

# Introduction into Error Correcting Codes

Prof. Dr.-Ing. Volker Kühn  
Institute of Communications Engineering  
Phone: 0381/498-7330, Room: W 8233  
Email: [volker.kuehn@uni-rostock.de](mailto:volker.kuehn@uni-rostock.de)

<http://www.int.uni-rostock.de/>

- Lesson 1: One Lesson of Information Theory
  - Principle structure of communication systems
  - Definitions of entropy, mutual information, ...
  - Channel coding theorem of Shannon
  
- Lesson 2: Introduction to Error Correcting Codes
  - Basics of error correcting codes
  - Linear block codes
  - Convolutional codes
  
- Lesson 3: State-of-the-art channel coding
  - Coding strategies to approach the capacity limits
  - Definition of soft-information and turbo decoding principle
  - Examples for state-of-the-art error correcting codes

## **Basics of error correcting codes**

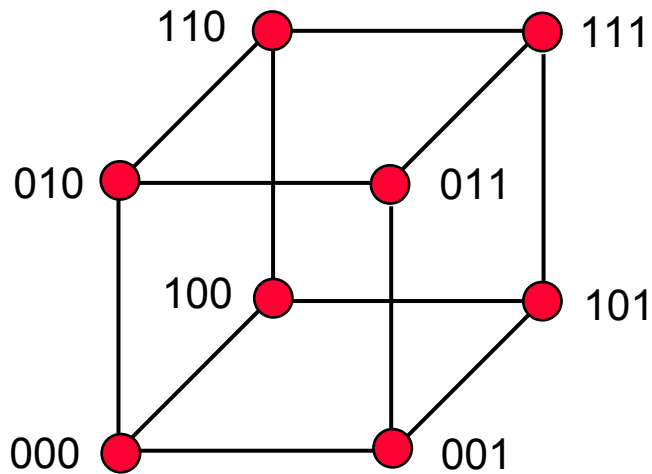
**Linear block codes**

**Convolutional codes - if time permits**

- **Source coding** (entropy coding)
  - Aims to represent a message with as few symbols as possible (data compression)
  - Digitizing analog data like audio and video signals
- **Channel Coding:**
  - Encoder adds redundancy (additional bits) to information in order to detect or even correct transmission errors
  - Joint source and channel coding possible
- **Cryptography**
  - Keeping messages private (security)

# Visualizing Channel Coding Principle

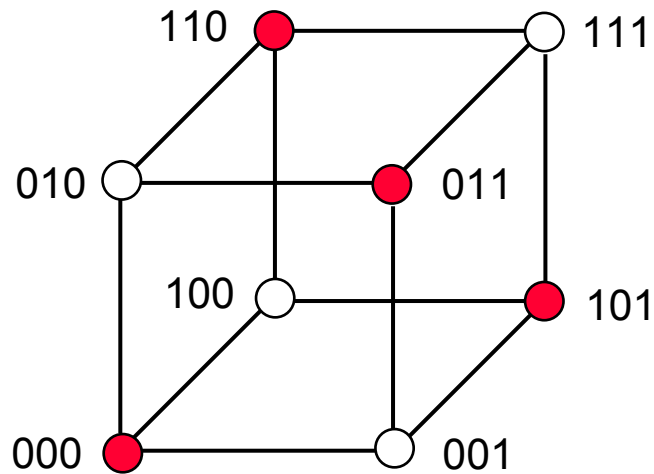
all vertices used



$d_{\min}=1$

- ◆ Code rate  $R_c = 1$
- ◆ No error correction
- ◆ No error detection

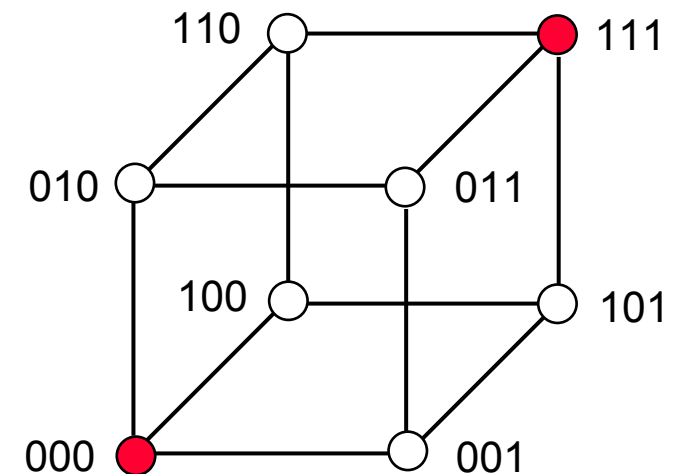
only 4 vertices used



$d_{\min}=2$

- ◆ Code rate  $R_c = 2/3$
- ◆ No error correction
- ◆ Detection of single error

only 2 vertices used

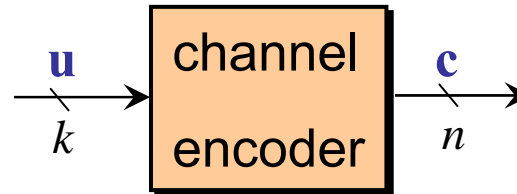


$d_{\min}=3$

- ◆ Code rate  $R_c = 1/3$
- ◆ Correction of single error
- ◆ Detection of 2 errors

- Forward Error Correction (FEC)
  - Correction of transmission errors at decoder
  - Advantage: low latency, less redundancy for intermediate channels
  - Drawbacks: too much redundancy for good channels, not enough redundancy for really bad channels,
  - **Throughput is constant, reliability depends on channel**
  
- Automatic Repeat Request (ARQ)
  - Decoder indicates erroneous message over feedback channel
  - Retransmission of erroneous packets
  - Advantage: redundancy adapted to channel quality
  - Drawback: lower throughput for intermediate channels
  - **Throughput depends on channel quality, reliability is constant**
  
- Hybrid schemes combine advantages of both approaches

- Importance of channel coding increased with digital communications
- First use for deep space communications:
  - AWGN channel, no bandwidth restrictions, only few receivers (costs negligible)
  - Examples:  
Viking (Mars), Voyager (Jupiter, Saturn), Galileo (Jupiter), ...
- Mass storage
  - hard disc, compact disc, digital versatile disc, magnetic tapes
- Digital wireless communications:
  - GSM, UMTS (HSDPA, HSUPA), WLAN, LTE, WIMAX, ...
- Digital wired communications
  - Modem transmission (V.90, ...)
- Digital broadcasting
  - DAB, DVB, DRM



- **Code**  $\Gamma = \{c_0, c_1, \dots, c_{2^k-1}\}$ : set of code words
- **Encoder**: maps information word **u** onto code word **c** by adding redundancy
  - Code rate  $R_c = k / n$
  - **Systematic encoder**: codeword **c** explicitly contains information word **u**

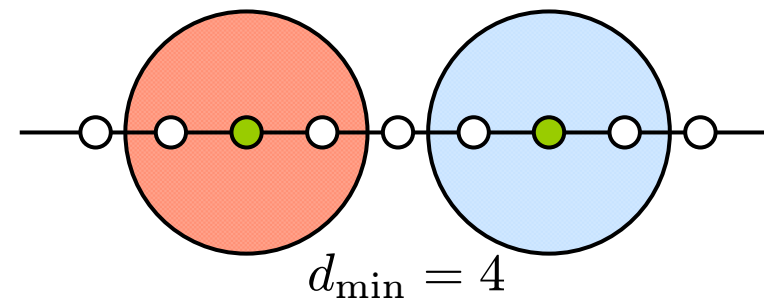
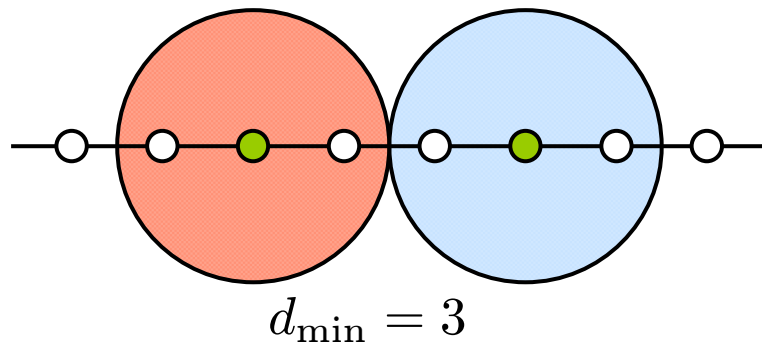


- **Non-systematic encoder**: information word **u** is not explicit part of **c**

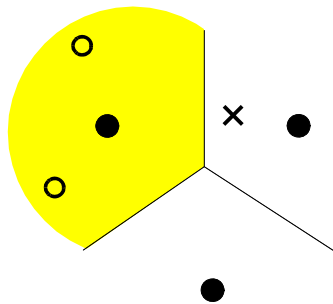
- Hamming distance  $d_H(\mathbf{c}_1, \mathbf{c}_2)$ : number of different symbols between 2 code words  $\mathbf{c}_1$  and  $\mathbf{c}_2$
- Hamming weight  $w_H(\mathbf{c}_1)$ : number of non-zero symbols in a code word
- Minimum Hamming distance  $d_{\min}$  determines capabilities of code

– Number of detectable errors:  $t' = d_{\min} - 1$

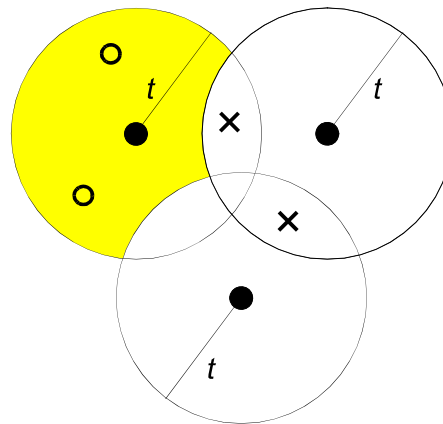
– Number of correctable errors:  $t = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor$



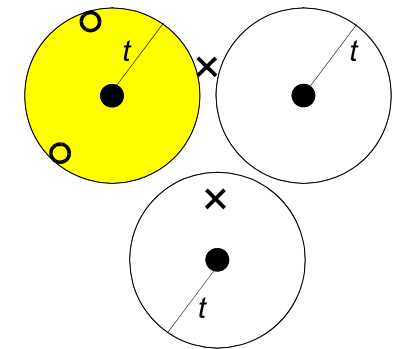
MLD:  
Maximum  
Likelihood  
Decoding



BDD:  
Bounded  
Distance  
Decoding



BMD:  
Bounded  
Minimum Distance  
Decoding



- correctable
- × not correctable
- code words

- Maximum likelihood principle:

$$\hat{\mathbf{x}} = \underset{\tilde{\mathbf{x}}}{\operatorname{argmax}} p(\mathbf{y} | \tilde{\mathbf{x}}) = \underset{\tilde{\mathbf{x}}}{\operatorname{argmax}} \log [p(\mathbf{y} | \tilde{\mathbf{x}})]$$

- Pairwise error probability for AWGN:

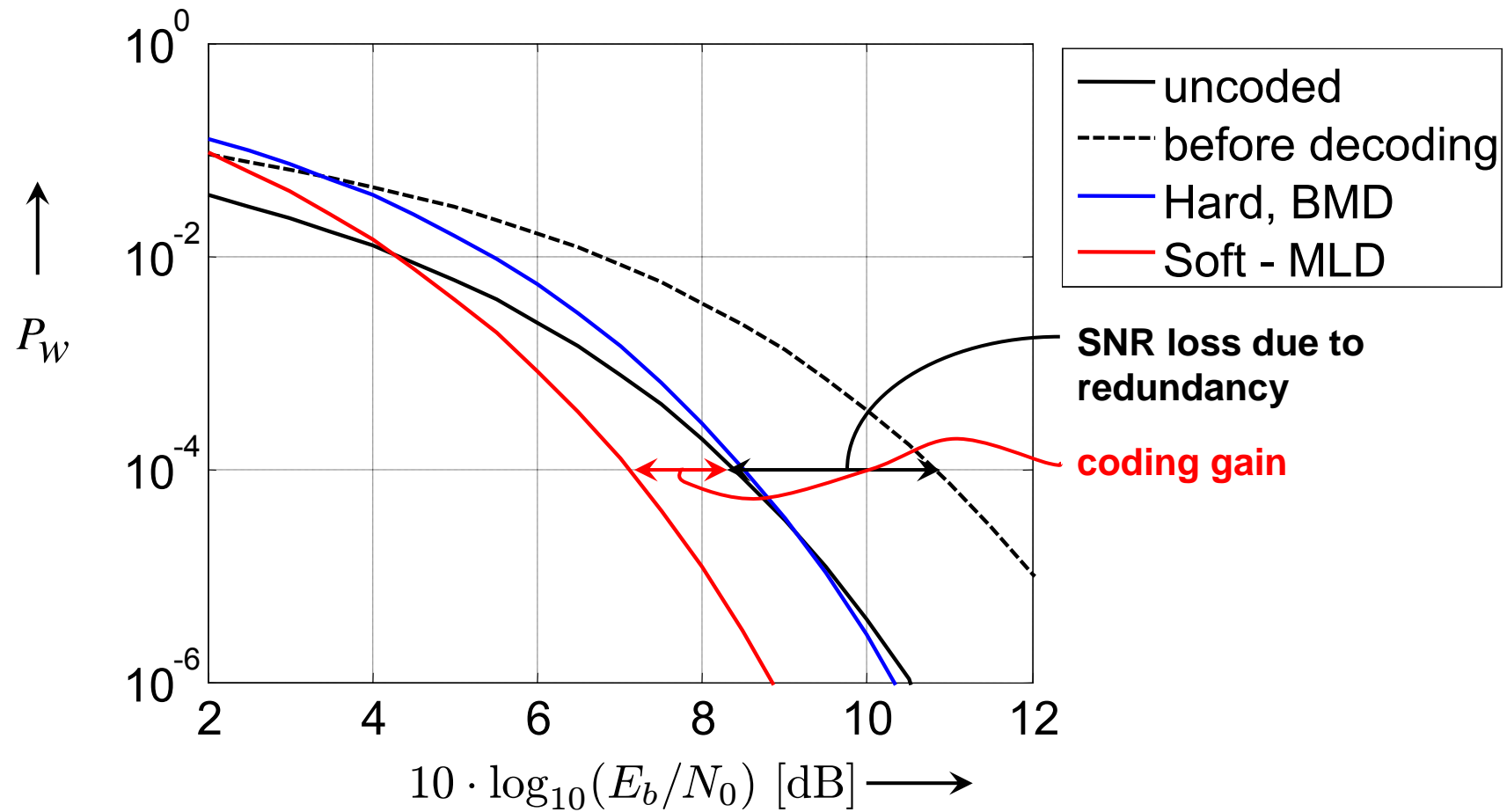
$$\Pr \{ \mathbf{x} \leftrightarrow \tilde{\mathbf{x}} | \mathbf{y} \} = \Pr \{ p(\mathbf{y} | \tilde{\mathbf{x}}) > p(\mathbf{y} | \mathbf{x}) \} = \Pr \{ \|\mathbf{y} - \tilde{\mathbf{x}}\|^2 < \|\mathbf{y} - \mathbf{x}\|^2 \}$$

- Average error probability with Maximum Likelihood Decoding:

$$\begin{aligned} P_w &\leq \frac{1}{2} \cdot \sum_{\substack{j=1 \\ j \neq i}}^{2^k} \operatorname{erfc} \left( \sqrt{d_{\text{H}}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \cdot E_s / N_0} \right) \\ &= \frac{1}{2} \cdot \sum_{d=d_{\min}}^n A_d \cdot \operatorname{erfc} \left( \sqrt{d \cdot E_s / N_0} \right) \end{aligned}$$

- Remember uncoded error probability:  $P_s = \frac{1}{2} \cdot \operatorname{erfc} \left( \sqrt{\frac{E_s}{N_0}} \right)$

- Error Probability for (7,4)-Hamming Code



Basics of error correcting codes

**Linear block codes**

**Convolutional codes - if time permits**

- Each row contains valid code word

$$\mathbf{G} = \begin{bmatrix} g_{0,0} & \cdots & g_{0,n-1} \\ \vdots & \ddots & \vdots \\ g_{k-1,0} & \cdots & g_{k-1,n-1} \end{bmatrix}$$

rows are linear independent  
they span the code (vector space)

- Encoding:  $\mathbf{x} = \mathbf{u} \cdot \mathbf{G} \bmod 2$

in the sequel, all calculations  
are carried out mod 2

- Code:  $\Gamma = \{ \mathbf{x} = \mathbf{u}\mathbf{G} \bmod 2 \mid \mathbf{u} \in \text{GF}(2)^k \}$

- Systematic encoder ( $\mathbf{u}$  explicitly part of  $\mathbf{x}$ ) ( $\mathbf{x} = [\mathbf{u} \quad \mathbf{p}]$ )

$$\mathbf{G} = [\mathbf{I}_k \mid \mathbf{P}_{k \times n-k}] = \left[ \begin{array}{ccc|c} 1 & & 0 & \\ & \ddots & & \\ 0 & & 1 & \mathbf{P}_{k \times n-k} \end{array} \right]$$

- Parity check matrix:

$$\mathbf{H} = \begin{bmatrix} h_{0,0} & \cdots & h_{0,n-1} \\ \vdots & \ddots & \vdots \\ h_{n-k-1,0} & \cdots & h_{n-k-1,n-1} \end{bmatrix}$$

rows are linear independent  
they span an orthogonal vector space

- $\mathbf{H}$  for systematic encoder

$$\mathbf{H} = \left[ -\mathbf{P}_{k \times n-k}^T \mid \mathbf{I}_{n-k} \right] = \left[ \begin{array}{c|ccc} & & 1 & 0 \\ & & & \vdots \\ -\mathbf{P}_{k \times n-k}^T & & 0 & 1 \end{array} \right]$$

- Generator and parity check matrix:

$$\mathbf{G} \cdot \mathbf{H}^T = \left[ \mathbf{I}_k \quad \mathbf{P}_{k \times n-k} \right] \cdot \left[ -\mathbf{P}_{k \times n-k}^T \quad \mathbf{I}_{n-k} \right]^T = \left[ \mathbf{I}_k \quad \mathbf{P}_{k \times n-k} \right] \cdot \begin{bmatrix} -\mathbf{P}_{k \times n-k} \\ \mathbf{I}_{n-k} \end{bmatrix} = \mathbf{0}$$

- Parity check equation:

$$\mathbf{x} \cdot \mathbf{H}^T \bmod 2 = \mathbf{0} \quad \Rightarrow \quad \Gamma = \{ \mathbf{x} = \text{GF}(2)^n \mid \mathbf{x} \cdot \mathbf{H}^T \bmod 2 = \mathbf{0} \}$$

# Examples of Codes (1)

- Repetition code:  $(n, 1, n)$ -code

$$\mathbf{G} = \underbrace{\begin{bmatrix} 1 & | & 1 & \cdots & 1 \end{bmatrix}}_n \quad \mathbf{H} = \left. \begin{bmatrix} 1 & | & 1 & & 0 \\ \vdots & & & \ddots & \\ 1 & | & 0 & & 1 \end{bmatrix} \right\} n-1$$

1	1	1	1	1
0	0	0	0	0

- Single parity check (SPC) code:  $(n, n-1, 2)$ -code

$$\mathbf{G} = \left. \begin{bmatrix} 1 & & 0 & | & 1 \\ & \ddots & & & \vdots \\ 0 & & 1 & | & 1 \end{bmatrix} \right\} n-1 \quad \mathbf{H} = \underbrace{\begin{bmatrix} 1 & 1 & \cdots & 1 \end{bmatrix}}_n$$

1	1	0
1	0	1
0	1	1
0	0	0

- Repetition code is dual code of single parity check code  
 → generator and parity check matrix are exchanged

- Hamming code:

- The length of an  $(n, n-r, 3)$ -Hamming code of rank  $r$  is defined by  $n = (2^r - 1)$ .
- Hamming codes are perfect codes, i.e. the number of distinct syndromes equals the number of correctable errors.
- All Hamming codes have a minimum distance of  $d_H = 3$ , whereas the code rate tends to  $R_c = 1$  for  $n \rightarrow \infty$ .

$$R_c = \frac{k}{n} = \frac{2^r - 1 - r}{2^r - 1} = \frac{1 - 2^{-r} - r2^{-r}}{1 - 2^{-r}} \xrightarrow{r \rightarrow \infty} 1$$

- Columns of parity check matrix represent all  $2^r - 1$  possible binary vectors of length  $r$  (except the all-zero-vector)
- $(3, 1)$ ,  $(7, 4)$ ,  $(15, 11)$ ,  $(31, 26)$ ,  $(63, 57)$ ,  $(127, 120)$

- Hamming code:

- Example: (7, 4) - Hamming code

$$\mathbf{G} = \left( \begin{array}{cccc|ccc} 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{array} \right); \quad \mathbf{H} = \left( \begin{array}{cccc|ccc} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{array} \right)$$

- Perfect code: columns of  $\mathbf{H}$  represent all  $2^{n-k}-1$  syndromes (except  $\mathbf{0}$ )
- For appropriate  $\mathbf{H}$ : position of syndrome  $\mathbf{s}$  corresponds to position of error in  $\mathbf{y}$  → encoder is not systematic any more!

- Simplex code:

- Simplex code is obtained by exchanging  $\mathbf{G}$  and  $\mathbf{H}$  (dual codes)
- All code words have same pair-wise Hamming distance  $d_H = 4$  → simplex

- Assumption:  $\mathbf{y} = \mathbf{x} + \mathbf{e} \bmod 2$  is received vector

- Decoding by calculating syndrome  $\mathbf{s} = [s_0 \cdots s_{n-k-1}]$

$$\mathbf{s} = \mathbf{y} \cdot \mathbf{H}^T \bmod 2 = (\mathbf{x} + \mathbf{e}) \cdot \mathbf{H}^T \bmod 2 = \underbrace{\mathbf{x} \cdot \mathbf{H}^T \bmod 2}_{=0} + \mathbf{e} \cdot \mathbf{H}^T \bmod 2$$

- Syndrome only depends on error and not on transmitted code word

- Error detection: check on

- $\mathbf{s} = \mathbf{0} \rightarrow$  no detectable error
- $\mathbf{s} \neq \mathbf{0} \rightarrow$  error detected

- Error correction:  $2^n - 2^k$  possible errors  $\mathbf{e}$ , but only  $2^{n-k}$  possible syndromes  $\mathbf{s}$

- no one-to-one correspondence of syndromes and error patterns
- not every error is correctable

- Partition set of all vectors into cosets  $M_\mu$  of those words that generate same syndrome  $\mathbf{s}_\mu$  and store cosets in table
- Determine coset leader  $\mathbf{e}_\mu$ , e.g. word with minimum Hamming weight  $w_H(\mathbf{e}_\mu)$  for each coset

- Calculate syndrome of received word  $\mathbf{y}$ :

$$\mathbf{s} = \mathbf{y} \cdot \mathbf{H}^T \text{ mod } 2$$

- Determine coset leader  $\mathbf{e}_\mu$  corresponding to  $\mathbf{s}$

- Correct error by subtracting  $\mathbf{e}_\mu$  from received word  $\mathbf{y}$ :

$$\hat{\mathbf{x}} = \mathbf{y} - \mathbf{e}_\mu$$

# Example for Syndrome Decoding

- Syndrome decoding for (7,4)-Hamming code

- There exist only  $2^{7-4} = 8$  different syndromes, but  $2^7 - 2^4 = 128 - 16 = 112$  error patterns
- Due to  $d_{\min} = 3$  only  $t = 1$  error correctable
- Coset leader represented by 7 vectors with  $w_H(\mathbf{e}_\mu) = 1$

$$\mathbf{H} = \left[ \begin{array}{cccc|ccc} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{array} \right]$$

syndrome	coset leader
0 0 1	0 0 0 0 0 0 1
0 1 0	0 0 0 0 0 1 0
0 1 1	1 0 0 0 0 0 0
1 0 0	0 0 0 0 1 0 0
1 0 1	0 1 0 0 0 0 0
1 1 0	0 0 1 0 0 0 0
1 1 1	0 0 0 1 0 0 0

# Many more powerful codes exist ...

- Cyclic block codes
  - CRC codes (Cyclic Redundancy Check) for detection of long bursty errors
  - Nonbinary Reed-Solomon codes for correction of long bursty errors
  - Binary BCH (Bose Chauduri Hocquenhem) codes for correction of single errors
- Fire codes
- Reed-Muller codes
- Convolutional Codes

Basics of error correcting codes

Linear block codes

**Convolutional codes - if time permits**

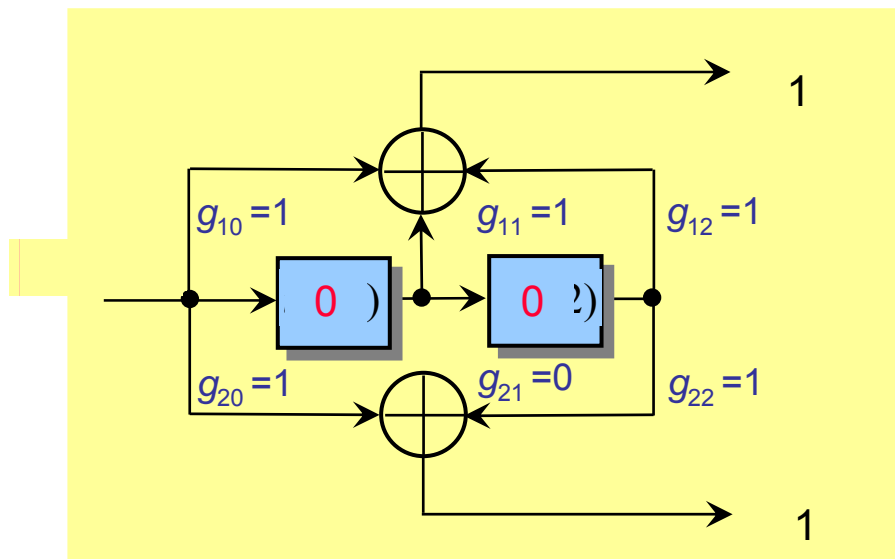
# Structure and Encoding

- Example: (2,1,3)-convolutional code with generators

$$g_1 = 7_8 \text{ and } g_2 = 5_8$$

- Encoder is non-systematic

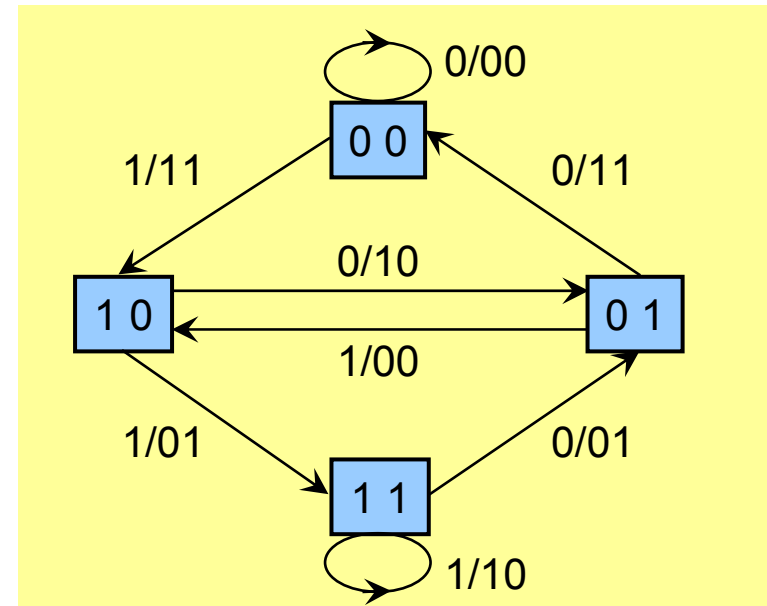
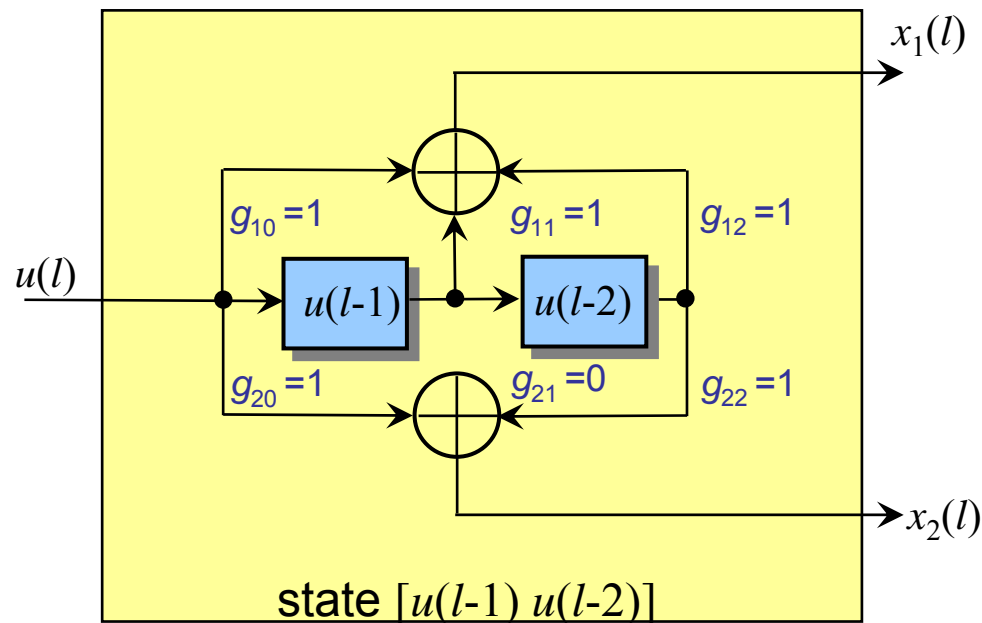
- $R_c = 1/2, L_c = 3 \rightarrow m = 2$



$u(l)$	state	following state	output
1	00	10	11
0	10	01	10
0	01	00	11
1	00	10	11
1	10	11	01
0	11	01	01
0	01	00	11

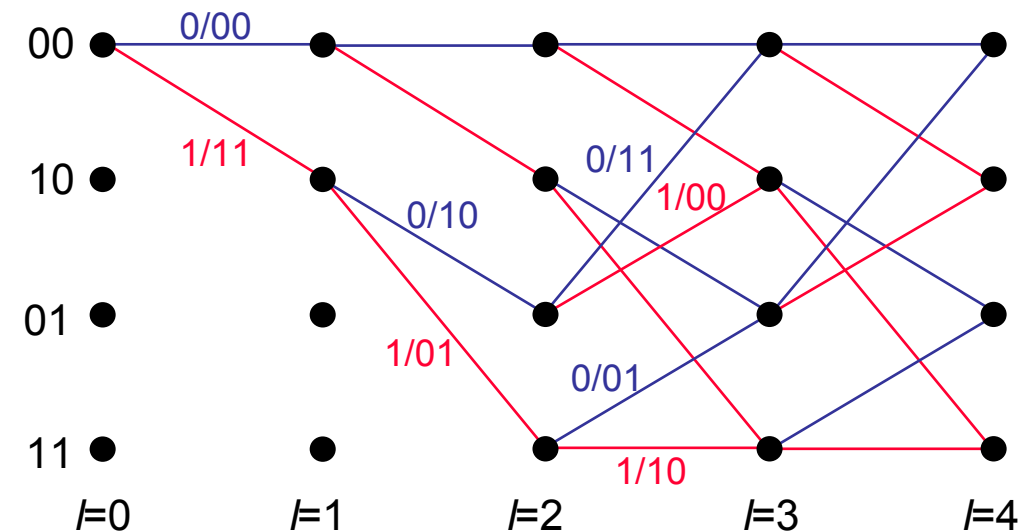
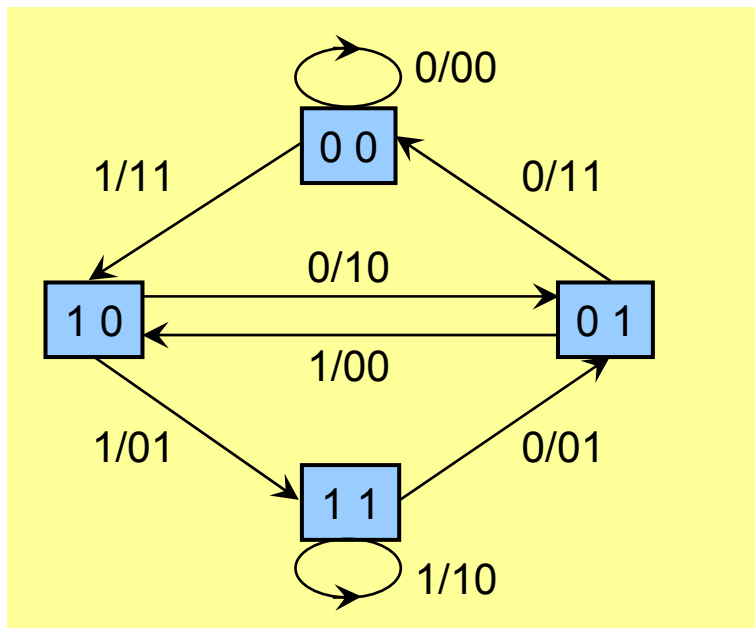
# Finite State Diagram

- Convolutional encoder can be interpreted as **Mealy** state machine  
 $\Rightarrow$  They can be described by a state transition (dependent on the current state and the input) and the corresponding output signal
- Example: (2,1,3)-code with generators  $g_1 = 7_8$  and  $g_2 = 5_8$



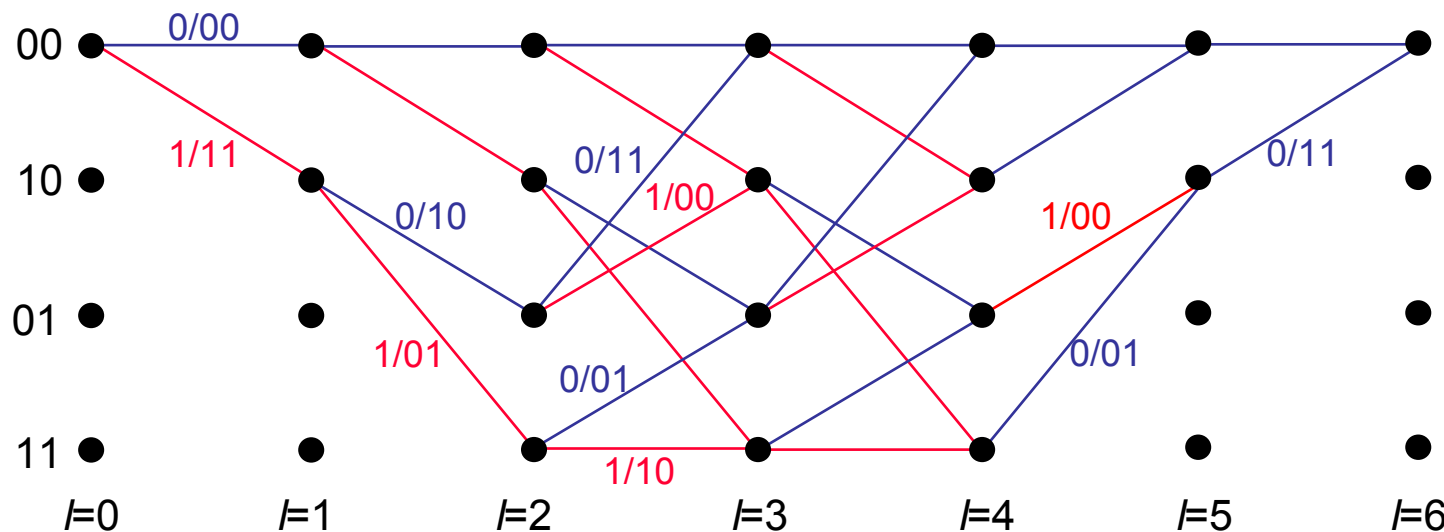
# Trellis Diagram

- Finite state diagram does not contain any information in time  
 $\Rightarrow$  State diagram expanded by temporal component results in Trellis diagram
- Example: (2,1,3)-code with generators  $g_1 = 7_8$  and  $g_2 = 5_8$



- Appending tail bits to information sequence  $\mathbf{u}$ 
  - ⇒ encoder stops in a predefined state (usually state 0)
  - ⇒ reliable decoding of the last information bits
- Number of tail bits equals memory of encoder (adding  $m$  zeroes)
- Adding tail bits reduces the code rate:

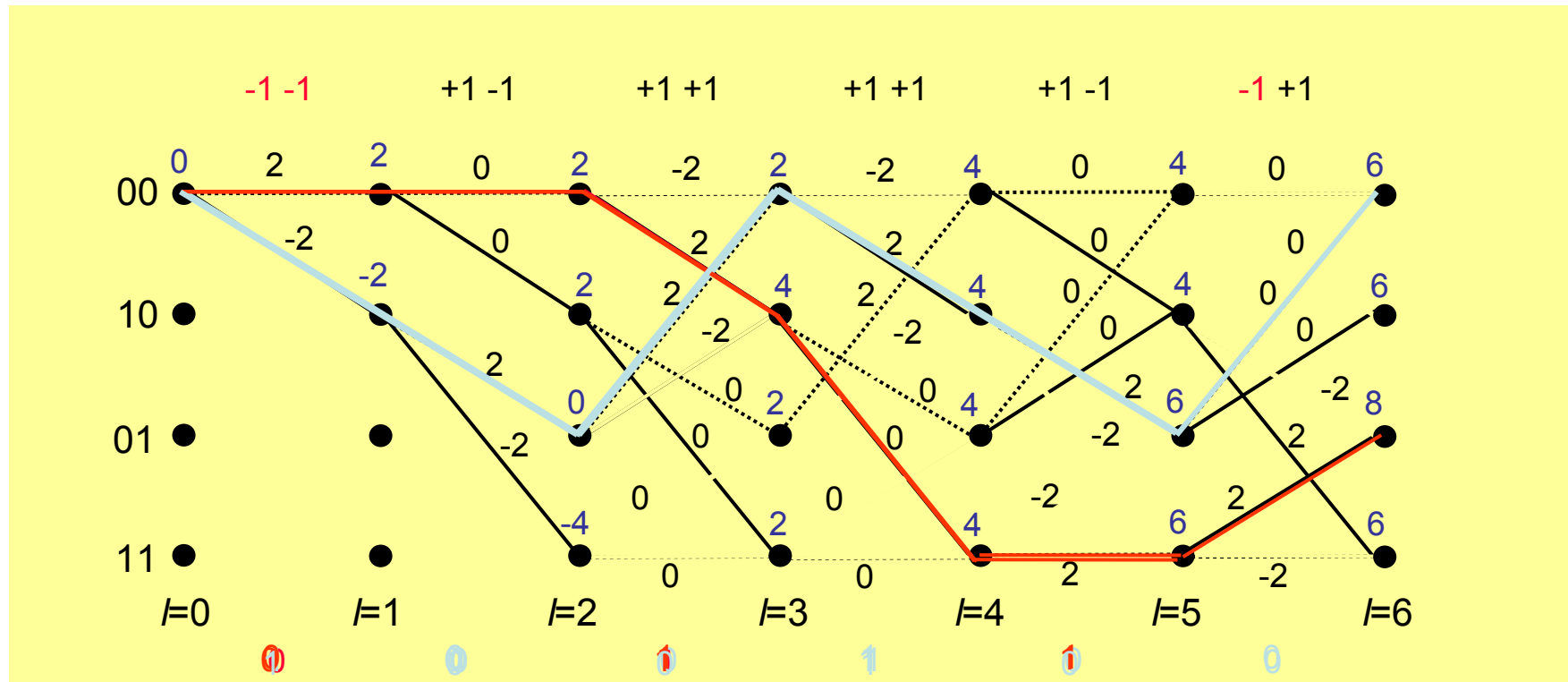
$$R_c^{\text{tail}} = \frac{N}{n \cdot (N + m)} = R_c \cdot \frac{N}{N + m} \approx \frac{1}{n} \quad \text{for } N \gg m$$



- Computational complexity of direct approach grows exponentially with sequence length  
→ **prohibitively high!**
  - For  $N = 100$  information bits,  $2^{100} \approx 1.27 \cdot 10^{30}$  comparisons
- Breakthrough for convolutional codes came with invention of **Viterbi algorithm** in 1967
  - Complexity depends only linearly on sequence length
  - But complexity still grows exponentially with memory of encoder (compare with number of states in trellis diagram)
- Trick: exploitation of Markov property of encoder

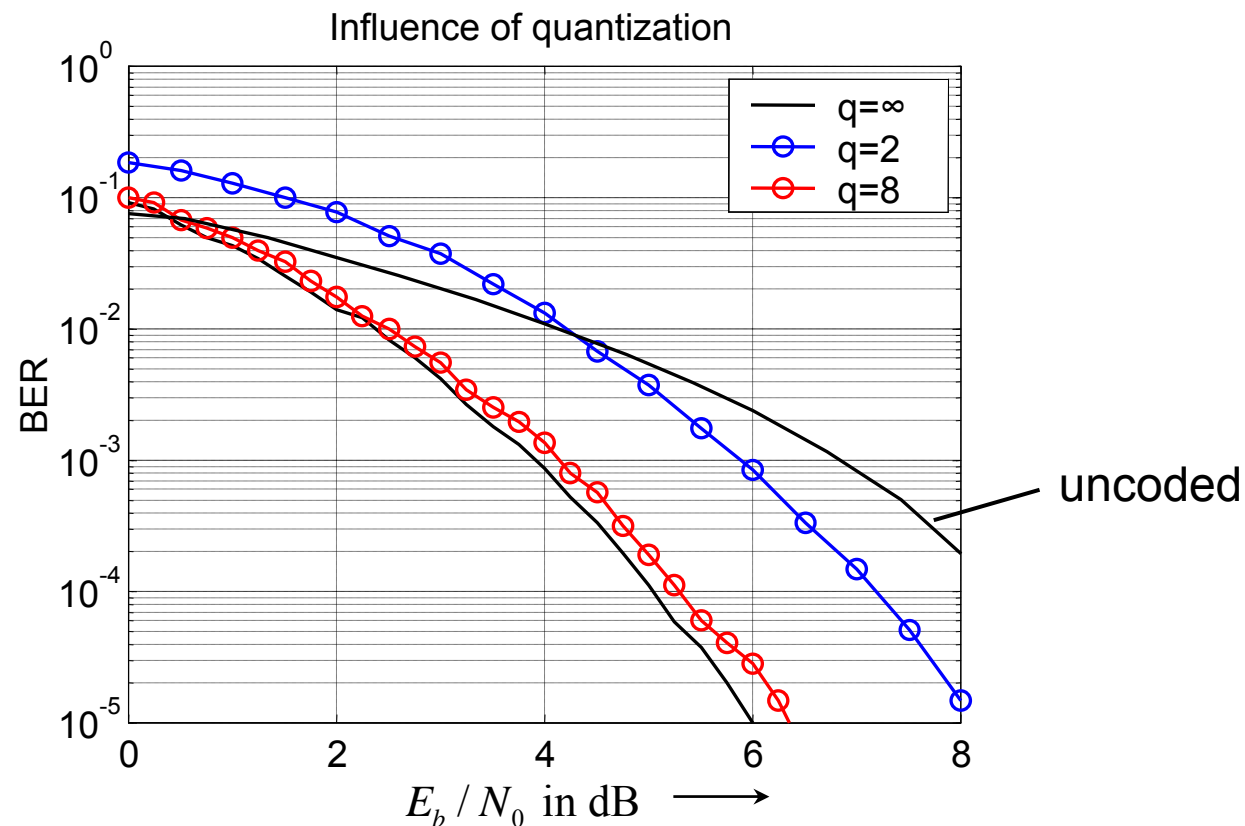
# Viterbi Decoding of Convolutional Codes

- Example : (2,1,3)-code with generators  $g_1 = 7_8$  and  $g_2 = 5_8$
- Information sequence:  $\mathbf{u} = [1 \ 0 \ 0 \ 1 \ | \ 0 \ 0]$



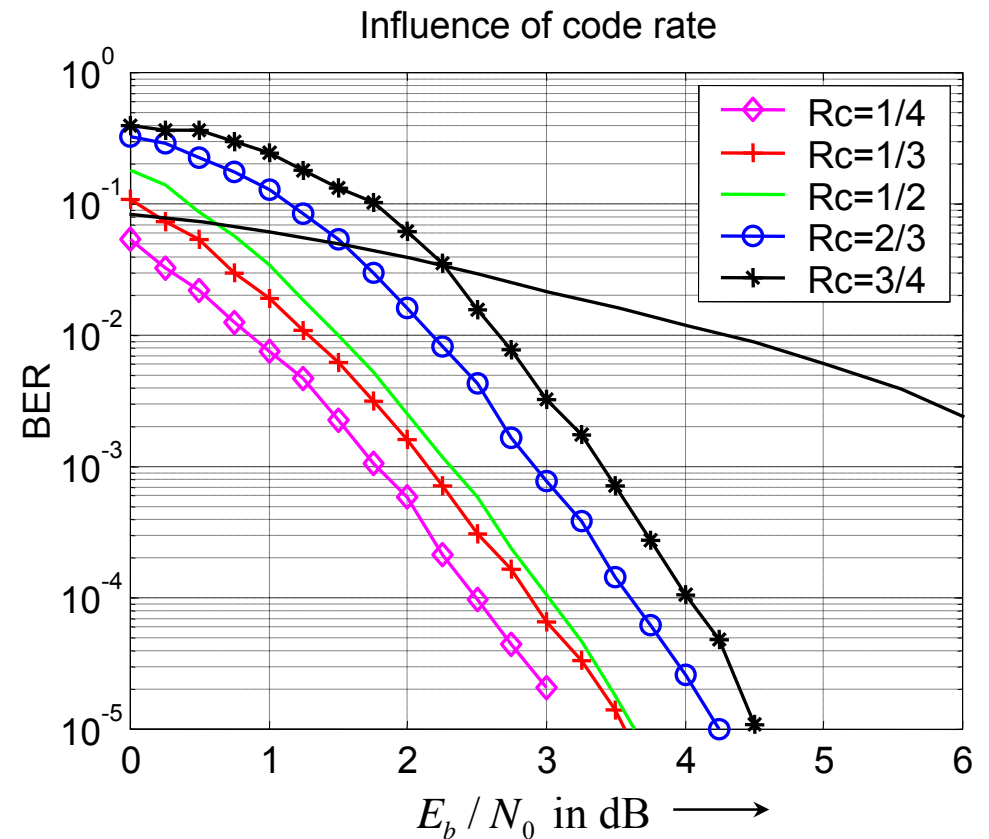
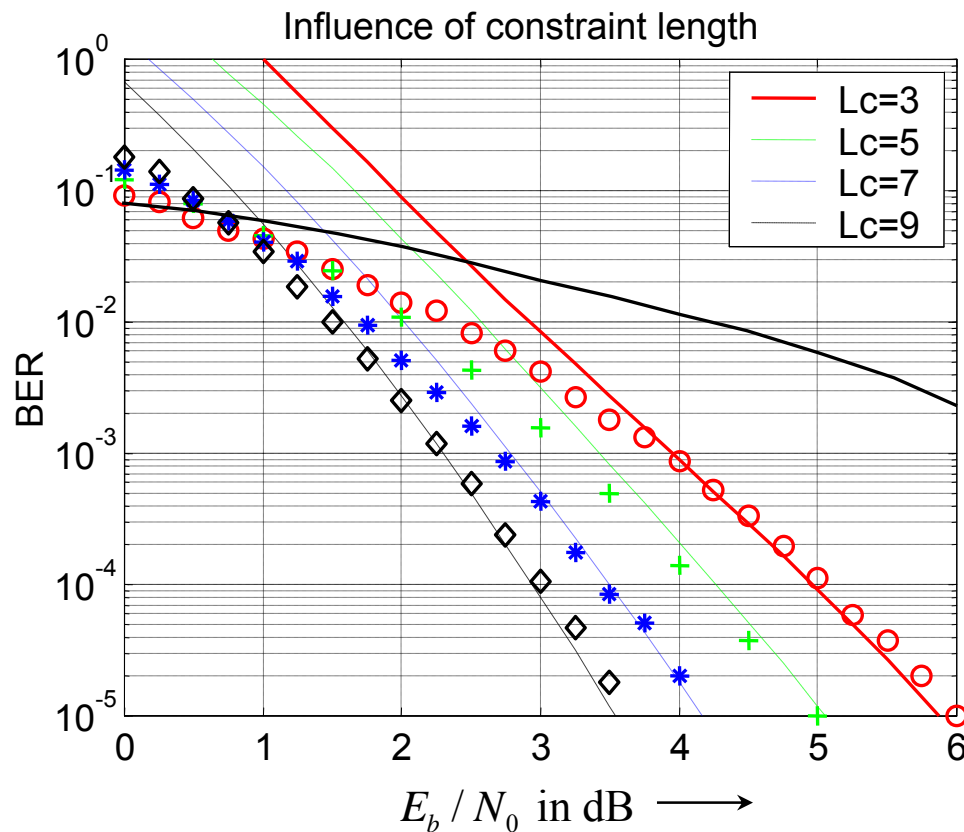
- Start trellis in state 0
- Calculate correlation between  $\mathbf{y}(l)$  and any possible code word  $\mathbf{a}(l)$
- Add the incremental path metric (correlation) to the old cumulative state metric  $M_j(l-1)$ ,  $j = 0, \dots, 2^m - 1$
- Select for each state the path with smallest cumulative metric (Euclidian distance) and discard the other paths  
⇒ effort increases only linearly with observation length (not exponentially)
- Return to step 2) until all  $N$  received words have been processed
- End of the Trellis
  - Terminated code (trellis ends in state 0): ⇒ select path with best metric  $M_0(N)$ ,
  - Truncated code: ⇒ select path with the overall best metric  $M_j(N)$
- Trace back the path selected in 6) (survivor) and output the corresponding information bits

- Quantizing received sequence before decoding wastes information
  - Hard-Decision ( $q = 2$ ): Strong performance degradation
  - 3-bit quantization ( $q = 8$ ): only a small performance degradation



# Performance of Convolutional Codes

- The larger the constraint length, the higher the performance (and the decoding complexity)
- Performance increases with decreasing code rate



**Thanks for your attention!**