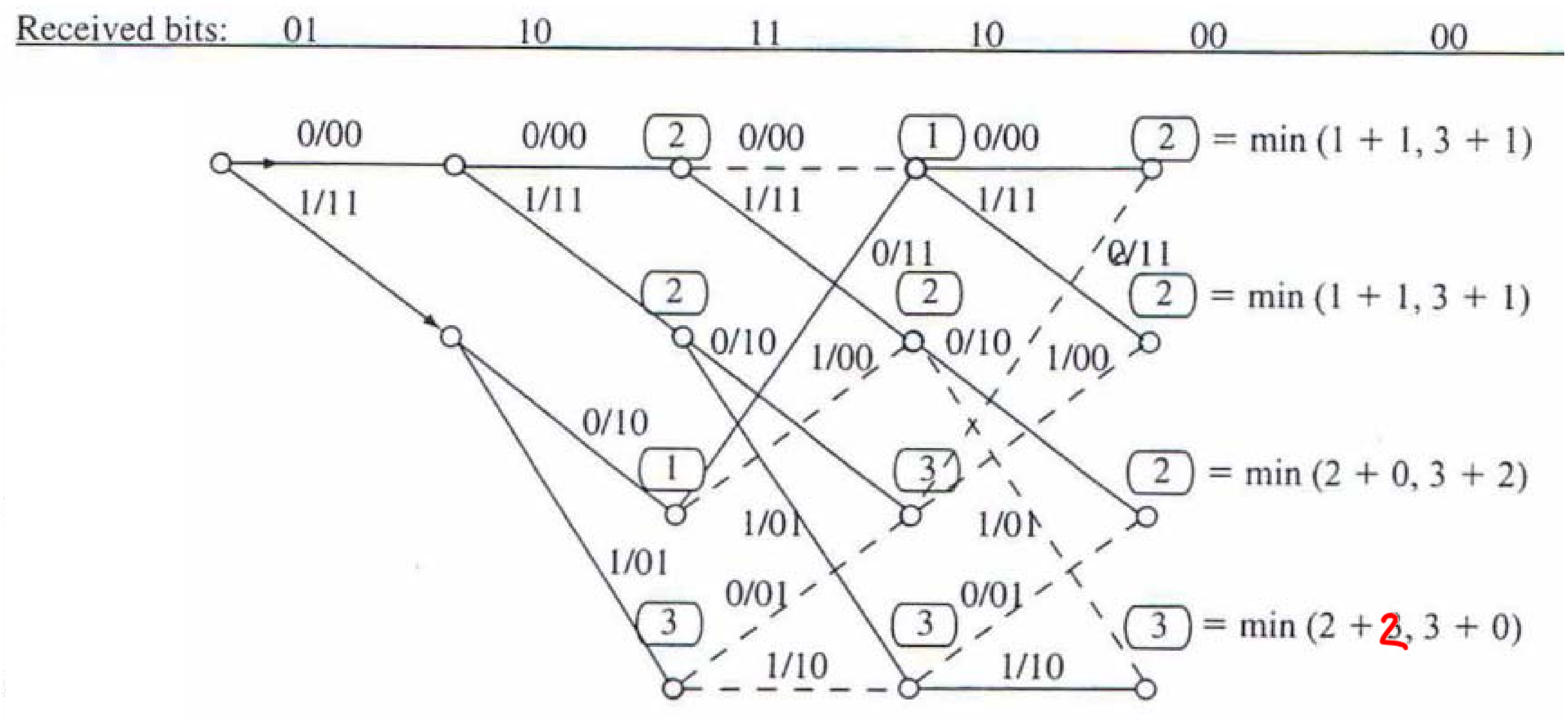


Each state at stage 3 has two possible paths. We keep the optimum path with the minimum distance (solid line).

The same procedure is repeated for stages 4, 5, and 6.

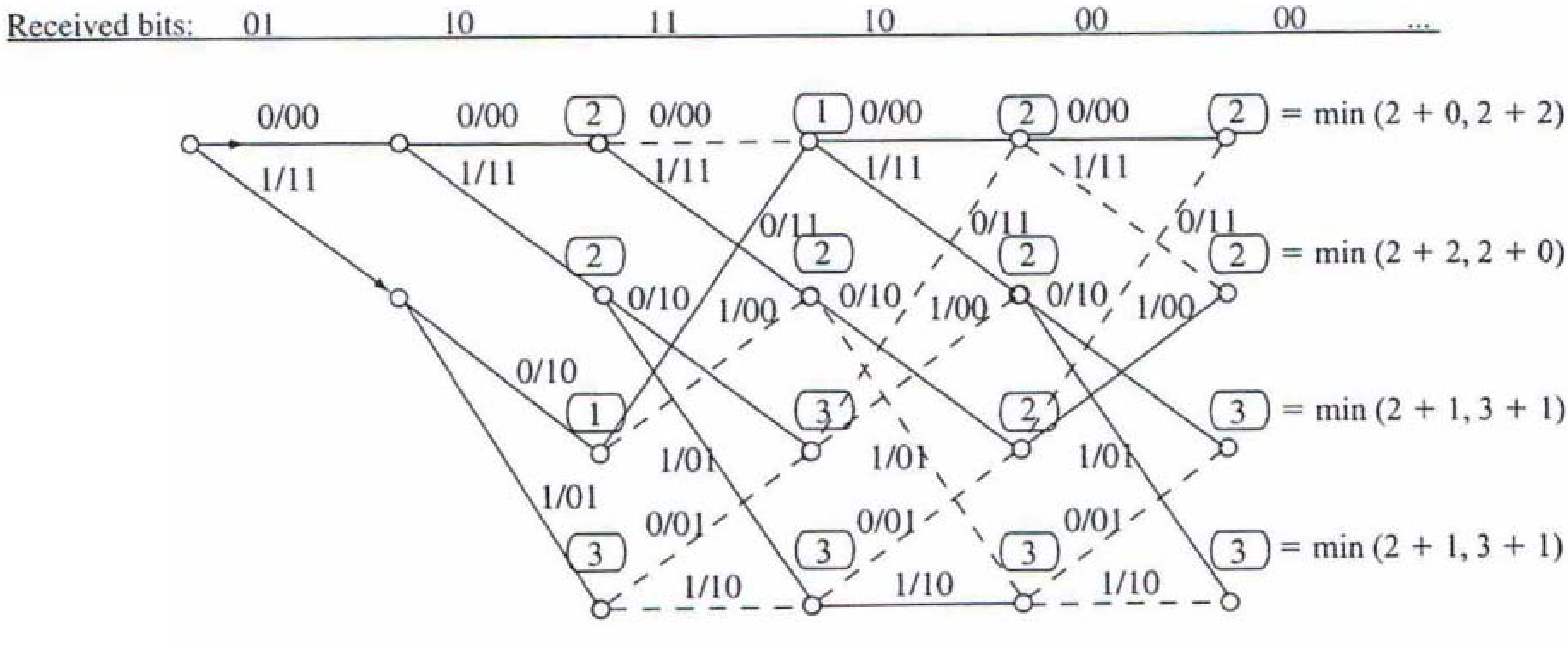
Ex. Viterbi Decoding

- Suppose $\boldsymbol{y}$ = [01 10 11 10 00 00].



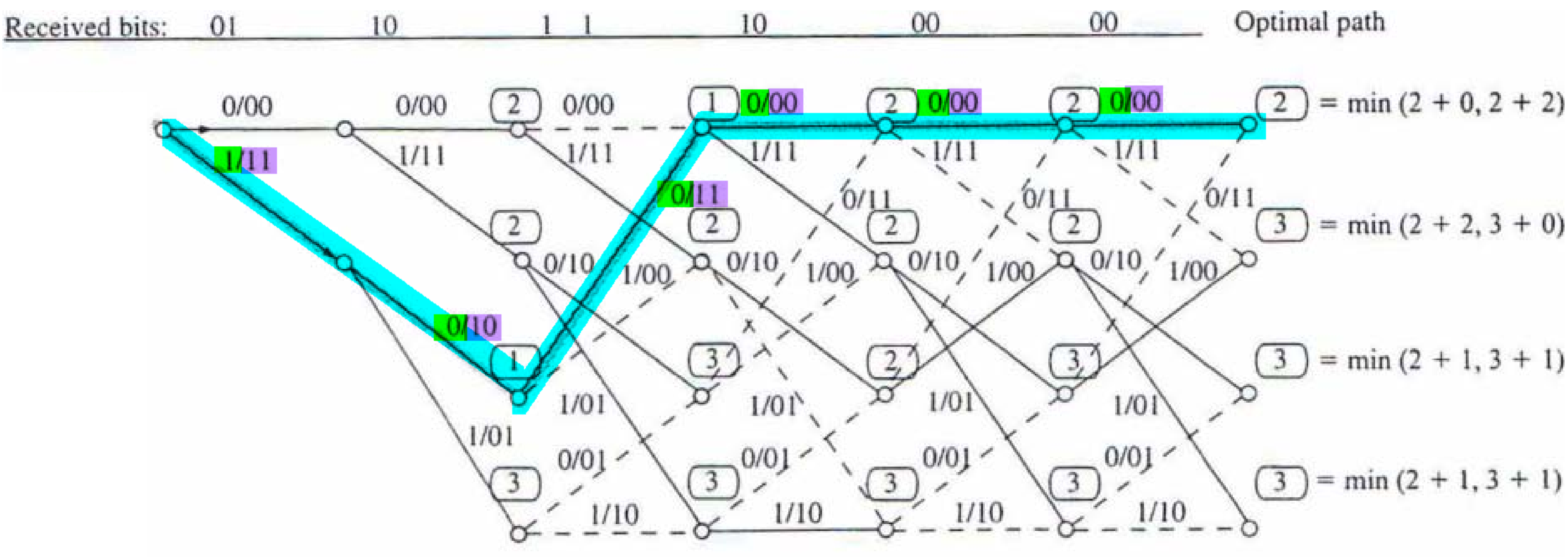
Ex. Viterbi Decoding

- Suppose $\underline{\boldsymbol{y}}$ = [01 10 11 10 00 00].



Ex. Viterbi Decoding

- Suppose $\underline{\boldsymbol{y}}$ = [01 10 11 10 00 00].

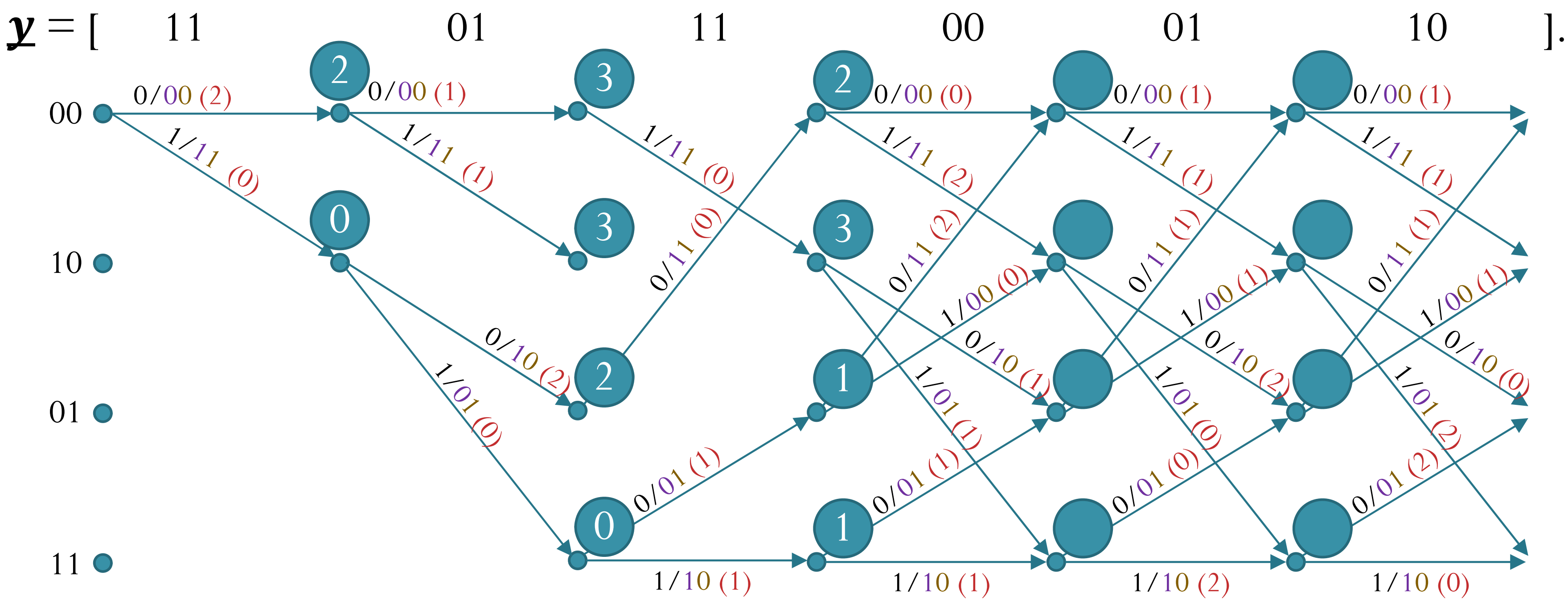


$\hat{\underline{\mathbf{x}}}$ = [11 10 11 00 00 00]

$\hat{\underline{\mathbf{b}}}$ = [1 0 0 0 0 0]

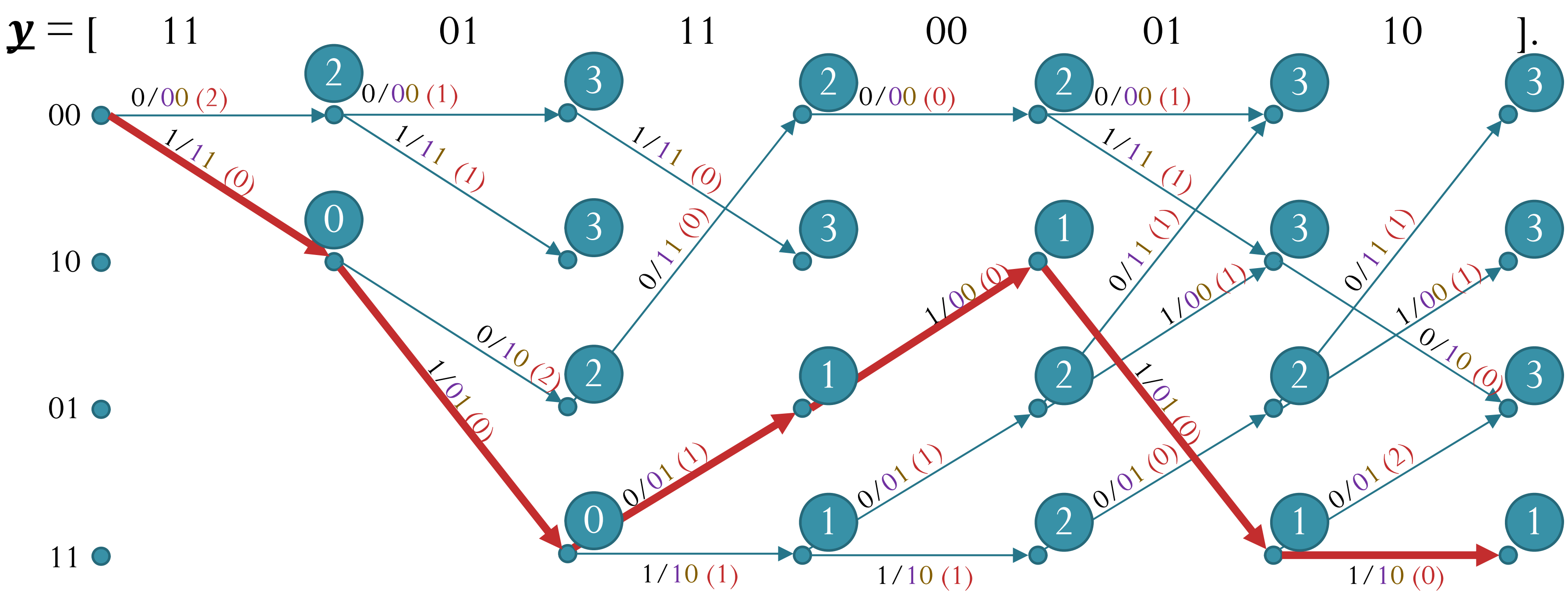
Viterbi Decoding

- Suppose $\underline{\boldsymbol{y}} = [11\ 01\ 11\ 00\ 01\ 10]$.
- Find $\hat{\underline{\mathbf{b}}}$.



Viterbi Decoding

- Suppose $\underline{y}$ = [11 01 11 00 01 10].
- Find $\hat{\underline{\mathbf{b}}}$.

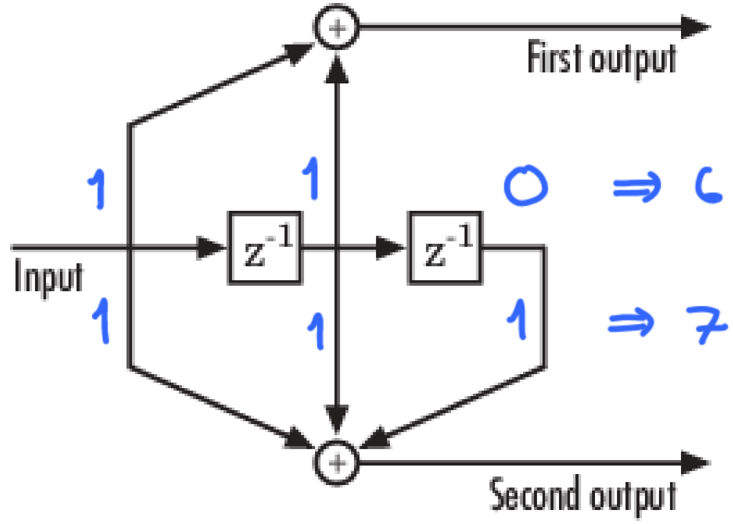


$\hat{\underline{\mathbf{x}}}$ = [11 01 01 00 01 10]

$\hat{\underline{\mathbf{b}}}$ = [1 1 0 1 1 1]

MATLAB: Generator Polynomial Matrix

- Build a binary number representation by placing a 1 in each spot where a connection line from the shift register feeds into the adder, and a 0 elsewhere.
  - The leftmost spot in the binary number represents the current input, while the rightmost spot represents the oldest input that still remains in the shift register.
- Convert this binary representation into an **octal representation**.
  - by considering consecutive triplets of bits
  - For example, interpret 1101010 as 001 101 010 and convert it to 152.
  - `str2num(dec2base(bin2dec('1101010'),8))`
- For example, the binary numbers corresponding to the upper and lower adders in the figure here are 110 and 111, respectively.
  - These binary numbers are equivalent to the octal numbers 6 and 7, respectively,
  - so the generator polynomial matrix is [6 7].

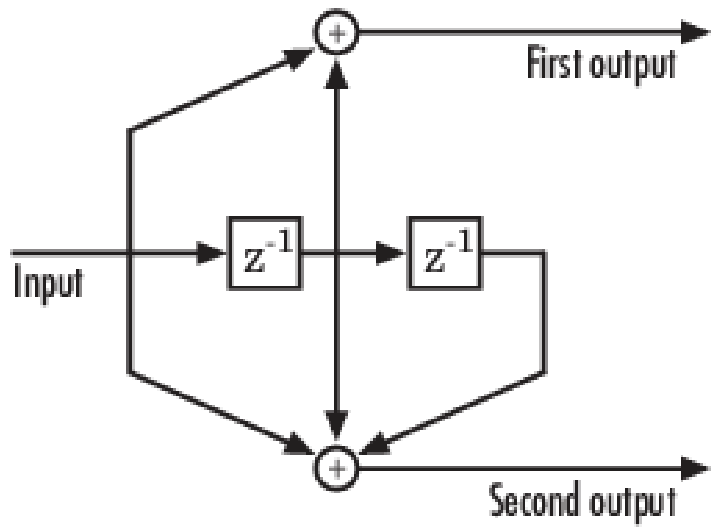


MATLAB:

- To use the polynomial description with the functions **`convenc`** and **`vitdec`**, first convert it into a trellis description using the **`poly2trellis`** function.
- For example,

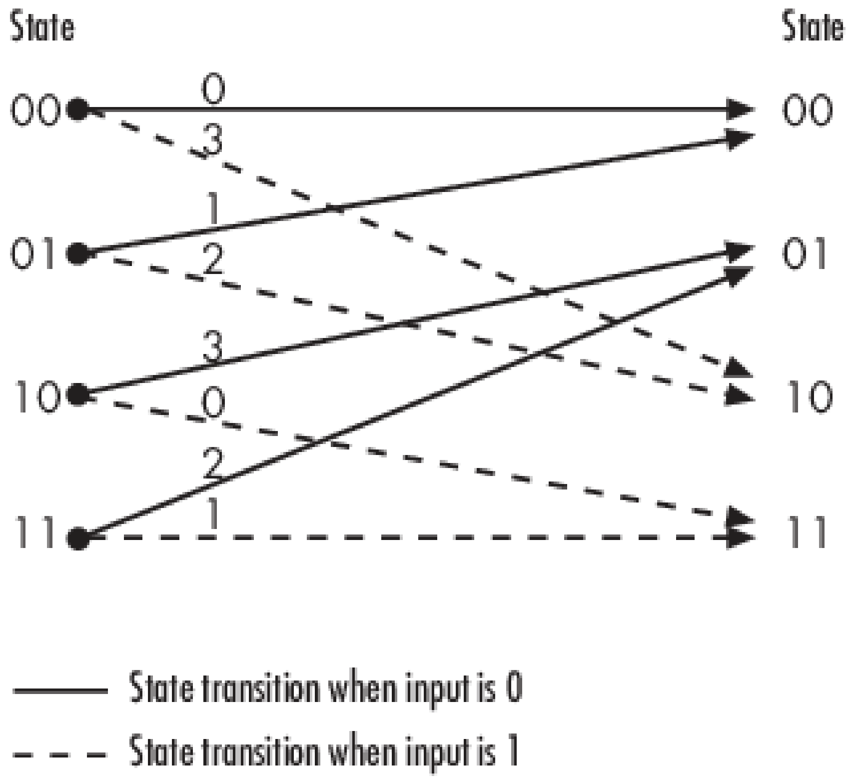
```
trellis = poly2trellis(3,[6 7]);
```

Constraint Length = #FFs + 1



MATLAB: Trellis Description

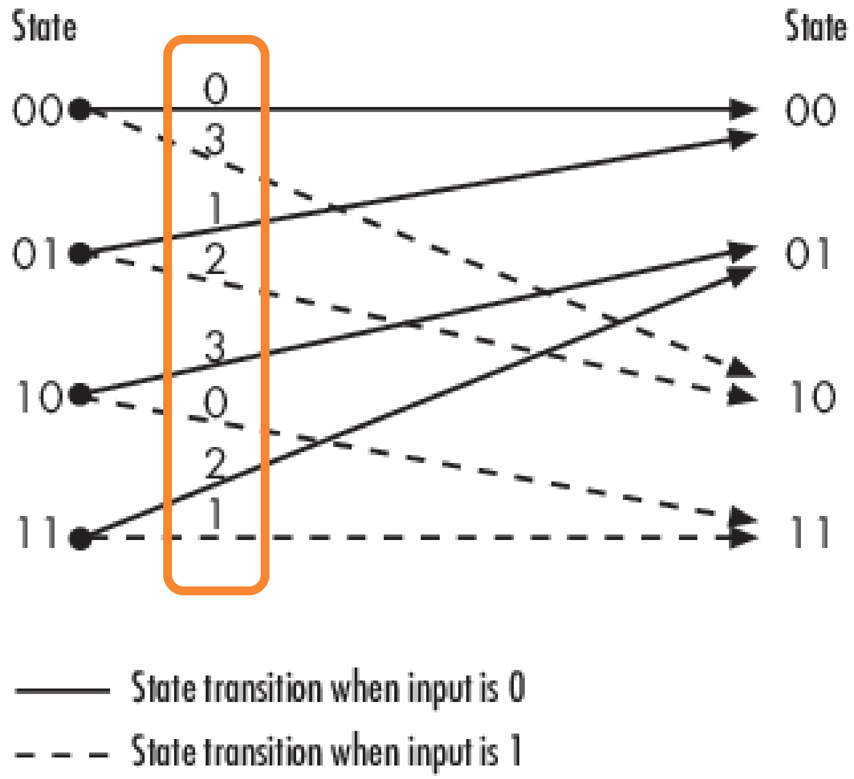
- Each solid arrow shows how the encoder changes its state if the current input is zero.
- Each dashed arrow shows how the encoder changes its state if the current input is one.
- The octal numbers above each arrow indicate the current output of the encoder.



MATLAB: Trellis Structure

2^k 2^n

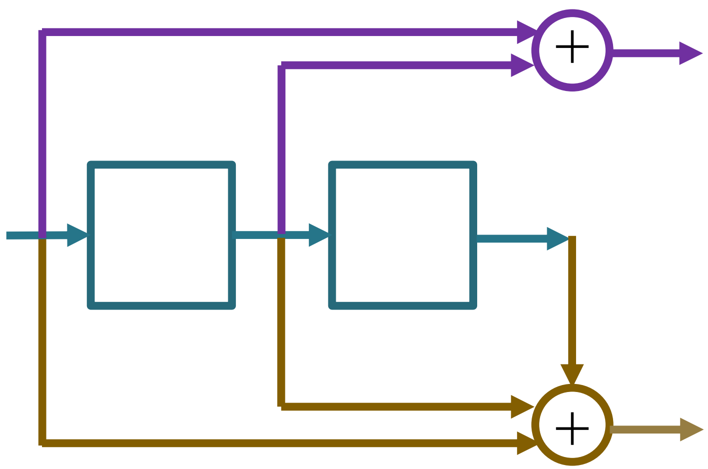
```
trellis = struct('numInputSymbols',2,'numOutputSymbols',4,...
'numStates',4,'nextStates',[0 2;0 2;1 3;1 3],...
'outputs',[0 3;1 2;3 0;2 1]);
```



MATLAB : Convolutional Encoding

- `x = convenc(b,trellis);`

Example (from the exercise)



```
trellis = poly2trellis(3,[6 7]);
b = [1 0 1 1 0];
x = convenc(b,trellis)
```

[ConvCode_Exer.m]

```
>> ConvCode_Exer
x =
     1     1     1     1     1     0     0     0     1     0
```

Reference

- Chapter 15 in [Lathi & Ding, 2009]
- Chapter 13 in [Carlson & Crilly, 2009]
- Section 7.11 in [Cover and Thomas, 2006]