Lecture 10: Convolutional codes.

Course instructor: Alexey Frolov

al.frolov@skoltech.ru

Teaching Assistant: Stanislav Kruglik

stanislav.kruglik@skolkovotech.ru

February 21, 2017

Outline

Introduction

The class of convolutional codes was invented by Elias in 1955 and has been widely in use for wireless, space, and broadcast communications since about 1970. Their popularity stems from the relative ease with which the *maximum-likelihood sequence decoder* may be implemented and from their effectiveness when concatenated with a ReedSolomon code.

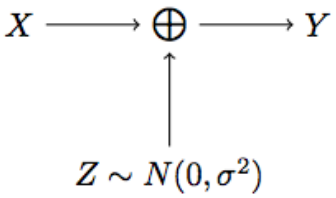
Outline

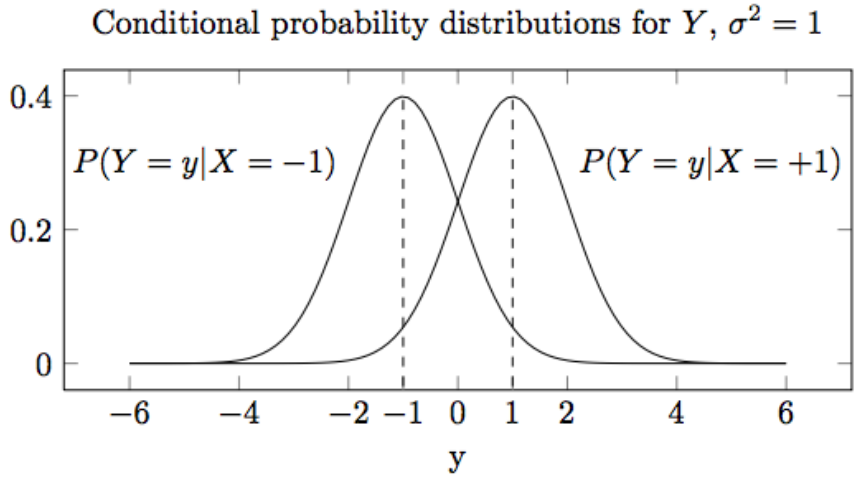
Soft decoding

In case of *hard decision decoding* demodulator makes a hard decision and provides the decoder with the symbols of the field. In case of *soft decision decoding* demodulator provides the decoder with probability distributions.

Example: Binary input AWGN channel

$X \in \{-1, +1\}$



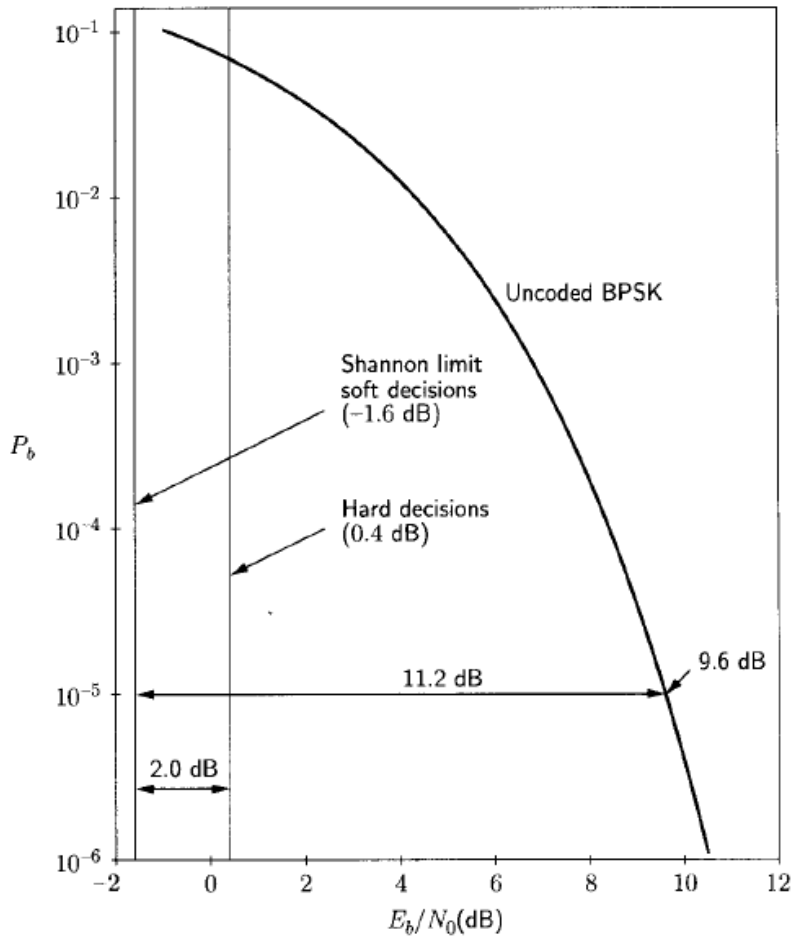


LLR calculation

Assume again, that 0 is sent with the level $+1$ and 0 is sent with the level -1. Let us calculate the log likelihood ratio given y

$$\begin{aligned} LLR &= \log \frac{\Pr(X=+1|y)}{\Pr(X=-1|y)} \\ &= \log \frac{\Pr(y|X=+1)\Pr(X=+1)\Pr(y)}{\Pr(y|X=-1)\Pr(X=-1)\Pr(y)} \\ &= \log \frac{f(y|+1)}{f(y|-1)} \\ &= \frac{(y+1)^2-(y-1)^2}{2\sigma^2} \\ &= \frac{2}{\sigma^2}y \end{aligned}$$

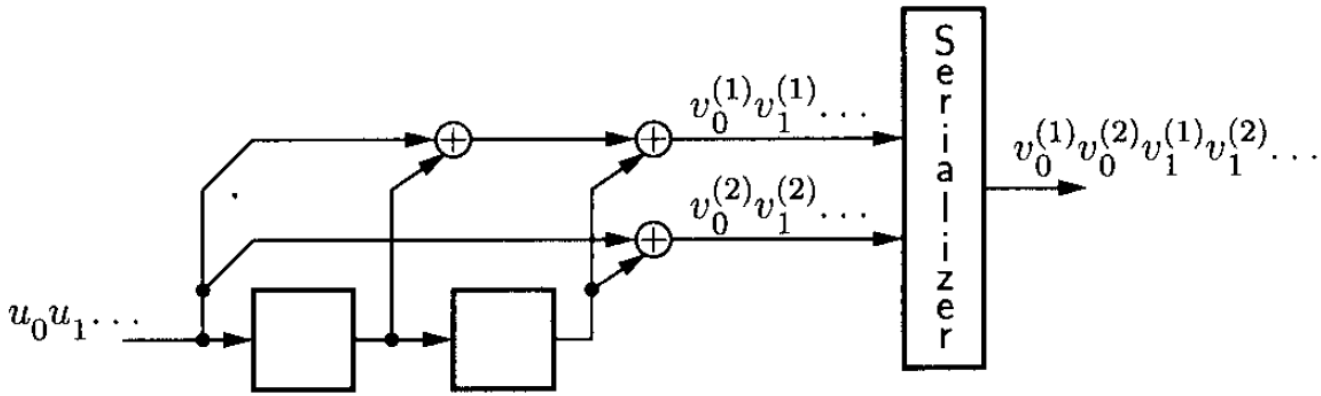
Soft vs hard decoding



Consider again BI-AWGN channel. We can use LLR values or make a hard decision ($+1$ if $LLR > 0$, -1 otherwise). The difference is approx. 2 dB!

Outline

First look at convolutional codes



An encoder for binary rate $R = 1/2$ convolutional code.

First look at convolutional codes

Infinite sequence of information digits

$$\mathbf{u} = u_0 u_1 \dots$$

The register length is called a *memory* of convolutional code.

Two output sequences (also infinite)

$$\mathbf{v}^{(1)} = v_0^{(1)} v_1^{(1)} \dots$$

and

$$\mathbf{v}^{(2)} = v_0^{(2)} v_1^{(2)} \dots$$

General rate $R = b/c$ convolutional encoder

Information sequence:

$$\mathbf{u} = \mathbf{u}_0\mathbf{u}_1 \ldots = u_0^{(1)} u_0^{(2)} \ldots u_0^{(b)} u_1^{(1)} u_1^{(2)} \ldots u_1^{(b)} \ldots$$

Encoded sequence:

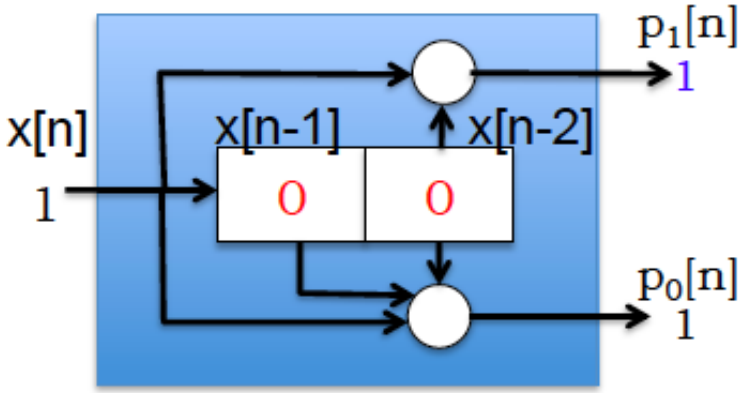
$$\mathbf{v} = \mathbf{v}_0\mathbf{v}_1 \ldots = v_0^{(1)} v_0^{(2)} \ldots v_0^{(c)} v_1^{(1)} v_1^{(2)} \ldots v_1^{(c)} \ldots,$$

where

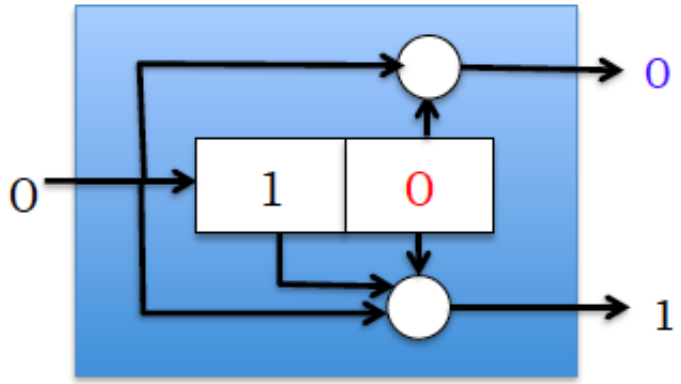
$$\mathbf{v}_t = f\big(\mathbf{u}_t, \mathbf{u}_{t-1}, \ldots, \mathbf{u}_{t-m}\big).$$

Recall, that m is the encoder memory.

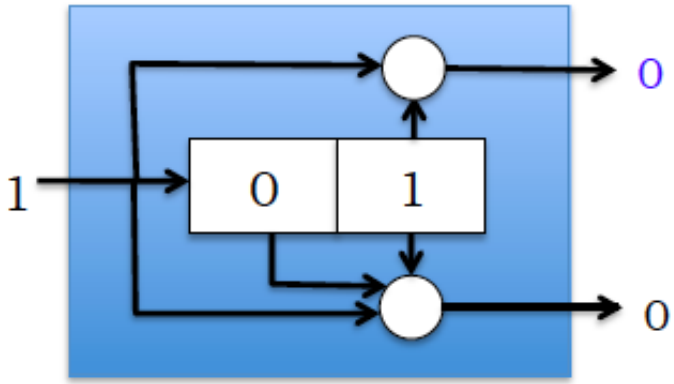
Example: transmit message 1011



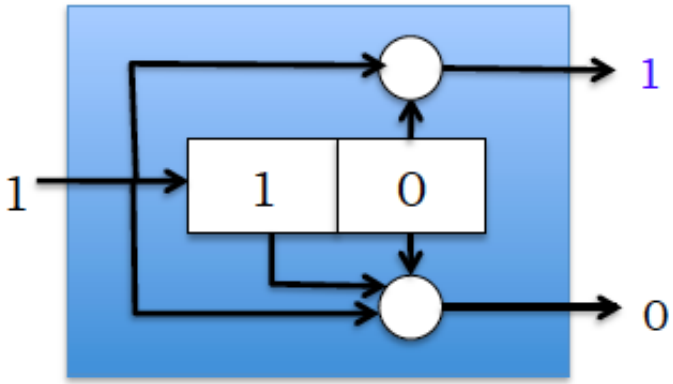
Processing x[0]



Processing x[1]



Processing x[2]



Processing x[3]

General rate $R = b/c$ convolutional encoder

The function f is a linear function (our requirement), thus

$$\mathbf{v}_t = \mathbf{u}_t G_0 + \mathbf{u}_{t-1} G_1 + \ldots, + \mathbf{u}_{t-m} G_m,$$

where G_i, $0 \leq i \leq m$ is a binary $b \times c$ matrix.

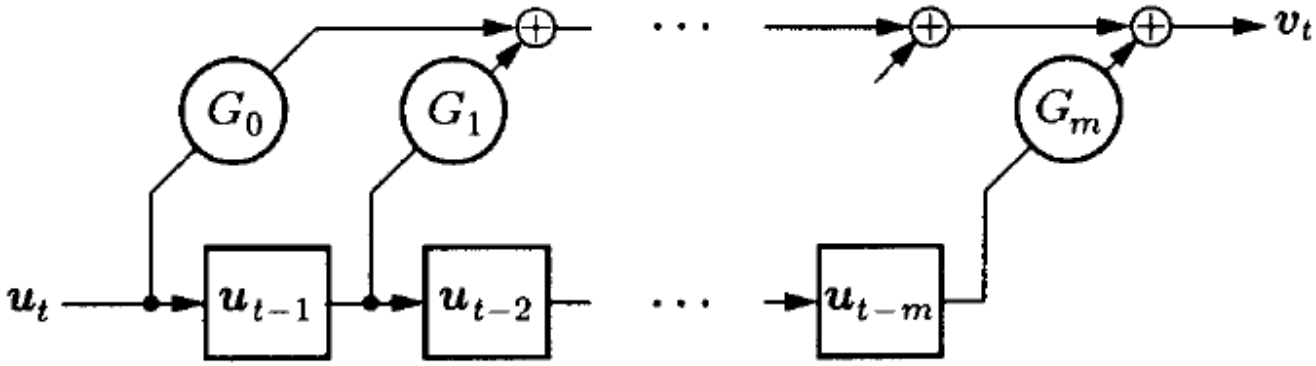
The encoding can be presented as follows

$$\mathbf{v} = \mathbf{u}\mathbf{G}.$$

$$\boldsymbol{G} = \begin{pmatrix} G_0 & G_1 & \ldots & G_m & & \\ & G_0 & G_1 & \ldots & G_m & \\ & & \ddots & \ddots & & \ddots \end{pmatrix}$$

Generator matrix

G is called a generator matrix, G_i, $0 \leq i \leq m$, – generator submatrices.



Example

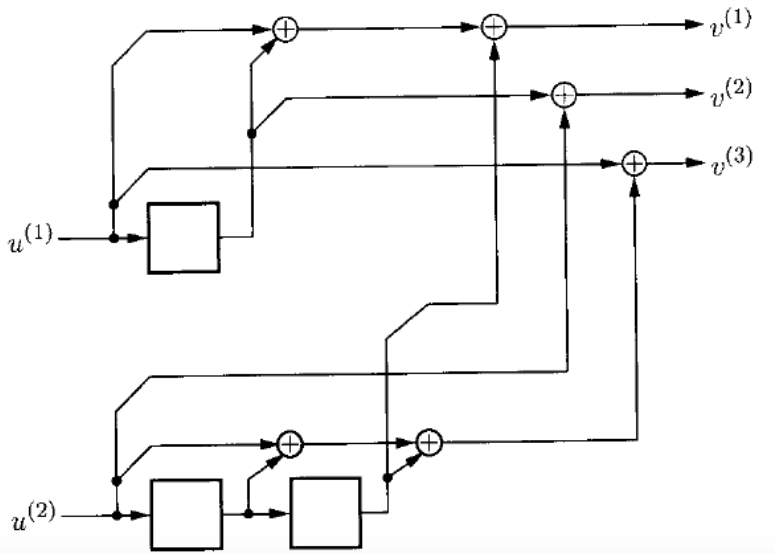
$$G_0 = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

$$G_1 = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

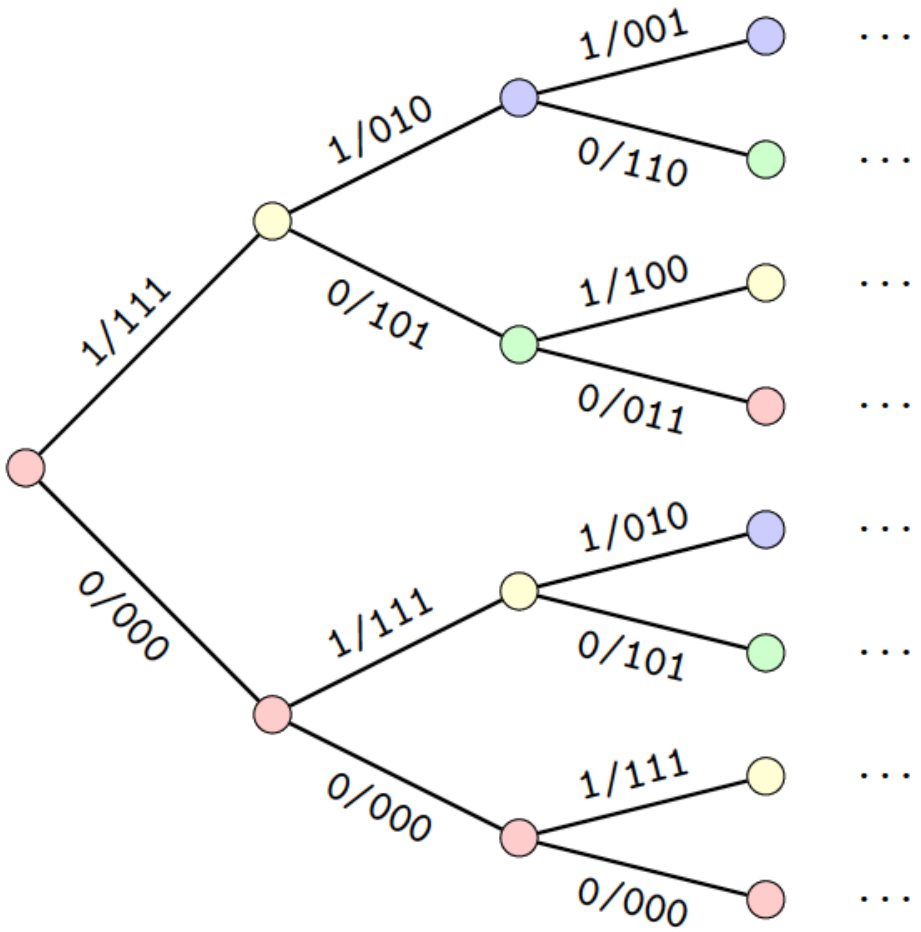
$$G_2 = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \end{pmatrix}$$

The generator matrix is

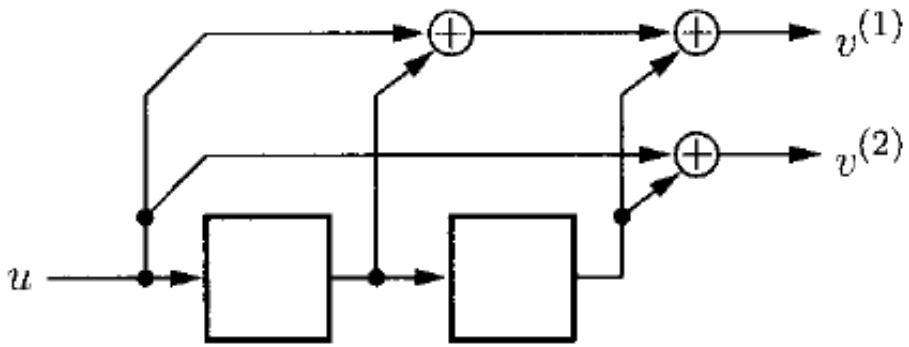
$$G = \begin{pmatrix} 101 & 110 & 000 & & \\ 011 & 001 & 101 & & \\ & 101 & 110 & 000 & \\ & 011 & 001 & 101 & \\ & & \ddots & \ddots & \ddots \end{pmatrix}$$

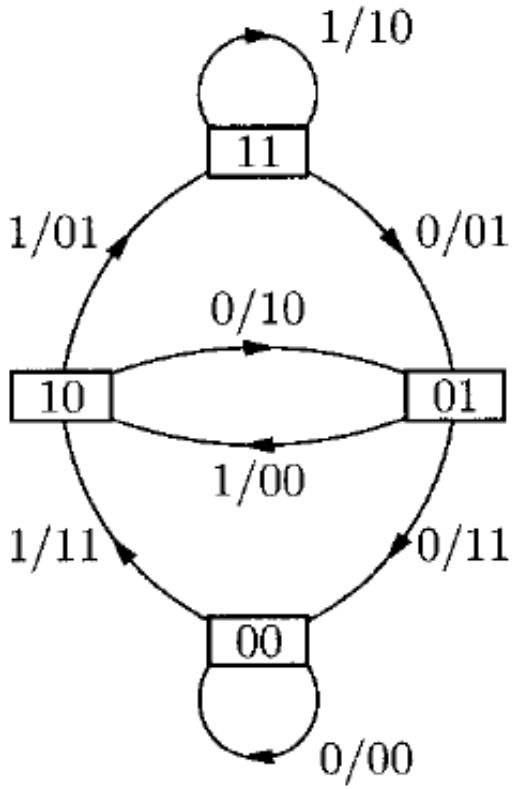


Code tree

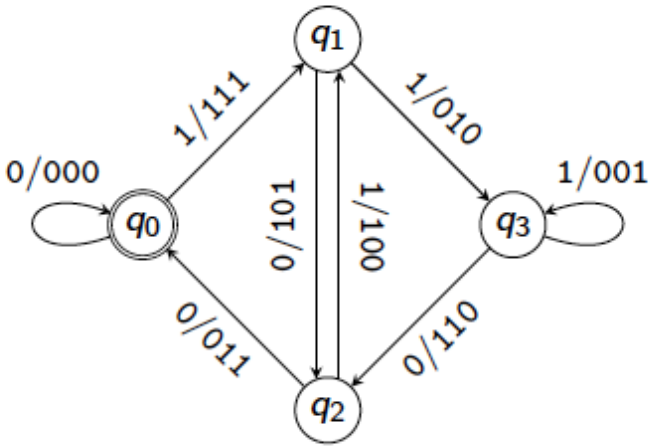

1/111
0/000
1/010
0/101
1/001
0/110
1/100
0/011
1/111
0/000
1/010
0/101
1/111
0/000
...

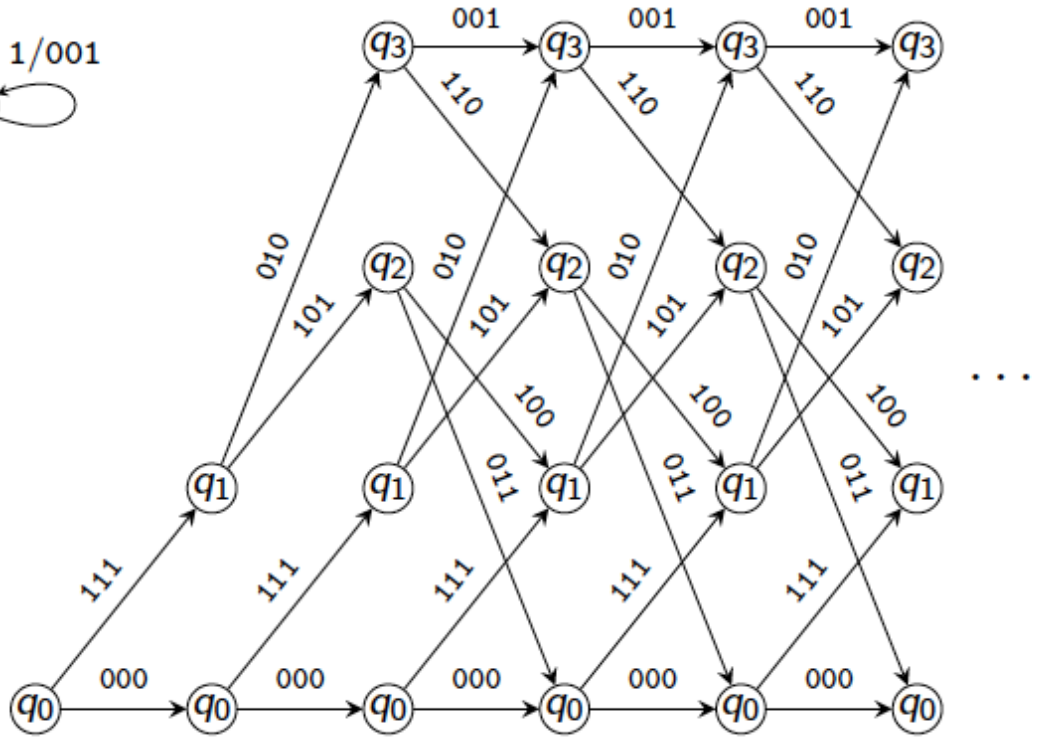
Finite state machine representation



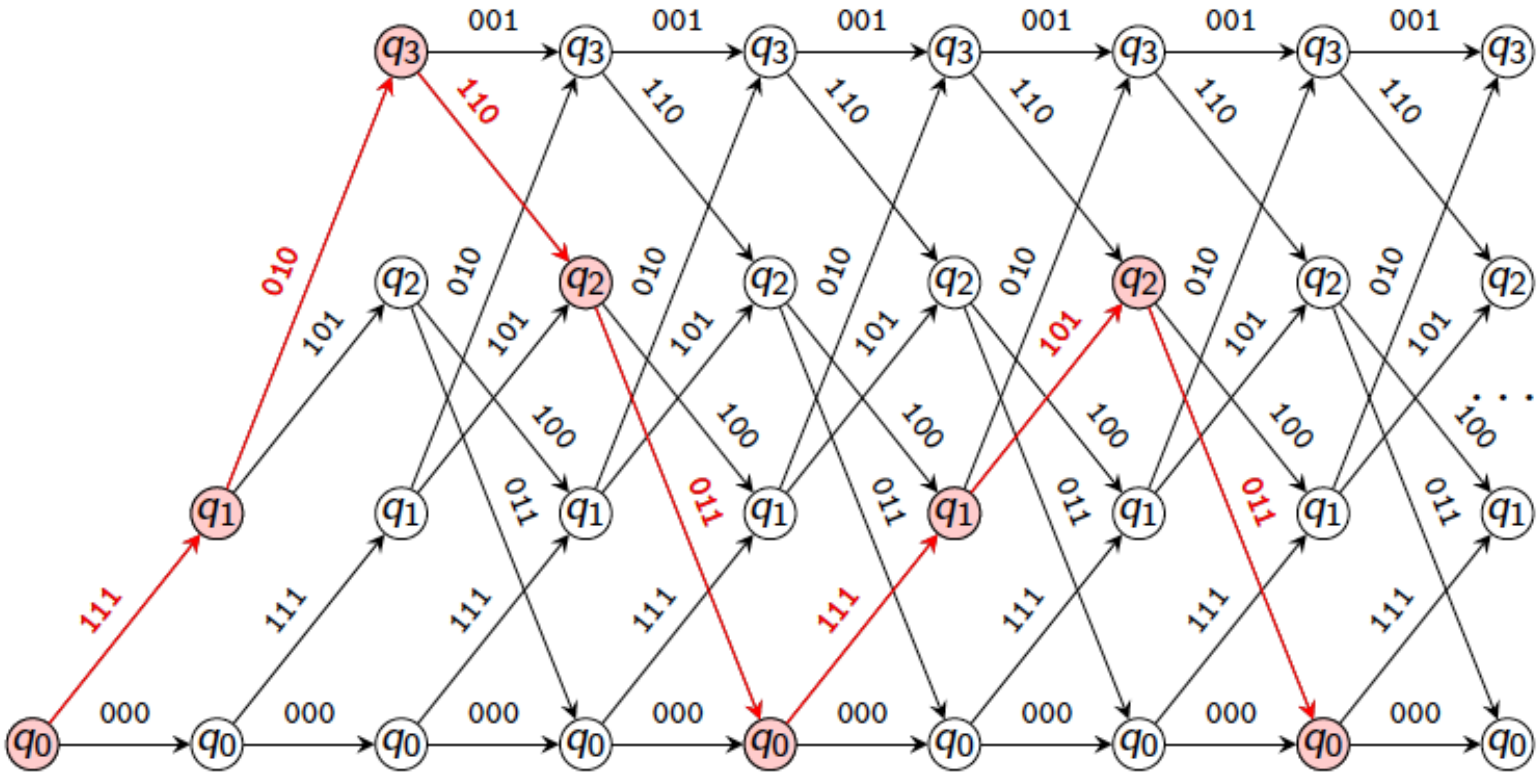


Trellis


q_1
1/111
1/010
0/000
1/001
q_0
0/101
1/100
q_3
0/011
0/110
q_2


q_3
001
110
010
101
100
011
111
000
q_2
q_1
q_0
...

Example



$$1100100\ldots \quad \mapsto \quad 111\ 010\ 110\ 011\ 111\ 101\ 011\ldots$$

Outline

Termination methods

1. Direct termination
2. Zero termination
3. Tail biting

Outline

Algebraic description

Convolutional codes are often thought of as nonblock codes over a finite field, but it can be an advantage to treat them as block codes over a finite field.

Main definitions

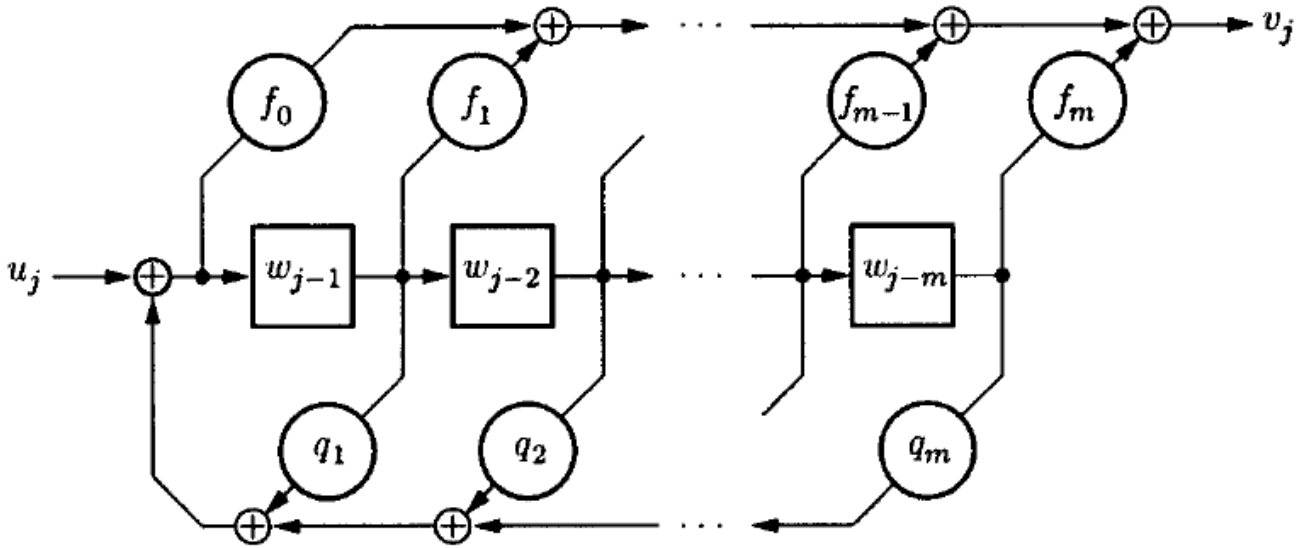
D - delay operator.

Representation of input and output sequences (D-transform)

$$\mathbf{u}(D) = \ldots \mathbf{u}_{-1}(D^{-1}) + \mathbf{u}_0 + \mathbf{u}_1 D + \mathbf{u}_2 D^2 + \ldots$$

$$\mathbf{v}(D) = \ldots \mathbf{v}_{-1}(D^{-1}) + \mathbf{v}_0 + \mathbf{v}_1 D + \mathbf{v}_2 D^2 + \ldots$$

Algebraic description



$$\mathbf{v}(D) = \mathbf{u}(D)\frac{f(D)}{q(D)} = \mathbf{u}(D)\frac{f_0 + f_1 D + f_m D^m}{1 + q_1 D + \ldots + q_m D^m}.$$

Main definitions

Definition

Let

$$\mathbb{F}[D] = \left\{ \sum_{i=r}^{d} x_i D^i : x_i \in \mathbb{F}, 0 \leq r \leq d < \infty \right\}$$

be the set of binary formal polynomials.

Definition

Let

$$\mathbb{F}(D) = \left\{ \frac{p(D)}{q(D)} : p(D), q(D) \in \mathbb{F}[D] \right\}$$

be the set of binary rational functions.

Main definitions

Definition

Let

$$\mathbb{F}[[D]] = \left\{ \sum_{i=r}^{\infty} x_i D^i : x_i \in \mathbb{F}, 0 \leq r \right\}$$

be the set of formal power series.

Definition

Let

$$\mathbb{F}((D)) = \left\{ \sum_{i=r}^{\infty} x_i D^i : x_i \in \mathbb{F}, r \in \mathbb{Z} \right\}$$

be the set of binary Laurent series.

Main definitions

Definition

Given $x(D) \in \mathbb{F}((D))$, its delay $del(x(D))$ is the smallest r for which $x_r = 1$.

Definition

Given $x(D)$ is called delay free if $del(x(D)) = 0$.

Definition

A rational function is called realizable if $q(D)$ is delay free.

Convolutional code

A rate $R = b/c$ binary convolutional encoder is a linear mapping

$$\Phi : \mathbb{F}^b((D)) \to \mathbb{F}^c((D)),$$

which can be represented as

$$\mathbf{v}(D) = \mathbf{u}(D)G(D),$$

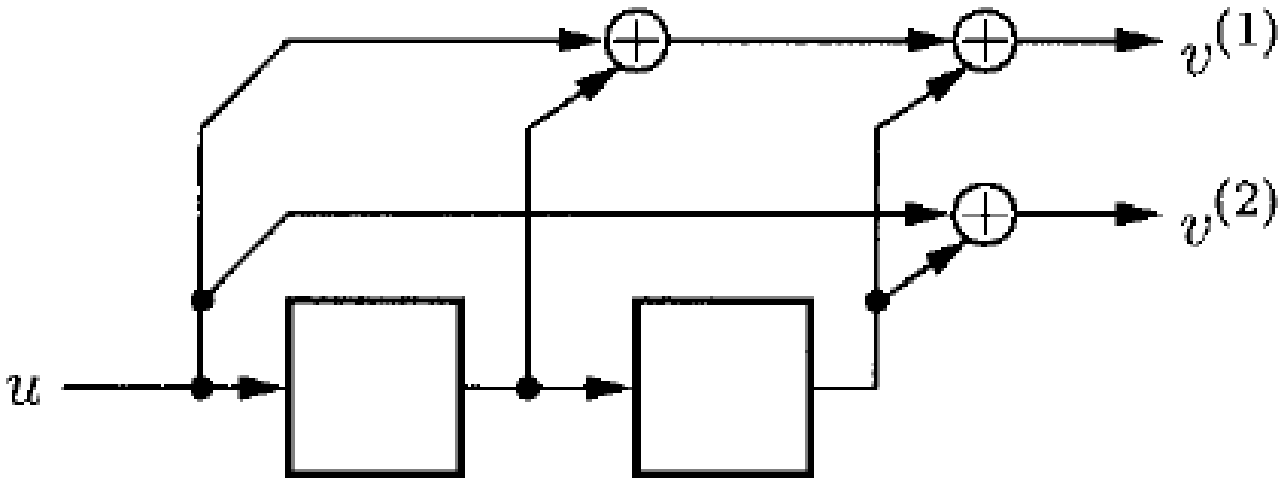
where $G(D)$ is of size $b \times c$ with entries from $\mathbb{F}(D)$.

Definition

$G(D)$ is a polynomial generator matrix if all the elements are realizable.

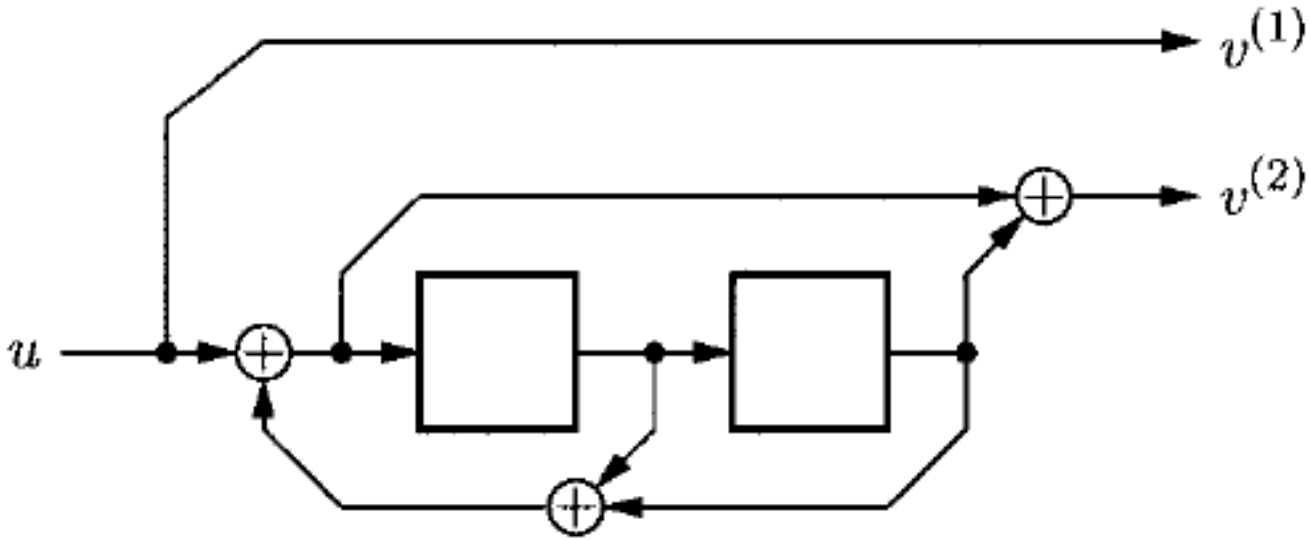
Example

$$G(D) = (1 + D + D^2, 1 + D^2)$$



Systematic encoder

$$G(D) = (1, \frac{1 + D^2}{1 + D + D^2})$$



Outline

Free distance

Free distance

$$d_f = \min_{\gamma} ||\gamma||,$$

where γ is a path on trellis with initial and final states equal to q_0.

Outline

Bit-wise MAP vs Block-wise MAP decoding

Block-wise MAP decoding

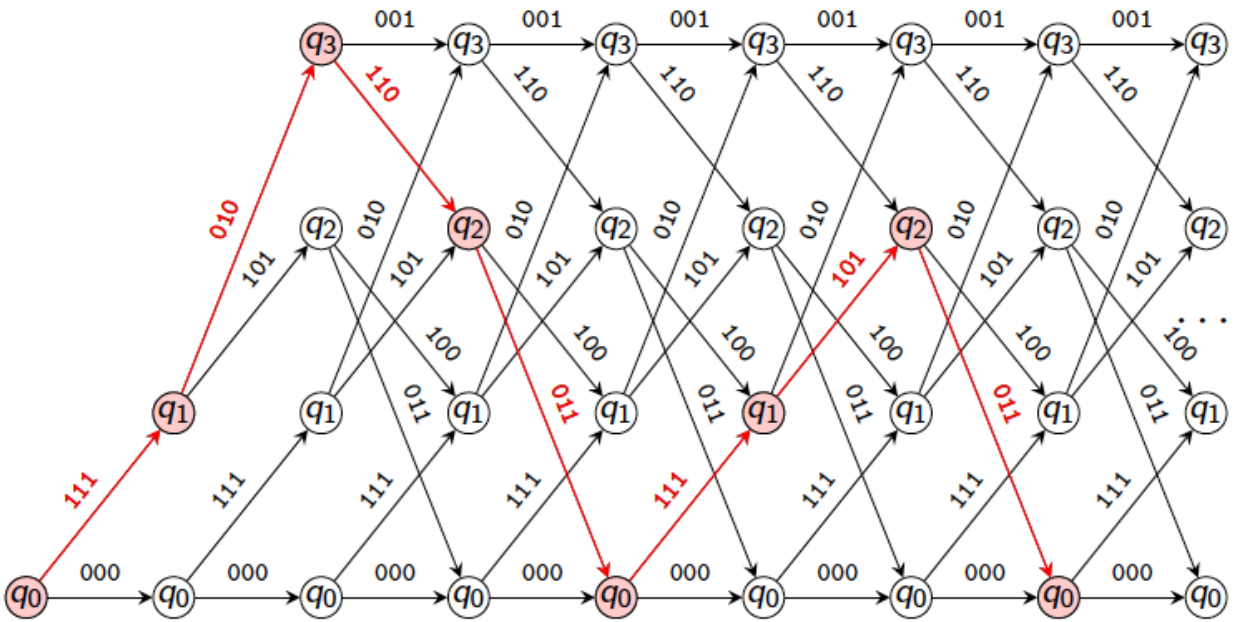
$$\hat{x}^{(MAP)}(y) = \arg\max_{x \in \mathcal{C}} \Pr(x|y).$$

Bit-wise MAP decoding

$$\hat{x_i}^{(MAP)}(y) = \arg\max_{x_i \in \{+1,-1\}} \sum_{x \in \mathcal{C}, x_i \text{ is fixed}} \Pr(x|y).$$

Viterbi algorithm

Again consider a trellis. Assume we use a zero-termination of length n, so we should only consider paths which start in state q_0 and end in q_0 (one of such paths is shown below)



$$1100100\ldots \quad \mapsto \quad 111\ 010\ 110\ 011\ 111\ 101\ 011\ldots$$

Viterbi algorithm

Assume we send a codeword $\mathbf{x} \in \mathbb{F}_2^n$ and received a sequence $\mathbf{y}$.

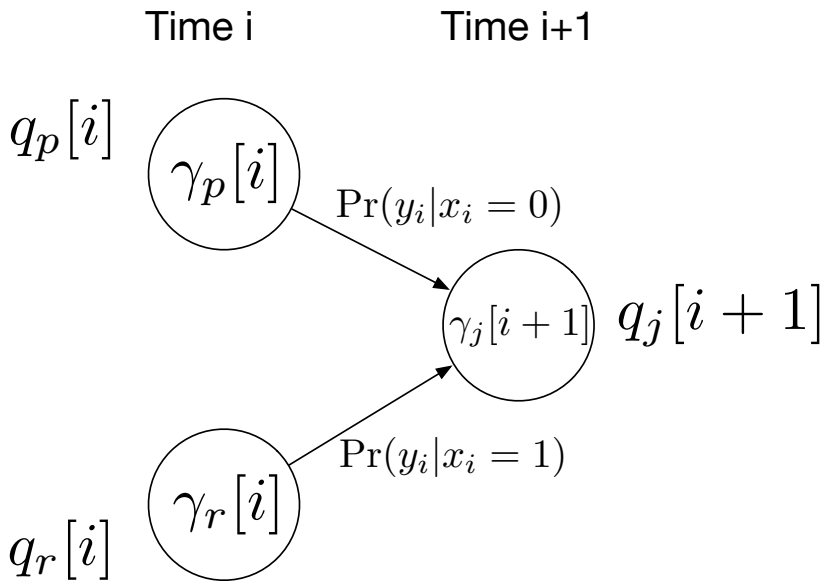
Recall, that the channel is memoryless, then

$$\Pr(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{n} \Pr(y_i|x_i).$$

Label the edges of trellis with $\Pr(y_i|x_i)$ values.

Dynamic programming rule

Viterbi algorithm is a instance of dynamic programming. Consider trellis states at time $i+1$.



$$\gamma_j[i+1] = \max\{\gamma_p[i]\Pr(y_i|x_i=0), \gamma_r[i]\Pr(y_i|x_i=1)\}$$

Thank you for your attention!