ELEC 405/511
Error Control Coding

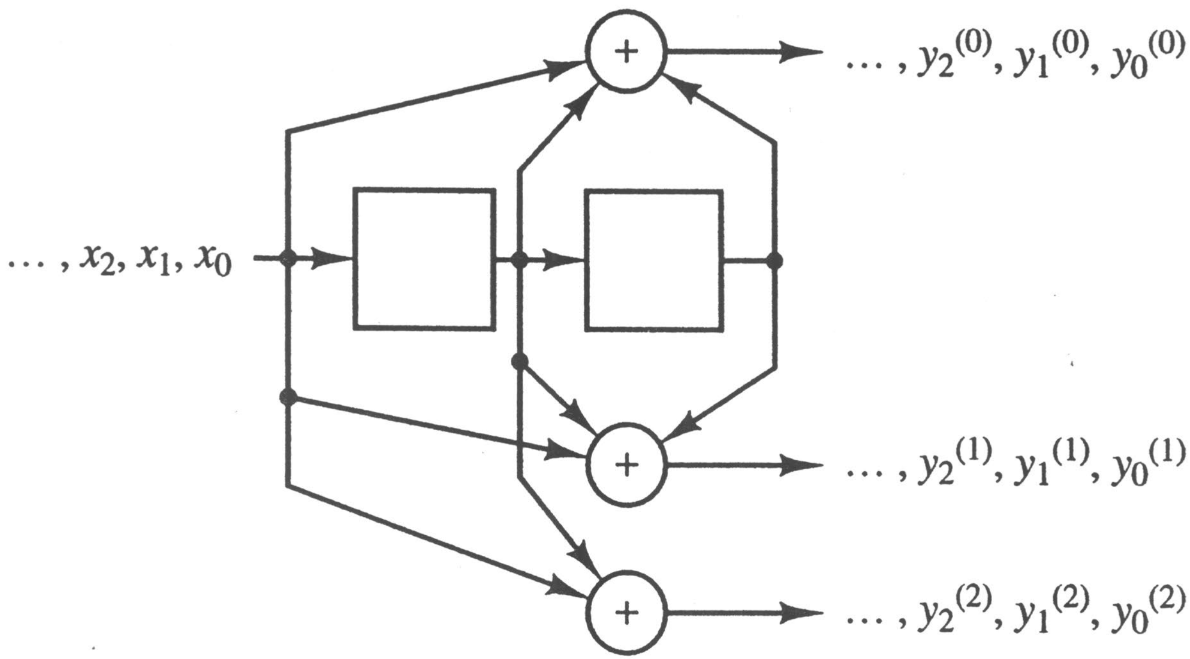
Decoding Convolutional Codes

Trellis Diagrams

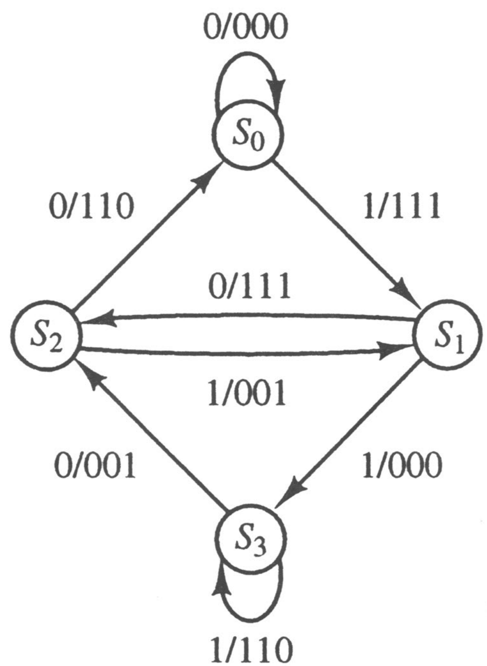
- A trellis diagram is an extension of the state diagram of a convolutional code that explicitly shows the time index
- Consider a rate 1/3 convolutional code with two memory cells

$$G(D) = [1+D+D^2 \quad 1+\mathrm{D}+D^2 \quad 1+D]$$

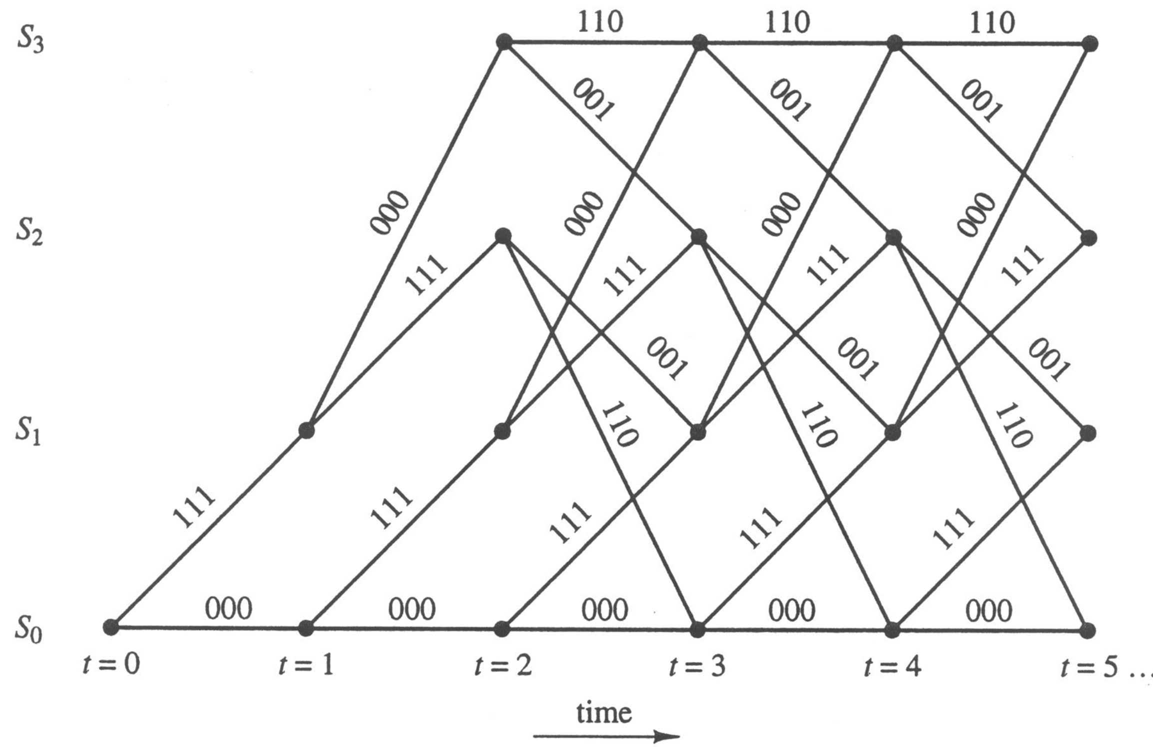
Encoder for a (3,1,3) Convolutional Code



State Diagram



Trellis Diagram



Codewords and the Trellis

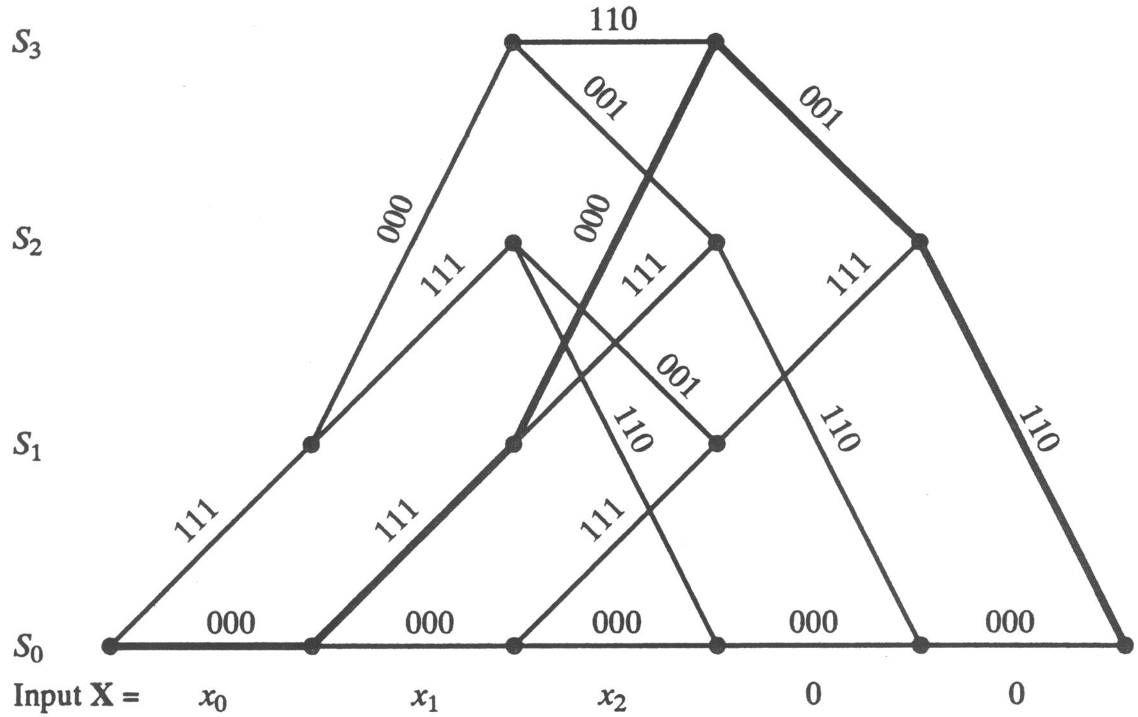
- Every path through the trellis is a codeword which starts and stops in state S_0
- Consider an (n,k) convolutional code with
  - total memory M and memory order m
- There are 2^M nodes at each stage corresponding to the states in the state diagram
- There are 2^k branches leaving each node, one for each combination of input values
- After time $t = m$, there are also 2^k branches entering each node

- After the input sequence of kL bits, m state transitions are required to return to the all-zero state
- The trellis has $L+m$ stages
- There are 2^{kL} paths through the trellis
- Each path has length $n(L+km)$

Rate 1/3 Code Example

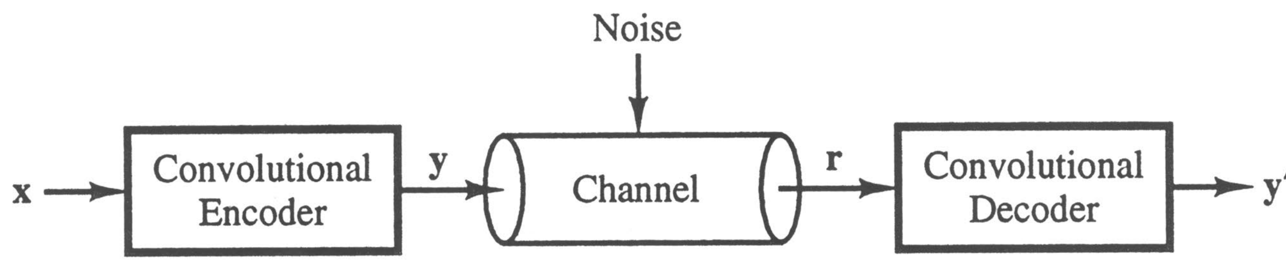
- Input **x** = (011)
- L+m = 3+2 = 5 trellis stages
- Codeword length $n(L+km)$ = 3(3+2) = 15
- Output **y** = (000,111,000,001,110)

Trellis Path for Input **x** = (011)



Codeword **y** = (000,111,000,001,110)
Length 3(3+2) = 15 bits

The Convolutional Decoding Problem



- The maximum likelihood (ML) decoder selects the estimate **y**’ that maximizes the probability p(**r**|**y**’)
- The maximum a posteriori (MAP) decoder selects the estimate that maximizes p(**y**’|**r**)
- If the source words **x** are uniformly distributed, the two decoders are identical
  - they are related through Bayes’ rule

- Likelihood function for **y**' assuming a memoryless channel

$$p(\mathbf{r} \mid \mathbf{y}') = \prod_{i=0}^{L+m-1} \left(\prod_{j=0}^{n-1} p(r_i^{(j)} \mid y'^{(j)}_i) \right)$$

- The log likelihood function is

$$\log p(\mathbf{r} \mid \mathbf{y}') = \sum_{i=0}^{L+m-1} \left(\sum_{j=0}^{n-1} \log p(r_i^{(j)} \mid y'^{(j)}_i) \right)$$

- For implementation, use

$$M(r_i^{(j)} \mid y'^{(j)}_i) = a\left[\log p(r_i^{(j)} \mid y'^{(j)}_i) + b\right]$$

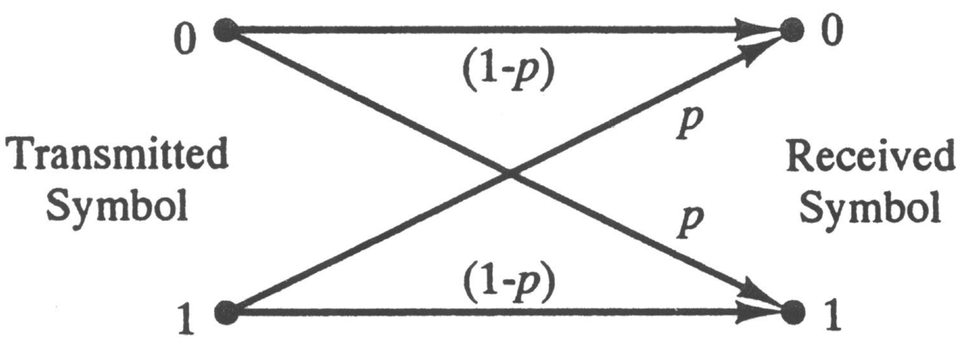
  - choose *a* and *b* such that the bit metrics are small positive integers

- The path metric is then

$$M(\mathbf{r} \mid \mathbf{y}') = \sum_{i=0}^{L+m-1} \left(\sum_{j=0}^{n-1} M(r_i^{(j)} \mid y'^{(j)}_i) \right)$$

Hard Decision Decoding

- Consider a binary symmetric channel
- Errors affect 0s and 1s with equal probability (symmetric)
- Errors occur randomly and are independent from bit to bit (memoryless)



p is the probability of bit error - crossover probability

Hard Decision Decoding

- Let $a = [\log_2 p - \log_2(1-p)]^{-1}$ and $b = -\log_2(1-p)$

$$M(r_i^{(j)} \mid y_i^{(j)}) = \frac{1}{\log_2 p - \log_2(1-p)}\left[\log p(r_i^{(j)} \mid y_i^{(j)}) - \log_2(1-p)\right]$$

- The metric table is then

$M(r_i^{(j)},y_i^{(j)})$	$r_i^{(j)} = 0$	$r_i^{(j)} = 1$
$y_i^{(j)} = 0$	0	1
$y_i^{(j)} = 1$	1	0

- The path metric is the Hamming distance $d(\mathbf{r},\mathbf{y})$
- The goal is to minimize the path metric

- If $a = [\log_2(1\text{-}p) - \log_2(p)]^{-1}$ and $b = -\log_2(p)$ the metric table becomes

$M(r_i^{(j)}, y_i^{(j)})$	$r_i^{(j)} = 0$	$r_i^{(j)} = 1$
$y_i^{(j)} = 0$	1	0
$y_i^{(j)} = 1$	0	1

- The path metric is the number of places where **r** and **y** are the same
- The goal is to maximize the path metric

Branch Metrics

- The *k*th branch metric for a codeword **y'** is defined as the sum of the bit metrics for the *k*th block of **r** given **y'**

$$M(r_k \mid y'_k) = \sum_{j=0}^{n-1} M(r_k^{(j)} \mid y'^{(j)}_k)$$

- The *k*th partial path metric for a path is obtained by summing the branch metrics for the first *k* branches that the path traverses

$$M^k(\mathbf{r} \mid \mathbf{y}') = \sum_{i=0}^{k-1} M(\mathbf{r}_i \mid \mathbf{y}'_i) = \sum_{i=0}^{k-1} \left(\sum_{j=0}^{n-1} M(r_i^{(j)} \mid y'^{(j)}_i) \right)$$

Decoding Approach

- ML decoding requires choosing the codeword **y**' which is closest to **r**
- How to do this practically?
- What is required is a path through the trellis
  - After *m* time steps, there are 2^k paths entering any node or state
  - At this point, a decision can be made as to which of these paths is closer to **r** up to this point in time
  - So at each time instant, 2^M survivor paths must be chosen
  - To make these decisions, the partial path metrics provide the distance between the
    - output bits corresponding to a given trellis path and the
    - corresponding bits from **r**
  - The ML path must be contained in one of the survivor paths at any time instant

The Viterbi Algorthm

- Let the node corresponding to state S_j at time t be denoted as $S_{j,t}$
- Each node in the trellis is to be assigned a value $V(S_{j,t})$
- The node values are computed as follows
  1. Set $V(S_{0,0}) = 0$ and $t = 1$
  2. At time t, compute the partial path metrics for all paths entering each node
  3. Set $V(S_{j,t})$ equal to the best partial path metric entering the node corresponding to state S_j at time t. Ties are broken randomly. Nonsurviving branches are ignored.
  4. If $t < L+m$, increment t and return to step 2
- Once all node values have been computed, start at state S_0, time $t = L+m$, and follow the surviving branches backward through the trellis. The resulting unique path is the ML codeword.

Example 12-2

- Using the encoder given previously, consider the input sequence **x** = (110101)
- The codeword is

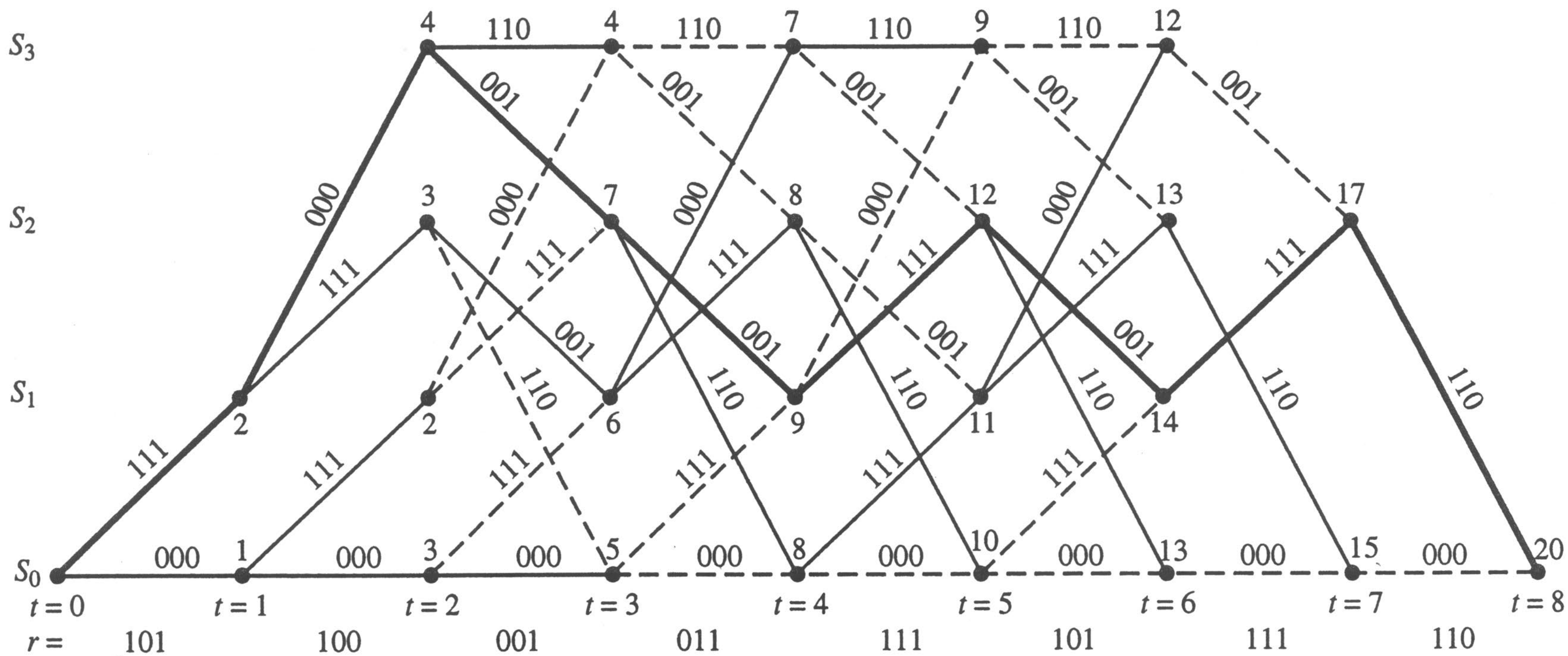
  y = (111,000,001,001,111,001,111,110)
- The received word is

  r = (1**0**1,**1**00,001,0**1**1,111,**1**01,111,110)
- The metric is the number of places the branch values are the same
  - survivors are the paths with the maximum metrics

Example 12-2 (Cont.)

- After $t = 2$, there are 2 paths entering each state
- The path with the largest partial metric is the survivor
- The last surviving path at time $t = L+m$ is the ML codeword

Trellis for Example 12-2



Example 12-3

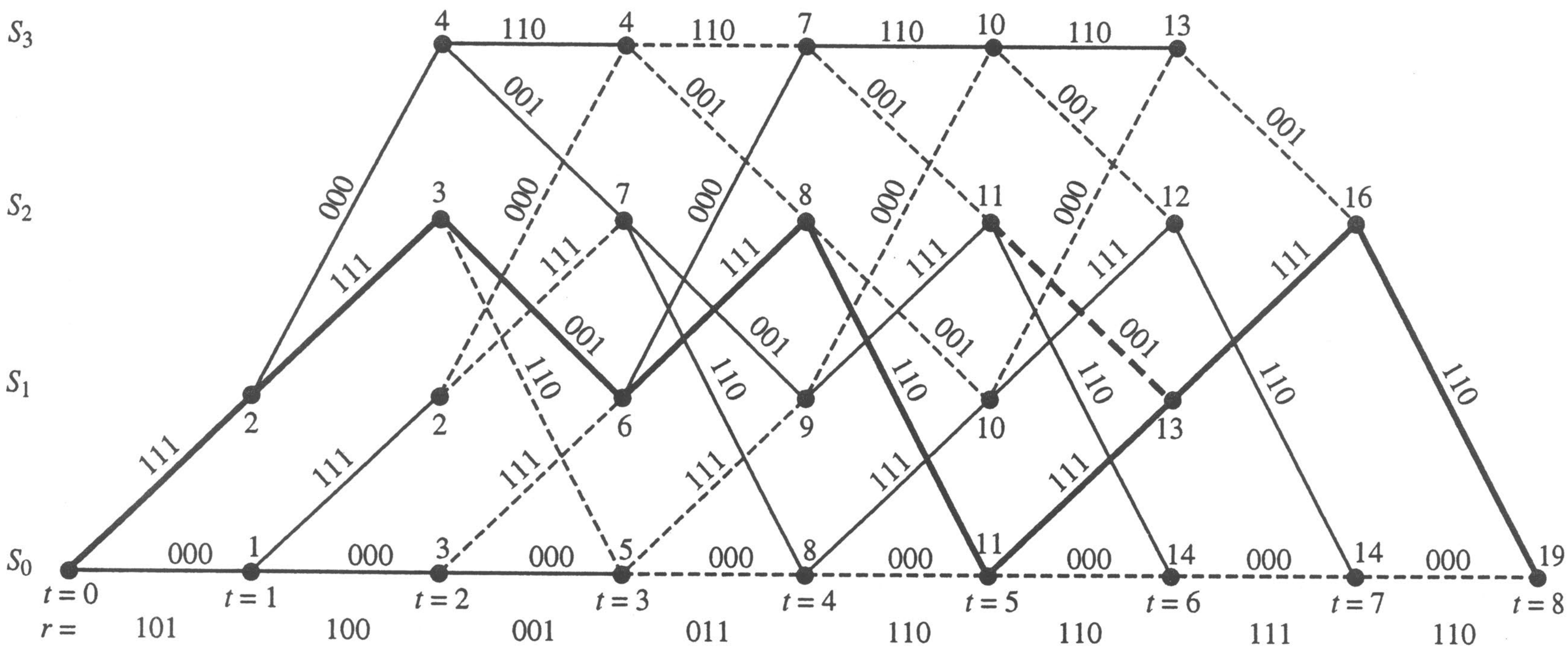
- Using the same encoder and input sequence **x** = (110101) with codeword

  y = (111,000,001,001,111,001,111,110)

  the received word is now

  r = (1**0**1,**1**00,001,0**1**1,11**0**,**110**,111,110)

Trellis for Example 12-3



Example 12-3 (Cont.)

- In this case the ML codeword is **not** the transmitted codeword
- At node $S_{1,6}$, the transmitted path (shown as a bold dotted line) does not survive
- The ML word is

  y' = (111,**111**,001,**11**1,11**0**,**11**1,111,110)

  and the corresponding data word is

  x' = (1**010**01)

  so a burst of 4 errors in the transmitted codeword has caused 3 errors in the decoded data

Soft Decision Decoding

- In soft decision decoding, the receiver takes advantage of additional information from the demodulator about the reliability of the bit decisions
- More reliable decisions should have a greater effect on decoding
- On an AWGN channel, soft decision decoding can provide 2-3 dB more coding gain
  - In fading channels, up to 9 dB can be gained

Decoding with BPSK Modulation

- Consider BPSK modulation over an AWGN channel
- The received energy per bit is E_b
- The noise PSD is N_0
- The transmitted bits take the values ±1
- The received signals are Gaussian random variables with mean $y_i^{(j)}\sqrt{E_b}$ and variance $N_0/2$
- The likelihood functions then have a Gaussian PDF

$$\mathrm{p}(r_i^{(j)} \mid y_i^{(j)}) = \frac{1}{\sqrt{\pi N_0}} e^{-\left(r_i^{(j)} - y_i^{(j)}\sqrt{E_b}\right)^2 / N_0}$$

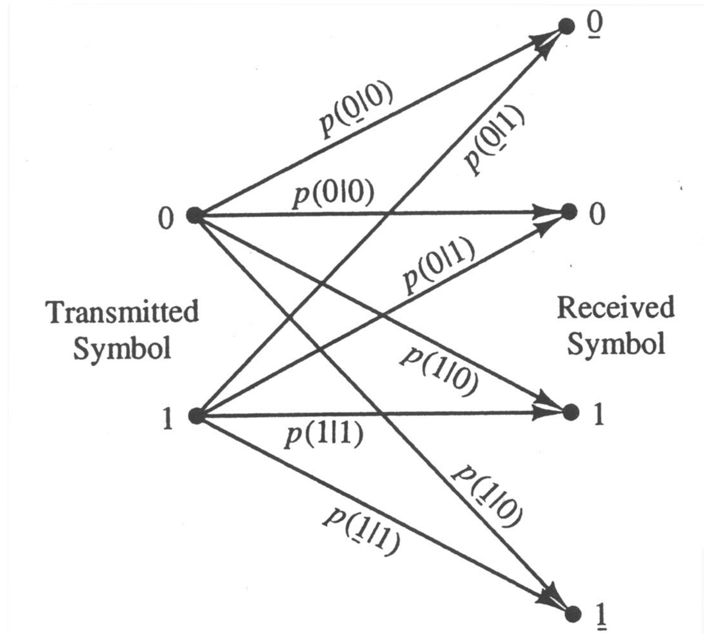
BPSK Log Likelihood Function

- After some manipulation
  log p($\mathbf{r}|\mathbf{y}$) = $C_1(\mathbf{r}\cdot\mathbf{y}) + C_2$
  where C_1 and C_2 are constants containing terms that are not a function of **y**
- The individual bit metrics are

$$M\left(r_i^{(j)} \mid y_i^{(j)}\right) = r_i^{(j)} y_i^{(j)}$$

- Thus minimizing the path metric is equivalent to finding the codeword **y** that is closest to **r** in terms of Euclidean distance

Discrete Symmetric Channel



- The receiver assigns one of 4 values to each received bit
- An underlined zero or one indicate a strong signal
- Nonunderlined denotes the reception of a weaker value

Example 12-4

- Consider the following conditional probabilities

$p(r\|y)$	$r=\underline{0}$	$r=0$	$r=1$	$r=\underline{1}$
$y=0$	$.50$	$.32$	$.13$	$.05$
$y=1$	$.05$	$.13$	$.32$	$.50$

- The corresponding log likelihood functions are

$\log_2 p(r\|y)$	$r=\underline{0}$	$r=0$	$r=1$	$r=\underline{1}$
$y=0$	-1.00	-1.64	-2.94	-4.32
$y=1$	-4.32	-2.94	-1.64	-1.00

- Using a = 1.5 and b = $-\log_2(0.05)$ gives

$$M(r \mid y) = 1.5\left[\log_2 \mathrm{p}(r \mid y) - \log_2(0.05)\right]$$

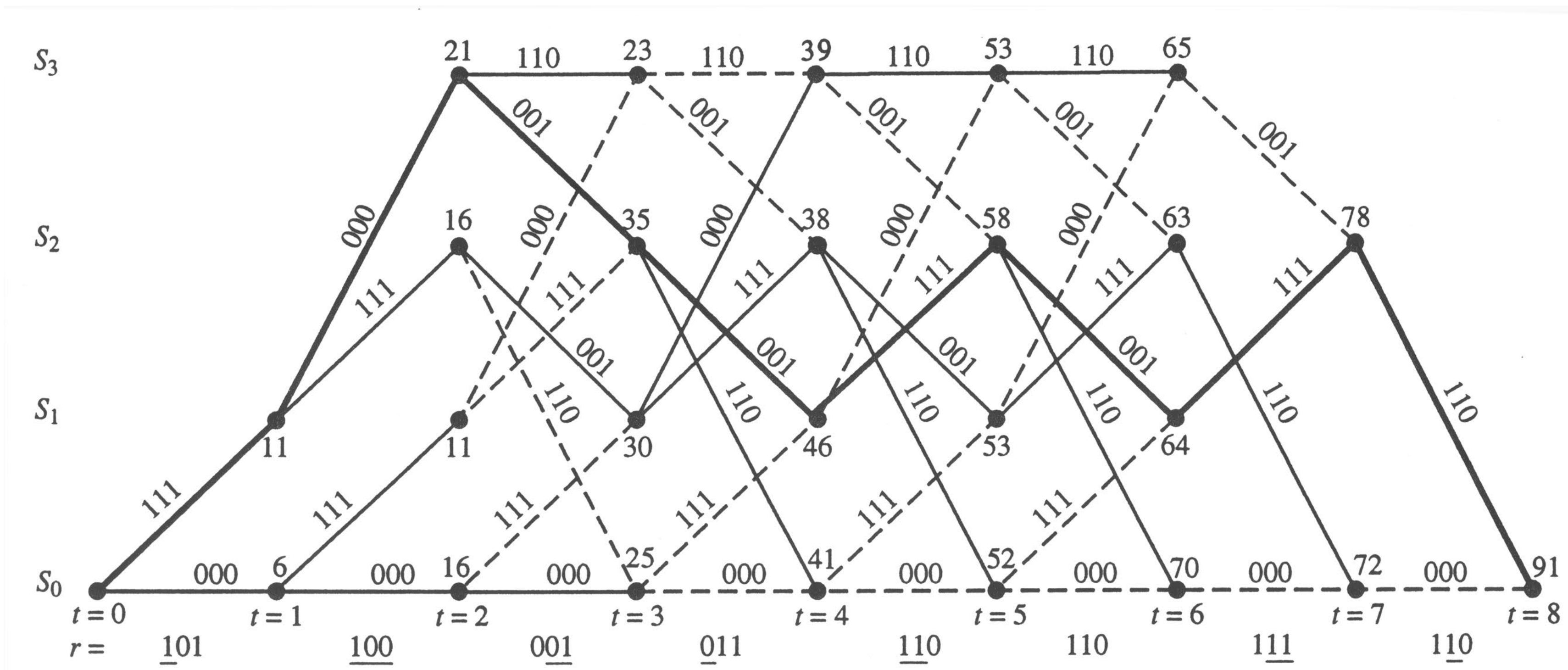
$M(r\|y)$	$r = \underline{0}$	$r = 0$	$r = 1$	$r = \underline{1}$
$y = 0$	5	4	2	0
$y = 1$	0	2	4	5

- Now as before let

  $\mathbf{y}$ = (111,000,001,001,111,001,111,110)
- With soft decision information

  $\mathbf{r} = (\underline{1}\mathbf{0}1, \underline{\mathbf{1}}\underline{0}\underline{0}, 0\underline{0}\underline{1}, \underline{0}\mathbf{1}1, \underline{1}\,\underline{1}0\mathbf{0}, \mathbf{110}, 1\underline{1}\,\underline{1}, 1\underline{1}0)$
- Note that only one of the erroneous bits is strong (underlined)

Soft Decision Viterbi Decoding



- Unlike the case with hard decision decoding, the ML decision is the correct codeword

  $\mathbf{y}$ = (111,000,001,001,111,001,111,110)

ECE 405/511 Final Exam

- Counts for 30% of the final grade
- Covers all course material up to and including convolutional codes (not weight enumerators)
  - Gazi Chapters 1 to 8
  - Wicker Chapters 5, 8, 9, 11 are available on Brightspace
- Aids allowed: 2 sheets of paper 8.5 × 11 in^2

  Calculator