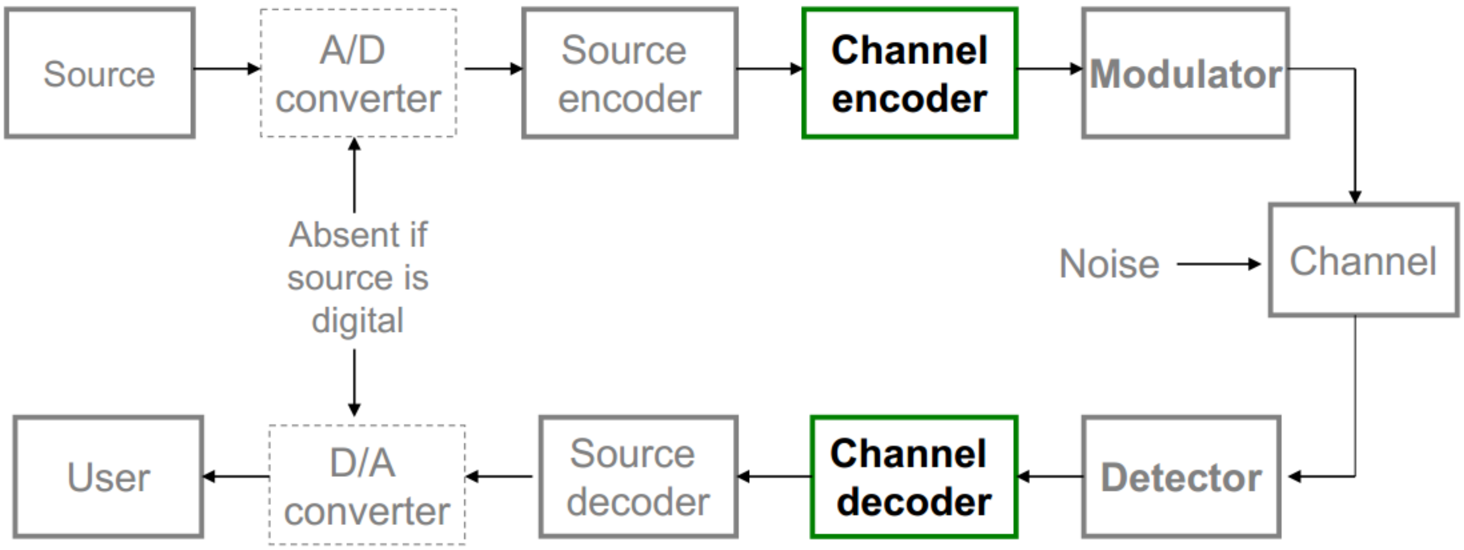
Outline

- Introduction
- Signal, random variable, random process and spectra
- Analog modulation
- Analog to digital conversion
- Digital transmission through baseband channels
- Signal space representation
- Optimal receivers
- Digital modulation techniques
- **Channel coding**
- Synchronization
- Information theory

Channel coding



- Linear block code
- Convolutional code

Chapter 13.1-13.3

Channel coding

- Information theory and channel coding
  - Shannon's noisy channel coding theorem tells us that adding controlled redundancy allows transmission at arbitrarily low bit error rate (BER) as long as $R \leq C$
  - Error control coding (ECC) uses this controlled redundancy to detect and correct errors
  - ECC depends on the system requirements and the nature of the channel
  - The key in ECC is to find a way to add redundancy to the channel so that the receiver can fully utilize that redundancy to detect and correct the errors, and to reduce the required transmit power (coding gain)

Channel coding

- Information theory and channel coding
  - Consider for example the case that we want to transmit data over a telephone link using a modem under the conditions that link bandwidth = 3 kHz and the modem can operate up to the speed of 3600 bits/sec at an error probability Pe = 8×10^{-4}.
  - Target: transmit the data the rate of 1200 bits/sec at maximum output SNR = 13 dB with a probability of error 1×10^{-4}

Channel coding

- Information theory and channel coding
  - Shannon theorem tells us that channel capacity is

    $$C = B\log_2\left(1+\frac{S}{N}\right) = 13,000 \text{ bits/sec}$$

    since B=3000, S/N=13 dB=20
  - Thus, by Shannon's theorem, we can transmit the data with an arbitrarily small error probability
  - Note that without coding Pe = 8×10^{-4}, the target Pe is not met.

Channel coding

- Information theory and channel coding
  - Consider a simple code design with repetition code.
  - Every bit is transmitted 3 times, e.g., when b_k="0" or "1", transmitted codewords are "000" or "111"
  - Based on the received codewords, the decoder attempts to extract the transmitted bits using majority-logic decoding scheme
  - Obviously, the transmitted bits will be recovered correctly as long as no more than one of the bits in the codewords is affected by noise

Tx bits b_k	0	0	0	0	1	1	1	1
Codewords	000	000	000	000	111	111	111	111
Rx bits	000	001	010	100	011	101	110	111
$\hat{b}_k$	0	0	0	0	1	1	1	1

Channel coding

- Information theory and channel coding
  - With this simple error control coding, the probability of error is

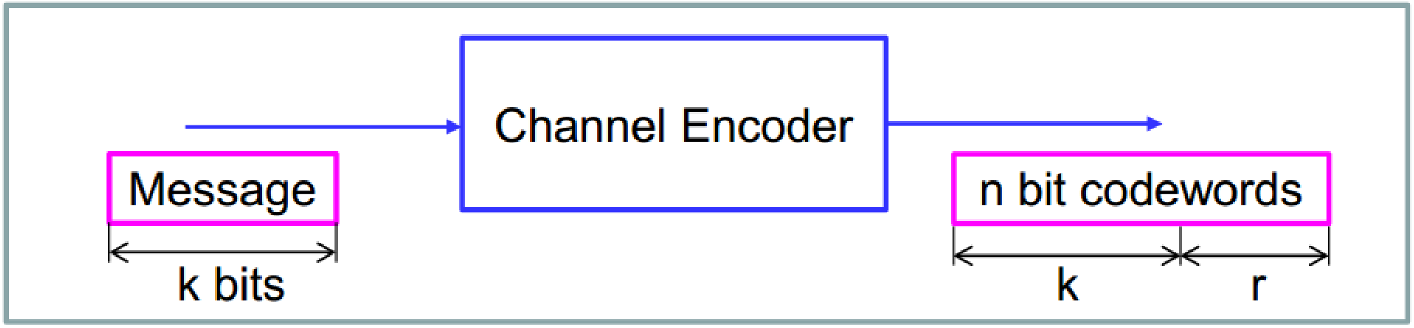
$$
\begin{aligned}
P_e &= P(b_k \neq \hat{b}_k) \\
&= P\ (\text{2 or more bits in codeword are in error}) \\
&= \binom{3}{2} q_c^2 (1 - q_c) + \binom{3}{3} q_c^3 \\
&= 3q_c^2 - 2q_c^3 \\
&= 0.0192 \times 10^{-4} \\
&\leq \text{Required } P_e \text{ of } 10^{-4}
\end{aligned}
$$

Channel coding

- From the above example, we can see the importance of coding techniques.
- Coding techniques are classified as either block codes or convolutional codes, depending on the presence or absence of memory
- A block code has no memory
  - Information sequence is broken into blocks of length k
  - Each block of k inf. bits is encoded into a block of n coded bits
  - No memory from one block to another block
- A convolutional code has memory
  - A shift register of length k_0L is used
  - Inf. bits enter the shift register k_0 bits at a time and n_0 coded bits are generated
  - These n_0 bits depend not only on the recent k_0 bits, but also on the $k_0(L-1)$ previous bits

Linear block codes

- Block codes
  - An (n,k) block code is a collection of $M=2^k$ codewords of length n
  - Each codeword has a block of k inf. bits followed by a group of r=n-k check bits that are derived from the k inf. bits in the block preceding the check bits



  - The code is said to be linear if any linear combination of 2 codewords is also a codeword, i.e., if c_i and c_j are codewords, then $c_i + c_j$ is also a codeword (addition is module-2)

Linear Block codes

- **Code rate** (rate efficiency) = k/n
- Matrix description:
  - Codeword $\mathbf{c} = (c_1, c_2, ..., c_n)$
  - Message bits $\mathbf{m} = (m_1, m_2, ..., m_k)$
- Each block code can be generated using a **Generator matrix** **G** (dim: kxn)
- Given **G**, then

$$\boxed{\mathbf{c} = \mathbf{m}\mathbf{G}}$$

Codeword message

Linear Block codes

- Generator matrix **G**

$$\mathbf{G} = [\mathbf{I}_k | \mathbf{P}]_{k\times n}$$

$$= \left[\begin{array}{cccc|cccc} 1 & 0 & \cdots & 0 & p_{11} & p_{12} & \cdots & p_{1,n-k} \\ 0 & 1 & & 0 & p_{21} & p_{22} & & p_{2,n-k} \\ \vdots & & \ddots & \vdots & \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & p_{k,1} & p_{k,2} & \cdots & p_{k,n-k} \end{array}\right]$$

  - $\mathbf{I}_k$ is an identity matrix of order k
  - **P** is a matrix of order kx(n-k), which is selected so that the code will have certain desired properties

Linear Block codes

- Generator matrix **G**
  - The form of **G** implies that the 1^{st} k components of any codeword are precisely the information symbols
  - This form of linear encoding is called systematic encoding
  - Systematic-form codes allow easy implementation and quick look-up features for decoding
  - For linear codes, any code is equivalent to a code in systematic form (given the same performance). Thus, we can restrict our study to only systematic codes

Linear Block codes

- Example
  - Hamming code is a family of (n,k) linear block codes that have the following parameters
    1. Codeword length $n = 2^m - 1, \quad m \geq 3$
    2. # of message bits $k = 2^m - m - 1$
    3. # of parity check bits $n - k = m$
    4. Capable of providing single-error correction capability with $d_{\min} = 3$
  - (7,4) Hamming code with generator matrix

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

Find all codewords

Linear Block codes

- Example
  - (7,4) Hamming code

Message				codeword						
0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1	1	0	1
0	0	1	0	0	0	1	0	1	1	1
0	0	1	1	0	0	1	1	0	1	0
0	1	0	0	0	1	0	0	0	1	1
0	1	0	1	0	1	0	1	1	1	0
0	1	1	0	0	1	1	0	1	0	0
0	1	1	1	0	1	1	1	0	0	1
1	0	0	0	1	0	0	0	1	1	0
1	0	0	1	1	0	0	1	0	1	1
1	0	1	0	1	0	1	0	0	0	1
1	0	1	1	1	0	1	1	1	0	0
1	1	0	0	1	1	0	0	1	0	1
1	1	0	1	1	1	0	1	0	0	0
1	1	1	0	1	1	1	0	0	1	0
1	1	1	1	1	1	1	1	1	1	1

Linear Block codes

- Parity check matrix
  - For each **G**, it is possible to find a corresponding parity check matrix **H** $\mathbf{H} = \left[\mathbf{P}^T \; | \mathbf{I}_{n-k}\right]_{(n-k)\times n}$
  - **H** can be used to verify if a codeword C is generated by **G**
  - Let C be a codeword generated by $\mathbf{G} = [\mathbf{I}_k | \mathbf{P}]_{k\times n}$

$$\Rightarrow \boxed{\mathbf{c}\mathbf{H}^T = \mathbf{m}\mathbf{G}\mathbf{H}^T = 0}$$

  - Think about the parity check matrix of (7,4) Hamming code

Linear Block codes

- Error syndrome
  - Received codeword **r**=**c**+**e**, where **e**=Error vector or Error pattern and it is 1 in every position where data word is in error
  - Example

$$\mathbf{c} = [1\ \ 0\ \ 1\ \ 0]$$

$$\mathbf{r} = [1\ \ 1\ \ 0\ \ 0]$$

$$\mathbf{e} = [0\ \ 1\ \ 1\ \ 0]$$

  - Error syndrome: $\mathbf{s} \triangleq \mathbf{r}\mathbf{H}^T$

Linear Block codes

- Error syndrome
  - Note that

$$\begin{aligned} \mathbf{s} &= \mathbf{r}\mathbf{H}^T = (\mathbf{c}+\mathbf{e})\mathbf{H}^T \\ &= \mathbf{c}\mathbf{H}^T + \mathbf{e}\mathbf{H}^T \\ &= \mathbf{e}\mathbf{H}^T \end{aligned}$$

  - If **s**=**0**, then **r** = **c** and m is the 1st k bits of **r**
  - If **s**≠**0**, and **s** is the jth row of $\mathbf{H}^T$, then 1 error in jth position of **r**

Linear Block codes

- Error syndrome
  - Consider the (7,4) Hamming code for example

$$\mathbf{H}^T = \left[\mathbf{P}^T | \mathbf{I}_{n-k}\right]^T = \begin{bmatrix} \mathbf{P} \\ \mathbf{I}_{n-k} \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ \hdashline 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

  - So if $\mathbf{r} = [1\ 1\ 1\ 1\ 1\ 1\ 1]$

$$\Longrightarrow \quad \mathbf{r}\mathbf{H}^T = [0\ 0\ 0]$$

  - But if $\mathbf{r} = [1\ 1\ 1\ 1\ 1\ 1\ 0]$

$$\Longrightarrow \quad \mathbf{r}\mathbf{H}^T = [0\ 0\ 1]$$

= Error syndrome $\mathbf{s}$

- Note that s is the last row of $\mathbf{H}^T$
- Also note error took place in the last bit
- Syndrome indicates error position

Linear Block codes

- Cyclic code
  - A code $C = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_{2^k}\}$ is cyclic if

$$(c_1, c_2, \ldots, c_n) \in C \Rightarrow (c_n, c_1, \ldots, c_{n-1}) \in C$$

  - (7,4) Hamming code is cyclic

message	codeword
0001	0001101
1000	1000110
0100	0100011

Linear Block codes

- Important parameters

  Hamming Distance between codewords c_i and c_j:

  $d(c_i, c_j) =$ # of components at which the 2 codewords differ

  Hamming weight of a codeword c_i is

  $w(c_i) =$ # of non-zero components in the codeword

  Minimum Hamming Distance of a code:

  $d_{min} = \min d(c_i, c_j)$ for all $i \neq j$

  Minimum Weight of a code:

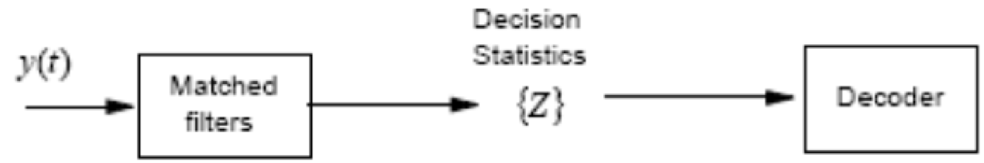
  $w_{min} = \min w(c_i)$ for all $c_i \neq 0$

  Theorem: In any linear code, $d_{min} = w_{min}$

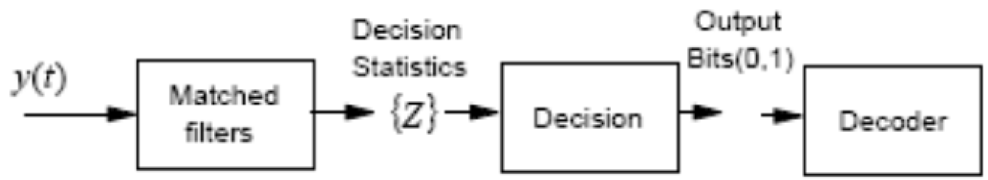
  Exercise: Find d_{min} for (7,4) Hamming code

Linear Block codes

- Soft-decision and hard-decision decoding
  - Soft-decision decoder operates directly on the decision statistics



  - Hard-decision decoder makes "hard" decision (0 or 1) on individual bits



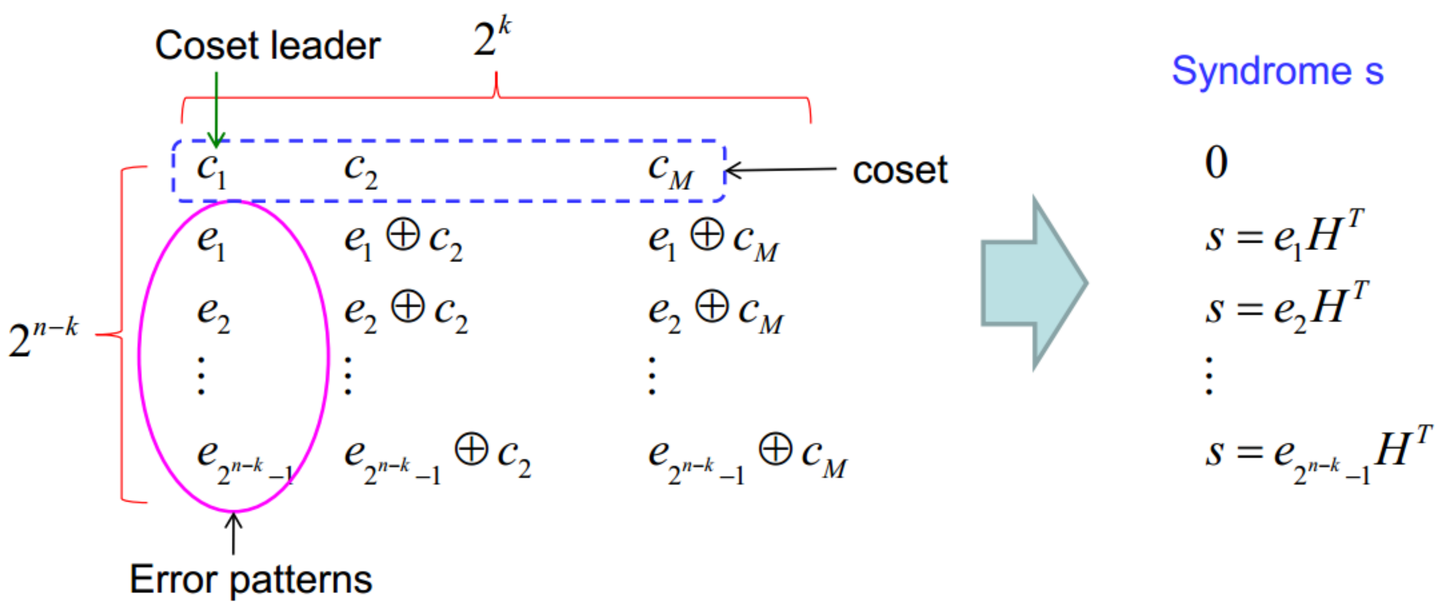
  - Here we only focus on hard decision decoder

Linear Block codes

- Hard-decision decoding
  - Minimum Hamming distance decoding
    1. Given the received codeword **r**, choose **c** which is closest to **r** in terms of Hamming distance
    2. To do so, one can do an exhaustive search (but complexity problem if k is large)
  - Syndrome decoding
    1. Syndrome testing: $\mathbf{r}=\mathbf{c}+\mathbf{e}$ with $\mathbf{s}=\mathbf{r}\mathbf{H}^T$
    2. This implies that the corrupted codeword **r** and the error pattern have the same syndrome
    3. A simplified decoding procedure based on the above observation can be used

Linear Block codes

- Hard-decision decoding
  - Let the codewords be denoted as $\{c_1, c_2, \ldots, c_M\}$ with c1 being the all-zero codeword
  - A standard array is constructed as

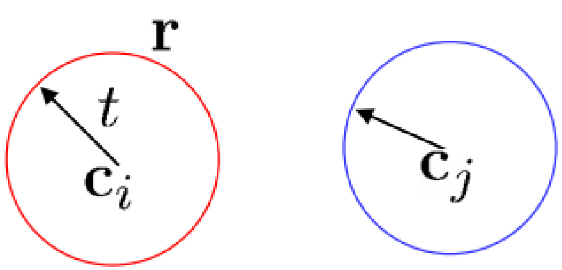


Linear Block codes

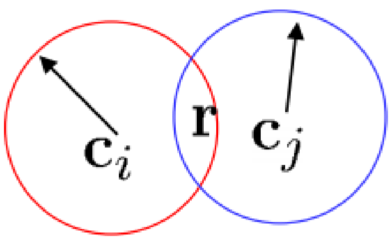
- Hard-decision decoding
  - Hard-decoding procedure
    1. Find the syndrome by **r** using $\mathbf{s}=\mathbf{r}\mathbf{H}^T$
    2. Find the coset corresponding to **s** by using the standard array
    3. Find the cost leader and decode as $\mathbf{c}=\mathbf{r}+\mathbf{e}_j$
  - Try on (7,4) Hamming code

Linear Block codes

- Hard-decision decoding
  - A linear block code with a minimum distance d_{min} can
    1. Detect up to (d_{min}-1) errors in each codeword
    2. Correct up to $t = \lfloor \frac{d_{min}-1}{2} \rfloor$ errors in each codeword
    3. t is known as the error correction capability of the codeword



$d(\mathbf{c}_i, \mathbf{c}_j) \geq 2t + 1$



$d(\mathbf{c}_i, \mathbf{c}_j) < 2t$

Linear Block codes

- Hard-decision decoding
  - Consider a linear block code (n,k) with an error correcting capability t. The decoder can correct all combination of errors up to and including t errors
  - Assume that the error probability of each individual coded bit is p and that bit errors occur independently since the channel is memoryless
  - If we send n-bit block, the probability of receiving a specific pattern of m errors and (n-m) correct bits is

$$p^m(1-p)^{n-m}$$

  - Total number of distinct patterns of n bits with m errors and (n-m) correct bits is

$$\binom{n}{m} = \frac{n!}{m!(n-m)!}$$

Linear Block codes

- Hard-decision decoding
  - Total probability of receiving a pattern with m errors is

$$P(m,n) = \binom{n}{m} \cdot p^m (1-p)^{n-m}$$

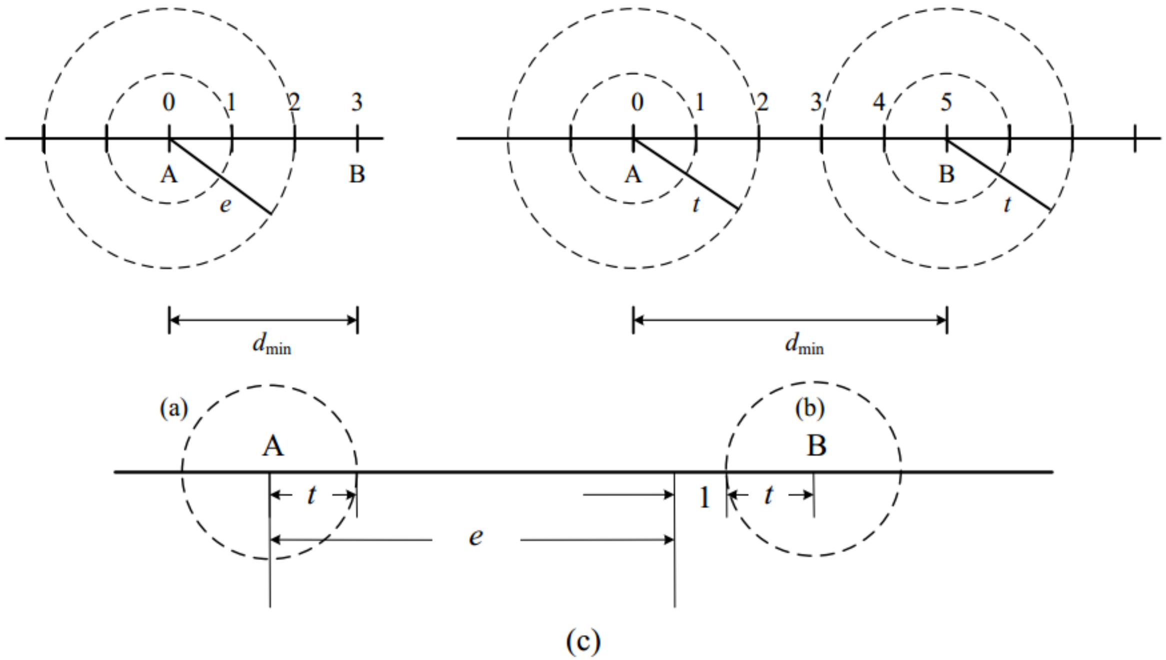
  - Thus, the codeword error probability is upperbounded by

$$P_M \leq \sum_{m=t+1}^{n} \binom{n}{m} p^m (1-p)^{n-m}$$

(with equality for perfect codes)

Linear Block codes

- Hard-decision decoding
  - ➢ Key parameters.



  - ➢ To detect e bit errors, we have $d_{min} \geq e+1$
  - ➢ To correct t bit errors, we have $d_{min} \geq 2t+1$

Linear Block codes

- Major classes of block codes
  - Repetition code
  - Hamming code
  - Golay code
  - BCH code
  - Reed-Solomon codes
  - Walsh codes
  - LDPC codes: invented by Robert Gallager in his PhD thesis in 1960, now proved to be capacity approaching and adopted in **5G** standards

Convolutional codes

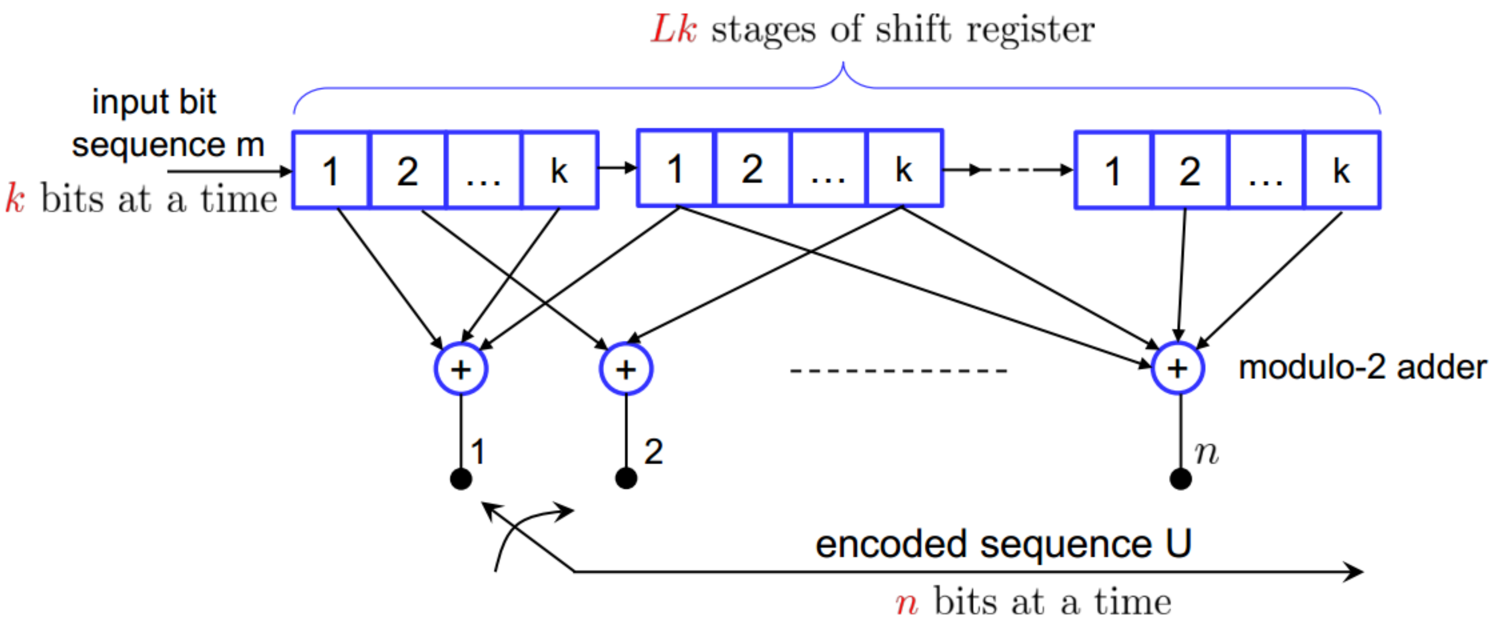
- A convolutional code has memory
  - It is described by 3 integers: n, k, and L
  - Maps k inf. bits into n bits using previous (L-1)k bits
  - The n bits emitted by the encoder are not only a function of the current input k bits, but also a function of the previous (L-1)k bits
  - Code rate = k/n (information bits/coded bits)
  - L is the constraint length and is a measure of the code memory
  - n does not define a block or codeword length

Convolutional codes

- Convolutional encoding
  - A rate k/n convolutional encoder with constraint length L consists of kL-stage shift register and n mod-2 adders
  - At each unit of time
    1. k bits are shifted into the 1^{st} k stages of the register
    2. All bits in the register are shifted k stages to the right
    3. The output of the n adders are sequentially sampled to give the coded bits
    4. There are n coded bits for each input group of k bits or message bits. Hence R=k/n information bits/coded bits is the code rate (k<n)

Convolutional codes

- Convolutional encoding
  - Encoder structure.



  - Typically, k=1 for binary codes. Hence, consider rate 1/n codes for example.

Convolutional codes

- Convolutional encoding
  - Encoding function: characterizes the relationship between the information sequence m and the output coded sequence U.
  - Four popular methods for representation
    1. Connection pictorial and connection polynomials (usually for encoder)
    2. State diagram
    3. Tree diagram
    4. Trellis diagram

Usually for decoder

Convolutional codes

- Convolutional encoding
  - Connection representation.
  - Specify n connection vectors, $\mathbf{g}_i, i = 1, \ldots, n$ for each of the n mod-2 adders
  - Each vector has kL dimension and describes the connection of the shift register to the mod-2 adders
  - A 1 in the i^{th} position of the connection vector implies shift register is connected
  - A 0 implies no connection exists

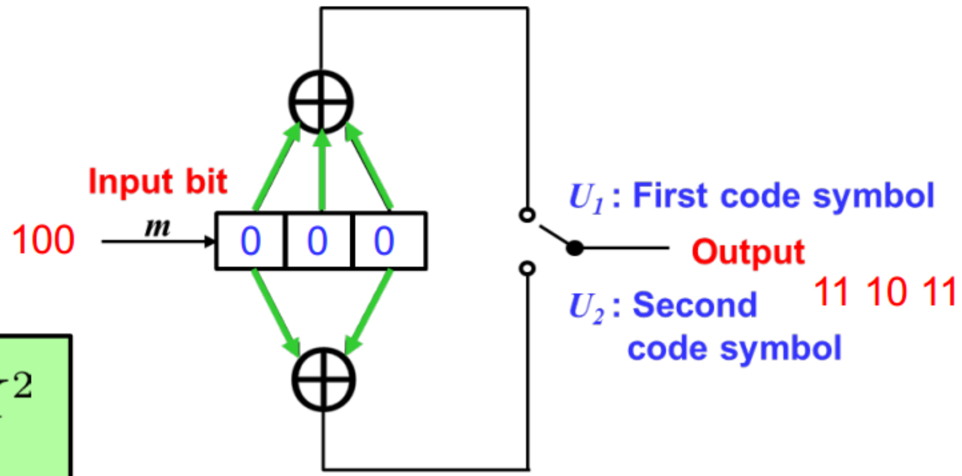
Convolutional codes

- Convolutional encoding
  - ➢ Connection representation (L=3, Rate 1/2).

$$g_1 = 1\,1\,1$$
$$g_2 = 1\,0\,1$$

Or

$$g_1(X) = 1 + X + X^2$$
$$g_2(X) = 1 + X^2$$



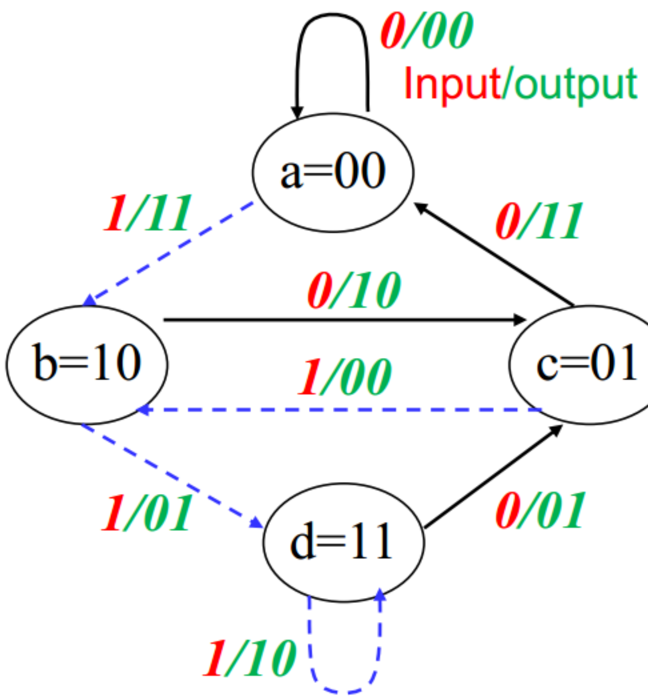
If Initial Register content is 0 0 0 and Input Sequence is 1 0 0. Then Output Sequence is 11 10 11

Convolutional codes

- Convolutional encoding
  - State diagram representation.
  - The contents of the rightmost L-1 stages (or the previous L-1 bits) are considered the current state, 2^{L-1} states
  - Knowledge of the current state and the next input is necessary and sufficient to determine the next output and next state
  - For each state, there are only 2 transitions (to the next state) corresponding to the 2 possible input bits
  - The transitions are represented by paths on which we write the output word associated with the state transition: A solid line path corresponds to an input bit 0, while dashed line for 1

Convolutional codes

- Convolutional encoding
  - ➢ State diagram representation (L=3, Rate 1/2).



Current State	Input	Next State	Output
00	0	00	***00***
	1	10	***11***
10	0	01	***10***
	1	11	***01***
01	0	00	***11***
	1	10	***00***
11	0	01	***01***
	1	11	***10***

Convolutional codes

- Convolutional encoding
  - State diagram representation.
  - Assume that m=11011 is the input followed by L-1=2 zeros to flush the register. Also assume that the initial register contents are all zero. Find the output sequence U

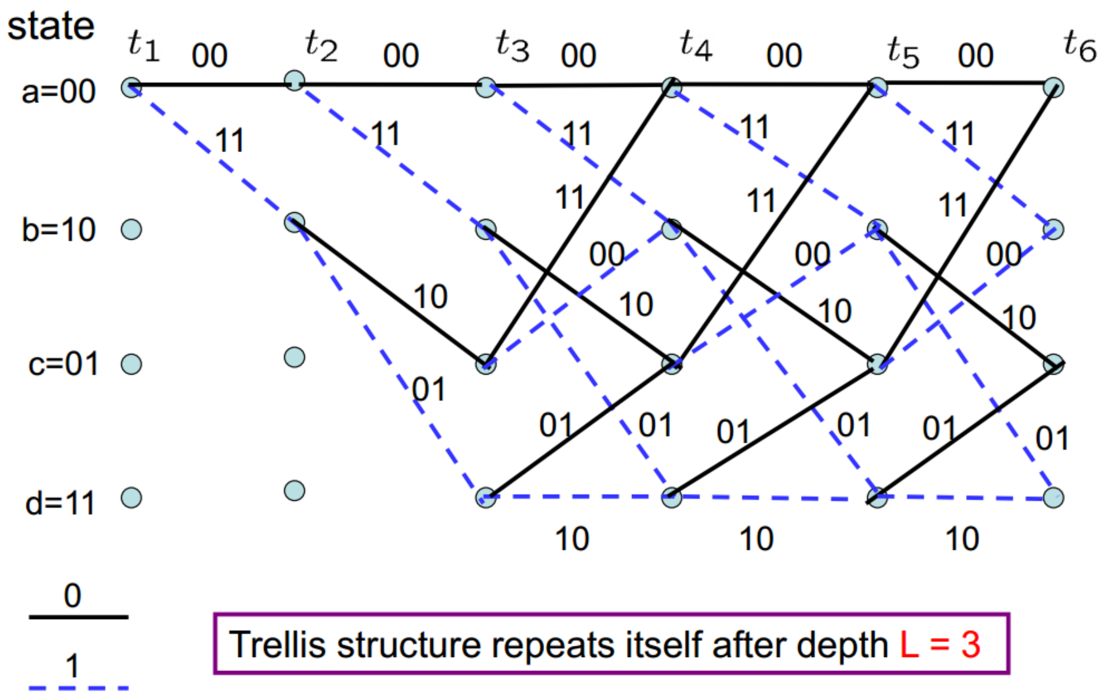
Input bit m_i	Register contents	State at time t_i	State at time t_{i+1}	Branch word at time t_i	
				u_1	u_2
--	000	00	00	--	--
1	100	00	10	1	1
1	110	10	11	1	1
0	011	11	01	0	0
1	101	01	10	1	1
1	110	10	11	1	1
0	011	11	01	0	0
0	001	01	00	0	0

State t_{i+1}

State t_i

Output sequence: U = **11 01 01 00 01 01 11**

- Convolutional encoding
  - Trellis diagram representation.
  - Trellis diagram is similar to the state diagram, except that it adds the dimension of time.
  - The code is represented by a trellis where each trellis branch describes an output word

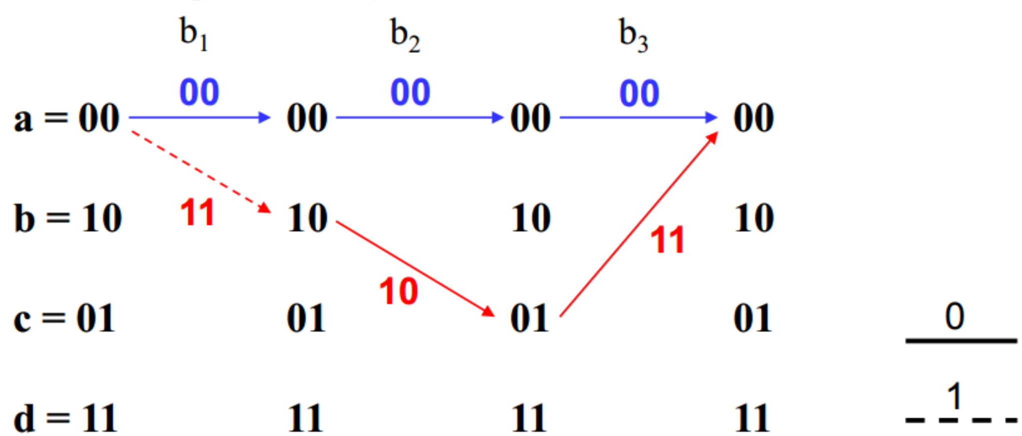


Convolutional codes

- Convolutional encoding
  - Trellis diagram representation.
  - Every input sequence $(m_1, m_2, \ldots)$ corresponds to
    1. a path in the trellis
    2. a state transition sequence $(s_0, s_1, \ldots)$, (assume $s_0 = 0$ is fixed)
    3. an output sequence $((u_1, u_2), (u_3, u_4), \ldots)$
  - For instance, let s_0=00, then

$b_1 b_2 b_3 = 000$ gives output *000000* and states *aaaa*

$b_1 b_2 b_3 = 100$ gives output *111011* and states *abca*



Convolutional codes

- Update
  - We have discussed conv. code with constraint length L and rate 1/n, and the different representations
  - We will talk about decoding of convolutional code with maximum likelihood decoding, Viterbi algorithm, and transfer function

Convolutional codes

- Maximum likelihood decoding
  - Transmit a coded sequence $U^{(m)}$ (corresponds to message sequence m) using a digital modulation scheme (e.g., BPSK or QPSK)
  - Received sequence z
  - Maximum likelihood decoder will
    1. Find the sequence $U^{(j)}$ such that

$$P(\mathbf{Z}|\mathbf{U}^{j}) = \max_{\forall \mathbf{U}^{(m)}} P(\mathbf{Z}|\mathbf{U}^{(m)})$$

    2. Minimize the probability of error if m is equally likely

Convolutional codes

- Maximum likelihood decoding
  - Assume a memoryless channel, i.e., noise components are independent. Then, for a rate 1/n code

$$P(\mathbf{Z}|\mathbf{U}^{(\mathbf{m})}) = \prod_{i=1}^{\infty} P(Z_i|U_i^{(m)}) = \prod_{i=1}^{\infty}\prod_{j=1}^{n} P(z_{ji}|u_{ji}^{(m)})$$

i-th branch of **Z**

  - Then, the problem is to find a path through the trellis such that

$$\max_{\mathbf{U}^{(m)}} \prod_{i=1}^{\infty}\prod_{j=1}^{n} P(z_{ji}|u_{ji}^{(m)})$$

by taking log

Log-likelihood path metric

$$\max_{\mathbf{U}^{(m)}} \sum_{i=1}^{\infty}\sum_{j=1}^{n} \log P(z_{ji}|u_{ji}^{(m)})$$

i-th branch metric

$$= \max_{\mathbf{U}^{(m)}} \sum_{i=1}^{\infty}\sum_{j=1}^{n} LL\left(z_{ji}|u_{ji}^{(m)}\right)$$

Log-likelihood of $z_{ji}|u_{ji}^{(m)}$

Convolutional codes

- Maximum likelihood decoding
  - Log-likelihood.
  - For AWGN channel with soft decision

$z_{ji} = u_{ji} + n_{ji}$ and P($z_{ji}|u_{ji}$) is Gaussian with mean u_{ji} and variance σ^2

Hence

$$\ln p(z_{ji}|u_{ji}) = -\frac{1}{2}\ln(2\pi\sigma^2) - \frac{(z_{ji}-u_{ji})^2}{2\sigma^2}$$

Note that the objective is to compare which Σ_i ln(p(z|u)) for different **u** is larger, hence, constant and scaling does not affect the results

Then, we let the log-likelihood be $LL(z_{ji}|u_{ji}) = -(z_{ji}-u_{ji})^2$

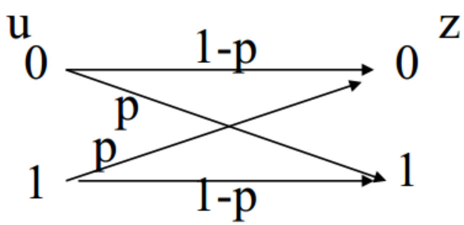
and

$$\boxed{\log P(Z|U^{(m)}) = -\sum_{i=1}^{\infty}\sum_{j=1}^{n}\left(z_{ji}-u_{ji}^{(m)}\right)^2}$$

Thus, soft decision ML decoder is to choose the path whose corresponding sequence is at the minimum Euclidean distance from the received sequence

Convolutional codes

- Maximum likelihood decoding
  - Log-likelihood.
  - For binary symmetric channel (hard decision)



$$p(z\,|\,u)=\begin{cases} p & \text{if } z\neq u \\ 1-p & \text{if } z=u \end{cases}$$

$$LL(z_{ji}\,|\,u_{ji}) = \ln p(z_{ji}\,|\,u_{ji}) = \begin{cases} \ln p & \text{if } z_{ji}\neq u_{ji_i} \\ \ln(1-p) & \text{if } z_{ji}=u_{ji} \end{cases}$$

$$= \begin{cases} \ln p/(1-p) & \text{if } z_{ji}\neq u_{ji} \\ 0 & \text{if } z_{ji}=u_{ji} \end{cases}$$

$$= \begin{cases} -1 & \text{if } z_{ji}\neq u_{ji} \\ 0 & \text{if } z_{ji}=u_{ji} \end{cases} \quad \text{(since p<0.5)}$$

- Thus

$$\boxed{\log P(Z|U^{(m)}) = -d_m}$$

Hamming distance between Z and $U^{(m)}$, i.e. they differ in d_m positions

Hard-Decision ML Decoder = Minimum Hamming Distance Decoder

Convolutional codes

- Maximum likelihood decoding
  - Decoding procedure:
    1. Compute, for each branch i, the branch metric using output bits $\{u_{1,i}, u_{2,i}, \ldots, u_{n,i}\}$ associated with that branch and the received symbols $\{z_{1,i}, z_{2,i}, \ldots, z_{n,i}\}$
    2. Compute, for each valid path through the trellis (a valid codeword sequence U(m)), the sum of the branch metrics along that path
    3. The path with the maximum path metric is the decoded path
  - To compare all possible valid paths, we need to do exhaustive search or brute-force, not practical as the # of paths grows exponentially as the path length increases
  - The optimal algorithm for solving this problem is the Viterbi decoding algorithm or Viterbi decoder

Convolutional codes

- Viterbi decoding

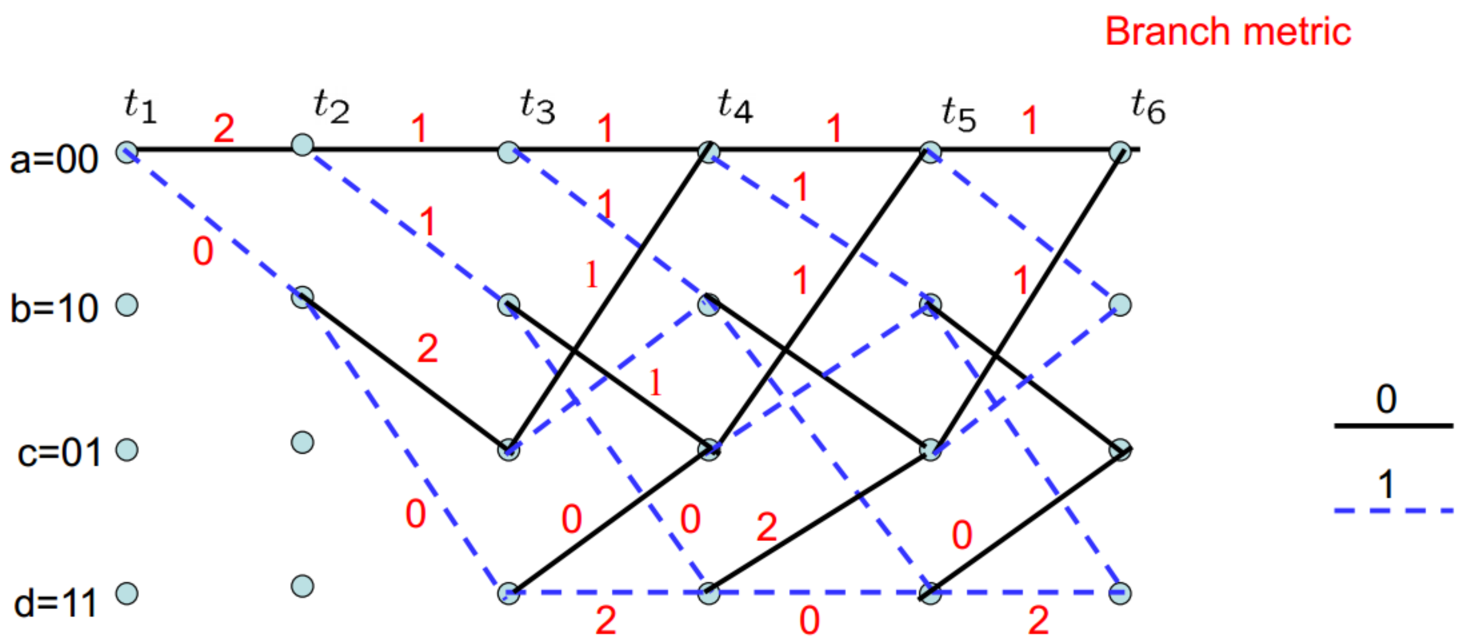
Andrew Viterbi
(1935-)

- BS & MS in MIT
- PhD in University of Southern California
- Invention of Viterbi algorithm in 1967
- Co-founder of Qualcomm Inc. in 1983

Convolutional codes

- Viterbi decoding
  - ➢ Consider R=1/2, L=3 for example.

Input data sequence **m**:	1	1	0	1	1	…
Coded sequence **U**:	11	0 1	01	00	01	…
Received sequence **Z**:	11	0 1	01	10	01	…

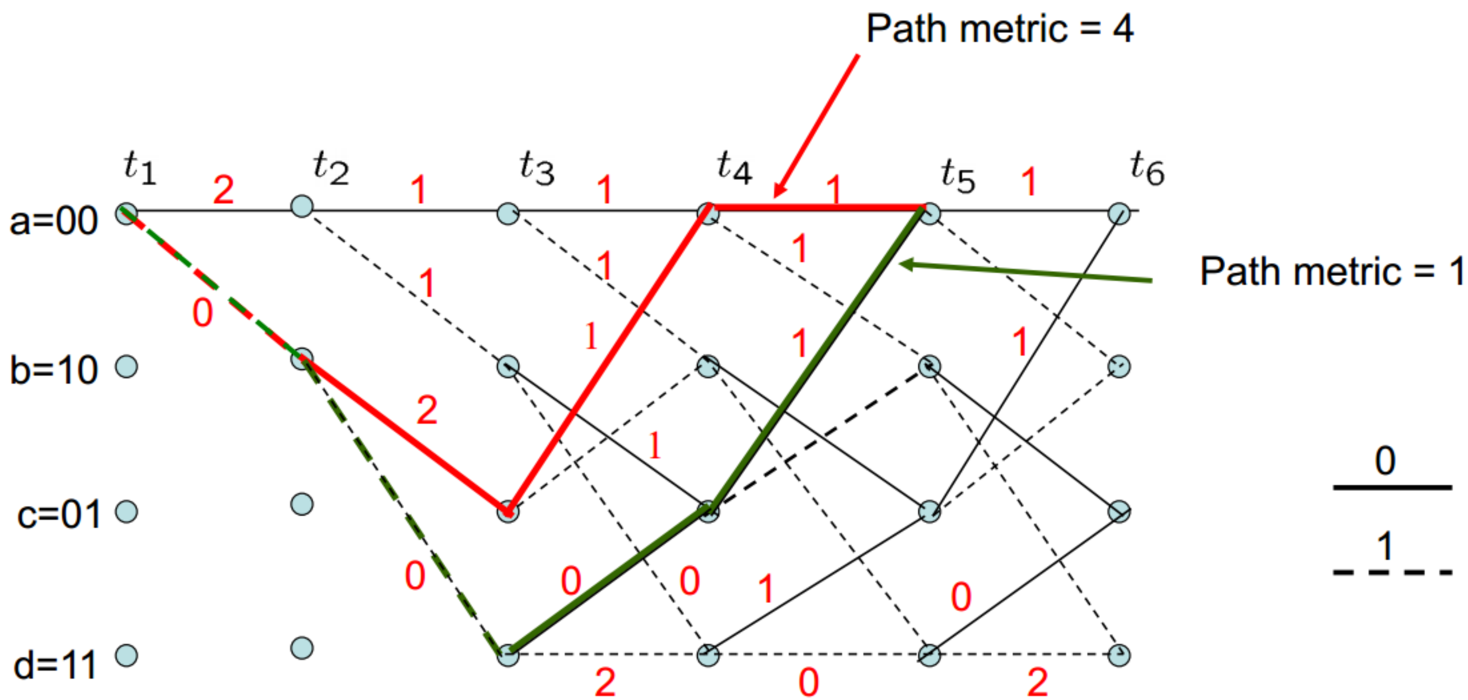


Convolutional codes

- Viterbi decoding
  - Basic idea: If any 2 paths in the trellis merge to a single state, one of them can always be eliminated in the search
  - Let cumulative path metric of a given path at t_i=sum of the branch metrics along that path up to time t_i
  - Consider t_5
    1. The upper path metric is 4, the lower path metric is 1
    2. The upper path metric cannot be path of the optima path since the lower path has a lower metric
    3. This is because future output branches depend on the current state and not the previous state

Convolutional codes

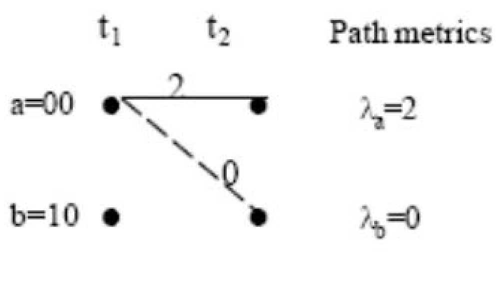
- Viterbi decoding



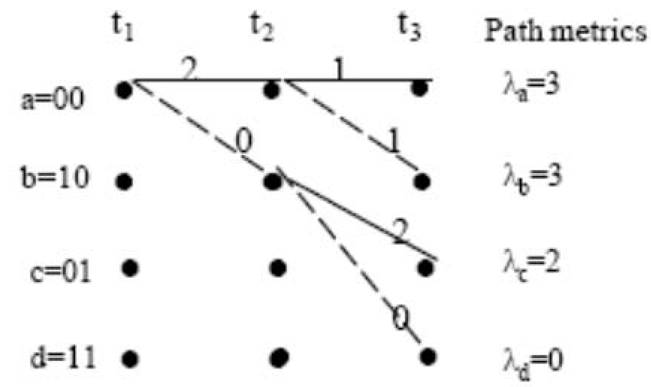
Convolutional codes

- Viterbi decoding
  - At time t_i, there are 2^{L}-1 states in the trellis
  - Each state can be entered by means of 2 states
  - Viterbi decoding consists of computing the metric of the 2 paths entering each state and eliminating one of them
  - This is done for each of the 2^{L}-1nodes at time ti
  - The decoder then moves to time t_{i+1} and repeat the process

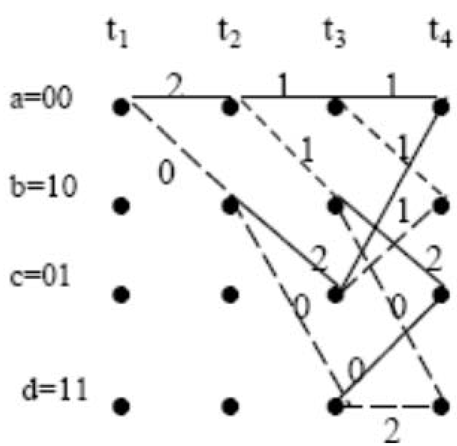
- Viterbi decoding
  - ➢ Example.

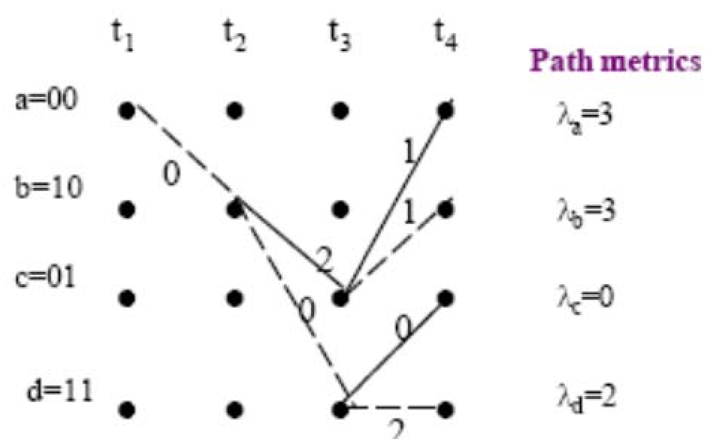


(a)



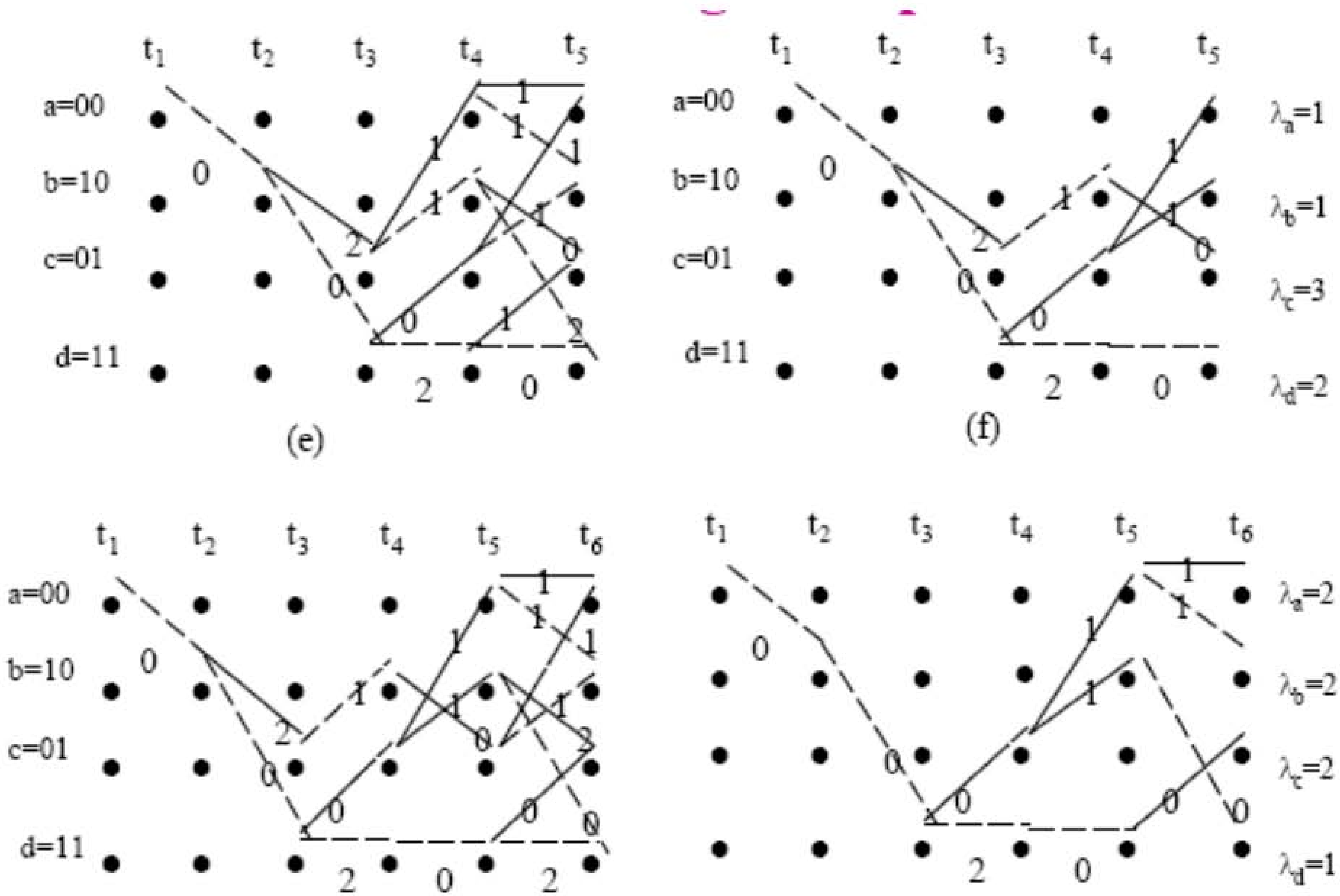
(b)





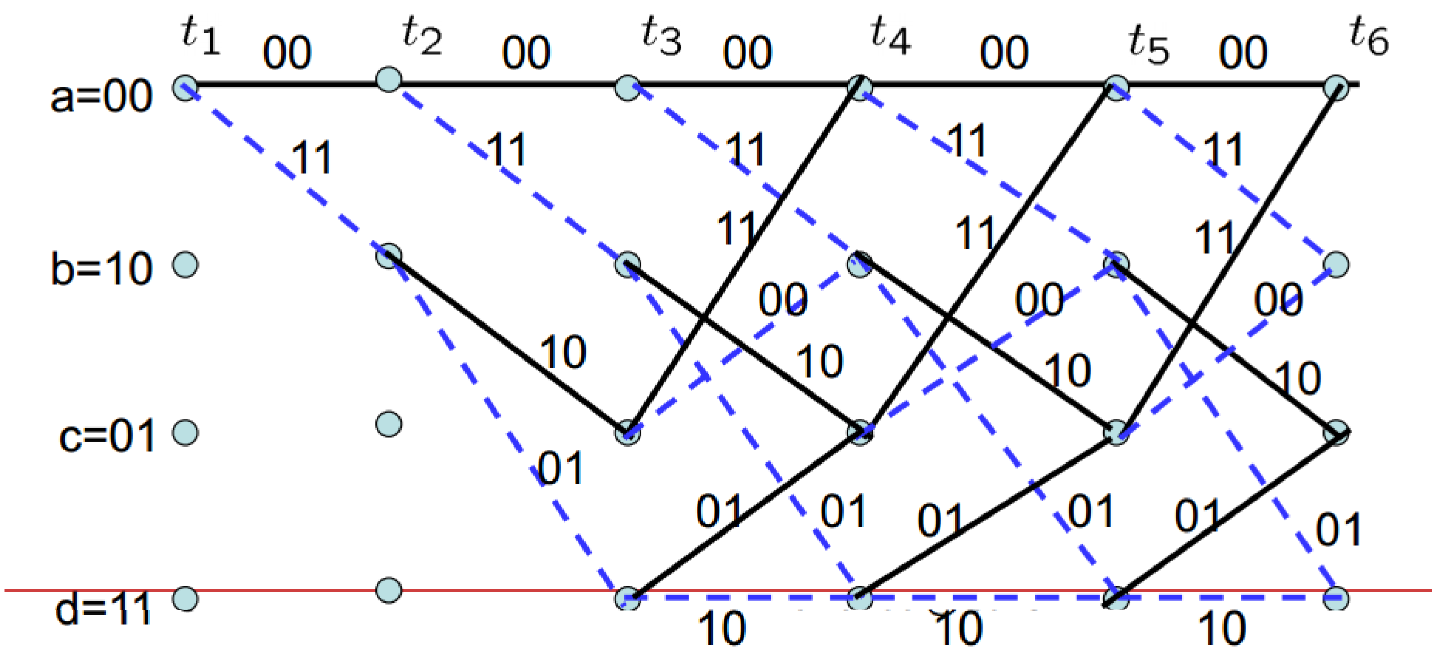
Convolutional codes

- Viterbi decoding
  - Example.



Convolutional codes

- Viterbi decoding
  - d_{free}=Minimum free distance=Minimum distance of any pair of arbitrarily long paths that diverge and remerge
  - A code can correct any t channel errors where (this is an approximation) $t \leq \lfloor \frac{d_{\text{free}} - 1}{2} \rfloor$

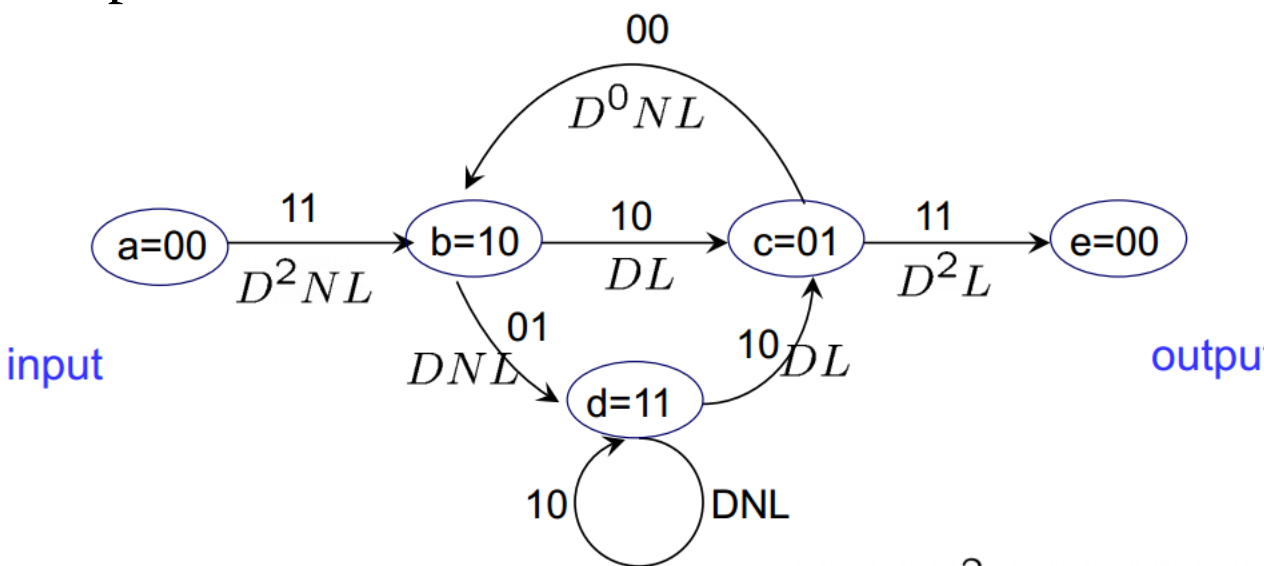


Convolutional codes

- Transfer function
  - The distance properties and the error rate performance of a convolutional code can be obtained from its transfer function
  - Since a convolutional code is linear, the set of Hamming distances of the code sequences generated up to some stages in the trellis, from the all-zero code sequence, is the same as the set of distances of the code sequences with respect to any other code sequence
  - Thus, we assume that the all-zero path is the input to the encoder

Convolutional codes

- Transfer function
  - State diagram labeled according to distance from all-zero path



D^m denote m non-zero output bits

N if the input bit is non-zero

L denote a branch in the path

$$
\begin{cases}
X_b = D^2NLX_a + LNX_c \\
X_c = DLX_b + DLX_d \\
X_d = DNLX_b + DNLX_d \\
X_e = D^2LX_c
\end{cases}
$$

Convolutional codes

- Transfer function
  - The transfer function T(D,N,L), also called the wieght enumerating function of the code is

$$T(D,N,L) = \frac{X_e}{X_a}$$

  - By solving the state equations we get

$$T(D,N,L) = \frac{D^5NL^3}{1-DNL(1+L)}$$

$$= D^5NL^3 + D^6N^2L^4(1+L) + D^7N^3L^5(1+L)^2 + \ldots + D^{l+5}N^{l+1}L^{l+3}(1+L)^l + \ldots$$

  - The transfer functions indicates that

1. There is one path at distance 5 and length 3, which differs 1 bit from the correct all-zeros path
2. There are 2 paths at distance 6, one of which is of length 4, the other length 5, and both differ in 2 input bits from all-zeros path
3. $d_{\text{free}} = 5$

Convolutional codes

- Good convolutional codes
  - Good convolutional codes can only be found in general by computer search
  - They are listed in tables and classified by their constraint length, code rate, and their generator polynomials or vectors (typically using octal notation).
  - The error-correction capability of a convolutional code incrases as n increases or as the code rate decreases.
  - Thus, the channel bandwidth and decoder complexity increases.

Convolutional codes

- Good convolutional codes
  - Rate 1/2.

Constraint Length	Generator Polynomials	d_{free}
3	(5,7,7)	8
4	(13,15,17)	10
5	(25,33,37)	12
6	(47,53,75)	13
7	(133,145,175)	15
8	(225,331,367)	16
9	(557,663,711)	18
10	(1117,1365,1633)	20

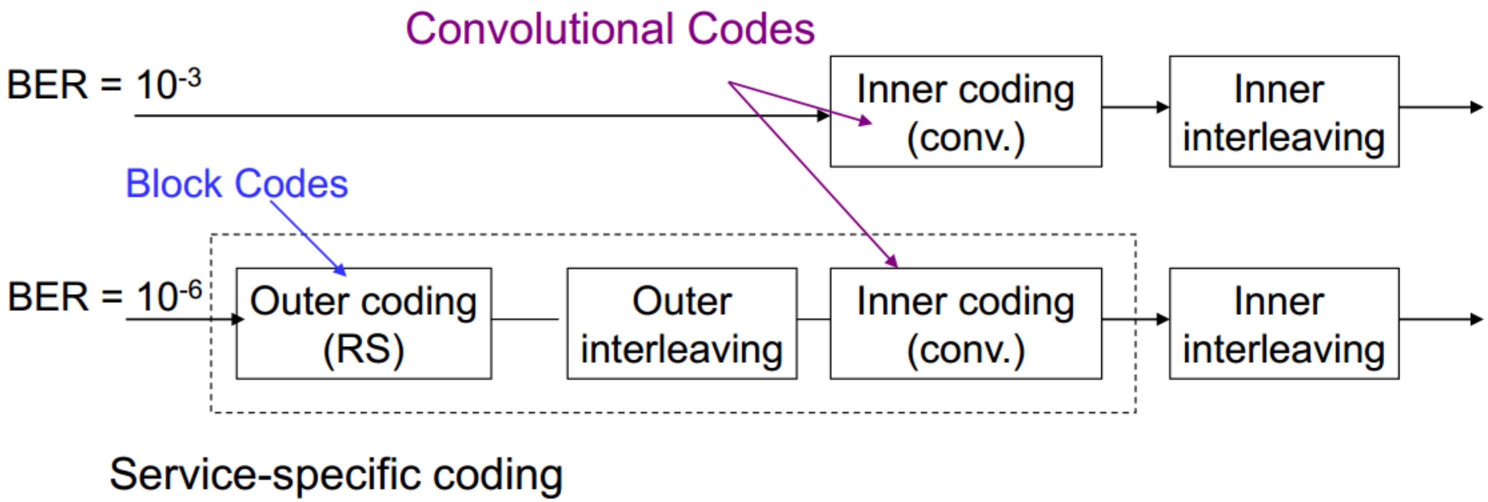
Convolutional codes

- Good convolutional codes
  - Rate 1/3.

Constraint Length	Generator Polynomials	d_{free}
3	(5,7)	5
4	(15,17)	6
5	(23,35)	7
6	(53,75)	8
7	(133,171)	10
8	(247,371)	10
9	(561,753)	12
10	(1167,1545)	12

Convolutional codes

- Channel coding for Wideband CDMA



Convolutional code is rate 1/3 and rate ½, all with constraint length 9

Convolutional codes

- Channel coding for Wireless LAN (IEEE 802.11a)

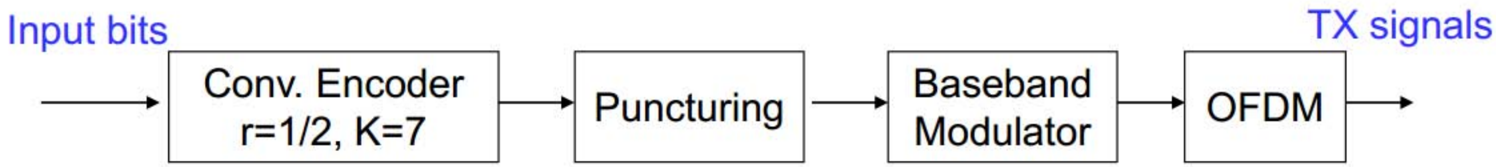


Table 11-3. Encoding details for different OFDM data rates

Speed (Mbps)	Modulation and coding rate (R)	Coded bits per carrier[a]	Coded bits per symbol	Data bits per symbol[b]
6	BPSK, R=1/2	1	48	24
9	BPSK, R=3/4	1	48	36
12	QPSK, R=1/2	2	96	48
18	QPSK, R=3/4	2	96	72
24	16-QAM, R=1/2	4	192	96
36	16-QAM, R=3/4	4	192	144
48	64-QAM, R=2/3	6	288	192
54	64-QAM, R=3/4	6	288	216

Convolutional codes

- Other advanced channel coding
  - Low density parity check codes: Robert Gallager 1960
  - Turbo codes: Berrou et al. 1993
  - Trellis-coded modulation: Ungerboeck 1982
  - Space-time coding: Vahid Tarokh et al. 1998
  - Polar codes: Erdal Arkan 2009

Check the latest coding techniques in 5G standards