Lecture 4: Linear Codes

Linear codes over $\mathbb{F}_q$

- We let $\mathcal{X} = \mathbb{F}_q$ for some prime power q. Most important case: $q = 2$ (binary codes).
- Without loss of generality, we may represent the information message as a sequence of k symbols from $\mathbb{F}_q$.
- We have $|\mathcal{C}| = q^k$, and $R = \frac{k}{n} \log_2 q$ bits/symbol.

Definition 22. *A (q^k, n) block code over $\mathcal{X} = \mathbb{F}_q$ is called a linear (n, k) code if its codewords form a k-dimensional vector subspace of the vector space $\mathbb{F}_q^n$.* $\Diamond$

Elementary properties of linear codes

- The code $\mathcal{C}$ is an additive group, in particular, if $\mathbf{c}, \mathbf{c}' \in \mathcal{C}$ then $\mathbf{c} + \mathbf{c}' \in \mathcal{C}$ and $-\mathbf{c} \in \mathcal{C}$.
- The all-zero vector is a codeword: $\mathbf{0} \in \mathcal{C}$.
- Linear combination of codewords are codewords: $\mathbf{c}_1, \ldots, \mathbf{c}_\ell \in \mathcal{C}$ and $a_1, \ldots, a_\ell \in \mathbb{F}_q$, then

$$a_1\mathbf{c}_1 + \cdots a_\ell\mathbf{c}_\ell \in \mathcal{C}$$

- There exist (non-unique) sets of k linearly independent codewords that generate the whole code, i.e.,

$$\mathcal{C} = \left\{ \sum_{\ell=0}^{k-1} u_\ell \mathbf{g}_\ell : u_0, \ldots, u_{k-1} \in \mathbb{F}_q \right\}$$

where $\mathbf{g}_0, \ldots, \mathbf{g}_{k-1}$ are codewords that form a basis for the code $\mathcal{C}$.

Generator matrix

- We can arrange the basis as rows of a $k \times n$ matrix

$$\mathbf{G} = \begin{bmatrix} g_{0,0} & g_{0,1} & g_{0,2} & \cdots & g_{0,n-1} \\ g_{1,0} & g_{1,1} & g_{1,2} & \cdots & g_{1,n-1} \\ \vdots & & & & \vdots \\ g_{k-1,0} & g_{k-1,1} & g_{k-1,2} & \cdots & g_{k-1,n-1} \end{bmatrix}$$

This is called a generator matrix for the code. Letting $\mathbf{u} = (u_0, \ldots, u_{k-1})$ we can write the encoding function as

$$\mathbf{c} = \mathbf{u}\mathbf{G}$$

$(7, 4)$ binary Hamming code

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Repetition and single-parity-check codes

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

$$\mathbf{G} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

Systematic encoding

- Sometimes, a desirable property of the encoding function is that the vector of information symbols appears explicitly as part of the codeword.

- For linear codes, a generator matrix in systematic form is given by

$$\mathbf{G} = [\mathbf{P}|\mathbf{I}_k]$$

where $\mathbf{P} \in \mathbb{F}_q^{k \times (n-k)}$ and $\mathbf{I}_k$ denotes the $k \times k$ identity.

- In this way, we have $\mathbf{c} = \mathbf{uG} = [\mathbf{uP}|\mathbf{u}]$.

Parity-check matrix

- Since $\mathcal{C}$ is a vector subspace of $\mathbb{F}_q$ of dimension k, then it can be seen as the Kernel of some linear transformation $\mathbf{F}_q^n \rightarrow \mathbf{F}_q^{n-k}$.

- The matrix of such linear transformation is called parity-check matrix and it is denoted by $\mathbf{H}^\mathsf{T}$, where $\mathbf{H} \in \mathbb{F}_q^{(n-k)\times n}$:

$$\mathcal{C} = \left\{ \mathbf{c} \in \mathbb{F}_q^n : \mathbf{c}\mathbf{H}^\mathsf{T} = \mathbf{0} \right\}$$

- In particular, the rows of $\mathbf{H}$ are n-vectors in $\mathbb{F}_q^n$ that are orthogonal to all codewords. Indeed, they generate the orthogonal subspace $\mathcal{C}^\perp$, also known as the dual code of $\mathcal{C}$.

- If $\mathbf{G} = [\mathbf{P}|\mathbf{I}_k]$, then $\mathbf{H} = [\mathbf{I}_{n-k}| - \mathbf{P}^\mathsf{T}]$, in fact, we have

$$\mathbf{G}\mathbf{H}^\mathsf{T} = [\mathbf{P}|\mathbf{I}_k] \begin{bmatrix} \mathbf{I}_{n-k} \\ -\mathbf{P} \end{bmatrix} = \mathbf{P} - \mathbf{P} = \mathbf{0}$$

q-ary symmetric channels

- A q-ary symmetric channel can always be represented as an additive noise channel over $\mathbb{F}_q$, such that

$$\mathbf{y} = \mathbf{c} + \mathbf{z}$$

where $\mathbf{z} \in \mathbb{F}_q^n$.

- The "noise" pmf is given by

$$P_Z(z) = \begin{cases} 1-\delta & \text{for } z = 0 \\ \delta/(q-1) & \text{for } z \neq 0 \end{cases}$$

- The syndrome of the error vector is given by

$$\mathbf{s} = \mathbf{z}\mathbf{H}^{\mathsf{T}}$$

Syndrome decoding

- Notice that the decoder can compute the syndrome even though it does not know $\mathbf{z}$, in fact,

$$\mathbf{y}\mathbf{H}^{\mathsf{T}} = (\mathbf{c}+\mathbf{z})\mathbf{H}^{\mathsf{T}} = \mathbf{c}\mathbf{H}^{\mathsf{T}} + \mathbf{z}\mathbf{H}^{\mathsf{T}} = \mathbf{z}\mathbf{H}^{\mathsf{T}} = \mathbf{s}$$

- Therefore, the syndrome of the error vector is an index that can be used by the decoder to "undo" the bit-flips.

- If $\mathbf{s} = \mathbf{0}$, then $\mathbf{y} \in \mathcal{C}$. In this case, we let $\widehat{\mathbf{c}} = \mathbf{y}$.

- If $\mathbf{s} \neq \mathbf{0}$, then $\mathbf{y} \notin \mathcal{C}$. In this case the decoder knows that an error has occurred (detectable error).

- In order to correct the error, the decoder needs to find $\widehat{\mathbf{z}}$, an estimate of $\mathbf{z}$, and correct the errors as $\widehat{\mathbf{c}} = \mathbf{y} - \widehat{\mathbf{z}}$.

- Unfortunately, the system of equations

$$\mathbf{s} = \mathbf{z}\mathbf{H}^{\mathsf{T}}$$

  where $\mathbf{s}$ is known (the syndrome) and $\mathbf{z}$ is unknown, is underdetermined (n unknowns and $n-k$ equations).

- For every syndrome $\mathbf{s} \in \mathbb{F}_q^{n-k}$, we have q^k possible error vectors.

- We have to solve this problem in a probabilistic sense ... for each set of q^k possible error vectors corresponding to a given syndrome, we shall pick the most likely.

Standard array

- The linear map $\mathbb{F}_q^n \to \mathbb{F}_q^{n-k}$ such that $\mathbf{y} \mapsto \mathbf{s} = \mathbf{y}\mathbf{H}^\mathsf{T}$ has Kernel

$$\mathrm{Ker}(\mathbf{H}^\mathsf{T}) = \mathcal{C}$$

- Linear maps are group homomorphisms (they preserve the group operation, that in this case is componentwise addition in $\mathbb{F}_q$).
- A coset of $\mathcal{C}$ in $\mathbb{F}_q^n$ is given by the translate $\mathbf{v} + \mathcal{C}$ for some $\mathbf{v} \in \mathbb{F}_q^n$.
- The factor group (group of cosets), with respect to the coset addition, satisfies (canonical isomorphism):

$$\mathbb{F}_q^n / \mathcal{C} \equiv \mathbb{F}_q^{n-k}$$

- The standard array is the correspondence between the syndromes $\mathbf{s} \in \mathbb{F}_q^{n-k}$ and the cosets of $\mathcal{C}$ in $\mathbb{F}_q^n$, induced by this isomorphism.

Building the standard array

$\mathbf{s}_0 = \mathbf{0}$	$\mathbf{c}_0 = \mathbf{0}$	$\mathbf{c}_1$	$\cdots$	$\mathbf{c}_{q^k-1}$
$\mathbf{s}_1$	$\mathbf{v}_1$	$\mathbf{v}_1 + \mathbf{c}_1$	$\cdots$	$\mathbf{v}_1 + \mathbf{c}_{q^k-1}$
$\mathbf{s}_2$	$\mathbf{v}_2$	$\mathbf{c}_2 + \mathbf{c}_1$	$\cdots$	$\mathbf{v}_2 + \mathbf{c}_{q^k-1}$
$\vdots$	$\vdots$			$\vdots$
$\mathbf{s}_{q^{n-k}-1}$	$\mathbf{v}_{q^{n-k}-1}$	$\mathbf{v}_{q^{n-k}-1} + \mathbf{c}_1$	$\cdots$	$\mathbf{v}_{q^{n-k}-1} + \mathbf{c}_{q^k-1}$

- For every new row, find the vector in $\mathbb{F}_q^n$ with minimum Hamming weight that not yet appeared in the array.
- The corresponding row is obtained by adding this vector to all codewords.
- All row are distinct, and yield the same syndrome.
- These vectors of minimum weight are called coset leaders.

Example

- Consider the SPC $(3, 2)$, with parity-check matrix

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 \end{bmatrix}$$

- Exercise: build the standard array for the Hamming $(7, 4)$ code.

Error correction algorithm

1. Compute the syndrome $\mathbf{s} = \mathbf{y}\mathbf{H}^{\mathsf{T}}$.

2. Use the standard array and find the most likely error vector $\widehat{\mathbf{z}}$ compatible with $\mathbf{s}$ (coset leader).

3. The minimum Hamming distance decision rule is given by

$$\widehat{\mathbf{c}} = \mathbf{y} - \widehat{\mathbf{z}}$$

4. A (n, k) linear code is able to correct q^{n-k} error patterns (error vectors)

The Tanner Graph of a linear code

- Consider a linear (n, k) block code $\mathcal{C} = \{\mathbf{c} : \mathbf{c}\mathbf{H}^{\mathsf{T}} = \mathbf{0}\}$.
- It is defined by the set of parity-check equations

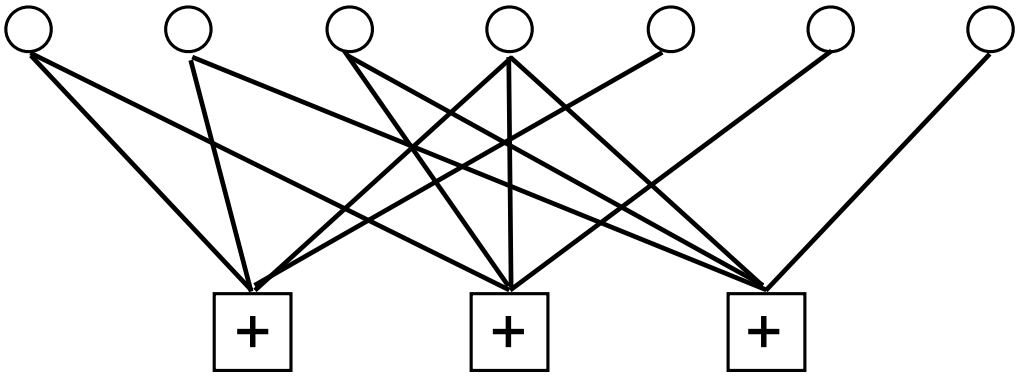
$$\begin{aligned} h_{1,1}x_1 + h_{1,2}x_2 + \cdots + h_{1,n}x_n &= 0 \\ h_{2,1}x_1 + h_{2,2}x_2 + \cdots + h_{2,n}x_n &= 0 \\ &\vdots \\ h_{n-k,1}x_1 + h_{n-k,2}x_2 + \cdots + h_{n-k,n}x_n &= 0 \end{aligned}$$

- The Tanner graph of the code is a bipartite graph with n "bit-nodes" and $n-k$ "check-nodes", such that an edge (i, j) exists if $h_{i,j} = 1$, that is, if bit x_j participate in the parity-check equation i.

Hamming $(7, 4)$ code

- Parity-check equations of the Hamming $(7, 4)$ code:

$$\begin{aligned} x_1 + x_2 + x_4 + x_5 &= 0 \\ x_1 + x_3 + x_4 + x_6 &= 0 \\ x_2 + x_3 + x_4 + x_7 &= 0 \end{aligned}$$



LDPC codes

- Low-Density Parity-Check (LDPC) codes are linear binary codes with the characteristic that their parity-check matrix is *sparse*: the number of "ones" in the matrix is proportional to the block length n.

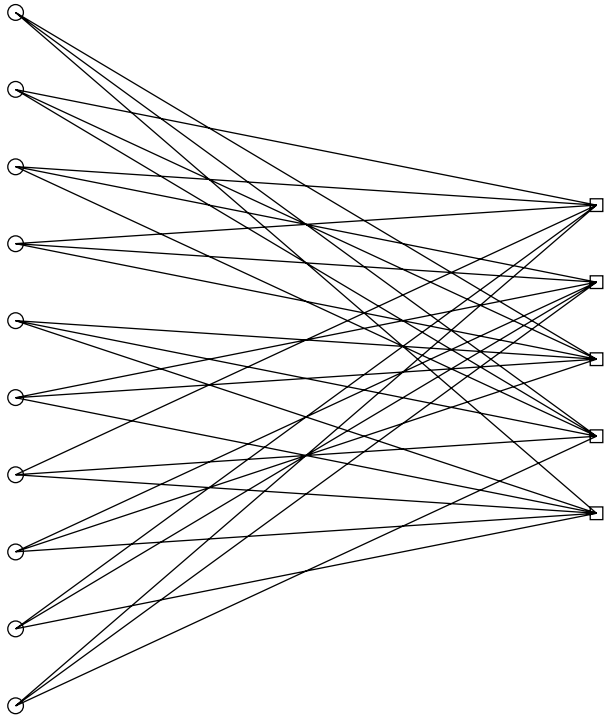
- Notice that a randomly generated binary matrix $\mathbf{H}$ with dimensions $n(1-R) \times n$ has an average number of ones equal to $n^2(1-R)/2$, i.e., quadratic with n.

- A regular (d_ℓ, d_r) LDPC code has Tanner graph with constant left and right degrees d_ℓ and d_r, respectively.

- Example: a $(3, 6)$ regular LDPC code of length $n = 10$, given by the parity-check matrix

$$\mathbf{H} = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

$(3, 6)$ LDPC code with $n = 10$



Introducing memory

- A binary linear block encoder is a linear transformation $\mathbb{F}_2^k \to \mathbb{F}_2^n$.
- What about using a Linear time-invariant linear system for encoding?
- Convolutional codes consider small k and n, but introduce memory into the encoding process of a sequence of consecutive blocks.
- We may see this a the convolution of a sequence of information blocks $\{\mathbf{u}_i\}$ with a matrix $\mathbf{G}$ of impulse responses, in order to generate a sequence of coded blocks $\{\mathbf{c}_i\}$.

"MA" $k \times n$ linear systems

- A $k \times n$ Moving Average (MA) system is defined by:

$$
\begin{aligned}
c_i^{(1)} &= \sum_{\ell=0}^{m_{1,1}} g_\ell^{(1,1)} u_{i-\ell}^{(1)} + \cdots + \sum_{\ell=0}^{m_{k,1}} g_\ell^{(k,1)} u_{i-\ell}^{(k)} \\
c_i^{(2)} &= \sum_{\ell=0}^{m_{1,2}} g_\ell^{(1,2)} u_{i-\ell}^{(1)} + \cdots + \sum_{\ell=0}^{m_{k,2}} g_\ell^{(k,2)} u_{i-\ell}^{(k)} \\
&\vdots \\
c_i^{(n)} &= \sum_{\ell=0}^{m_{1,n}} g_\ell^{(1,n)} u_{i-\ell}^{(1)} + \cdots + \sum_{\ell=0}^{m_{k,n}} g_\ell^{(k,n)} u_{i-\ell}^{(k)}
\end{aligned}
$$

- Defining a vector output sequence $\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2, \ldots$ such that $\mathbf{c}_i = (c_i^{(1)}, \ldots, c_i^{(n)})$ and a vector input sequence $\mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2, \ldots$ such that $\mathbf{u}_i = (u_i^{(1)}, \ldots, u_i^{(k)})$, we can write

$$\mathbf{c}_i = \sum_{\ell=0}^{m} \mathbf{u}_{i-\ell} \mathbf{G}_\ell$$

where we let $m = \max\{m_{(i,j)}\}$.

- We obtain a block-Toeplitz notation

$$(\mathbf{c}_0, \mathbf{c}_1, \mathbf{c}_2, \mathbf{c}_3, \ldots) = (\mathbf{u}_0, \mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3, \ldots) \begin{bmatrix} \mathbf{G}_0 & \mathbf{G}_1 & \cdots & \mathbf{G}_m & 0 & \cdots \\ 0 & \mathbf{G}_0 & & \mathbf{G}_{m-1} & \mathbf{G}_m & \\ \vdots & 0 & \ddots & \vdots & \mathbf{G}_{m-1} & \ddots \\ & & & \mathbf{G}_0 & \vdots & \\ & & & 0 & \mathbf{G}_0 & \ddots \end{bmatrix}$$

- The impulse responses $\mathbf{g}^{(i,j)}$ are called the code generators.

D-transform

- D-transform domain

$$\mathbf{u}_i \rightarrow \mathbf{u}(D) = \sum_i \mathbf{u}_i D^i \quad \text{(Laurent series)}$$

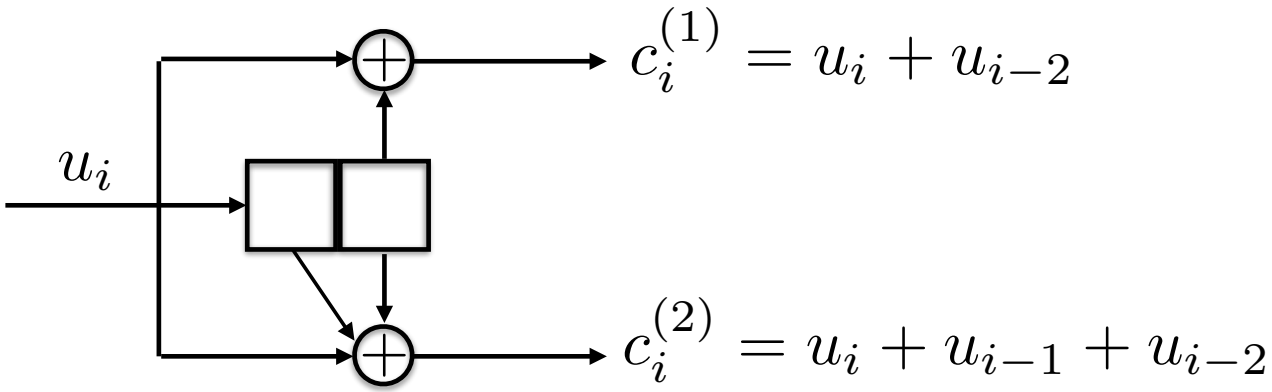
- Convolutional encoding in the D-transform domain:

$$\mathbf{c}(D) = \mathbf{u}(D)\mathbf{G}(D)$$

or, equivalently,

$$(c_1(D), \ldots, c_n(D)) = (u_1(D), \ldots, u_k(D)) \begin{bmatrix} g_{1,1}(D) & g_{1,2}(D) & \cdots & g_{1,n}(D) \\ g_{2,1}(D) & g_{2,2}(D) & \cdots & g_{2,n}(D) \\ \vdots & & & \vdots \\ g_{k,1}(D) & g_{k,2}(D) & \cdots & g_{k,n}(D) \end{bmatrix}$$

Example: a $(2, 1)$ convolutional code



Encoder canonical forms

- A code $\mathcal{C}$ is defined as the set of all output sequences (code sequences).
- As for block codes, a convolutional code $\mathcal{C}$ may have several input-ouput encoder implementations.
- We seek encoders in canonical form: a general problem in system theory is for a given system, defined as the ensemble of all its output sequences, what is the minimal canonical realization?
- State-space representation (ABCD):

$$\mathbf{s}_{i+1} = \mathbf{s}_i\mathbf{A} + \mathbf{u}_i\mathbf{B}, \quad \mathbf{c}_i = \mathbf{s}_i\mathbf{C} + \mathbf{u}_i\mathbf{D}$$

a minimal representation is a representation with the minimum number of state variables.

More on the $(2,1)$ example

- In the $(2,1)$ example of before, the state is defined as the content of the memory elements,

$$\mathbf{s}_i = (u_{i-1}, u_{i-2})$$

therefore we have

$$\mathbf{s}_{i+1} = \mathbf{s}_i \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} + u_i \begin{bmatrix} 1 & 0 \end{bmatrix}$$

and

$$\mathbf{c}_i = \mathbf{s}_i \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} + u_i \begin{bmatrix} 1 & 1 \end{bmatrix}$$

Generalization ...

- A convolutional code can be seen as a block code defined on the field $\mathbb{F}_q(D)$ of rational functions over $\mathbb{F}_q$.
- Roughly speaking: rational functions are to polynomials as rationals $\mathbb{Q}$ to the integers $\mathbb{Z}$.
- Generalizing what seen before, we can consider $\mathbf{G}(D)$ with rational elements $g_{i,j}(D)$.
- In system theory, this corresponds to AR-MA linear systems.
- The code is preserved by elementary row operations.
- It follows that for any $\mathbf{G}(D)$, we can find a systematic generator matrix in the form

$$\mathbf{G}(D) = [\mathbf{I}|\mathbf{P}(D)]$$

where $\mathbf{I}$ is the $k \times k$ identity, and $\mathbf{P}(D)$ is a $k \times (n-k)$ matrix of rational functions.

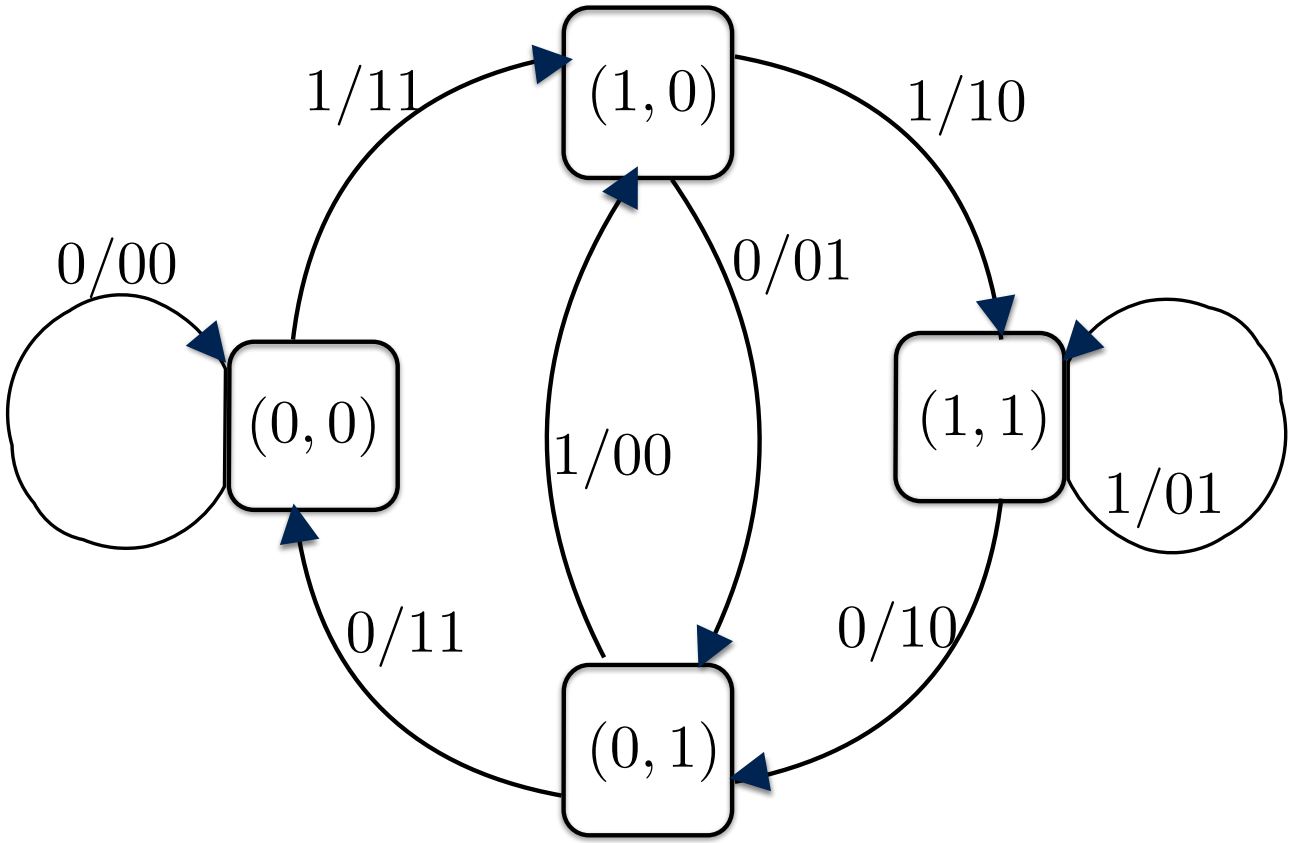
State diagram

- A state-space realization with m binary state variables is a finite-state machine (FSM) with a state space $\Sigma = \mathbb{F}_2^m$.

- In general, a FSM is described by its state transition diagram, i.e., by a graph with $|\Sigma|$ vertices, corresponding to all possible state configurations, and edges connecting those states for which a transition is possible.

- Each edge $(\mathbf{s}, \mathbf{s}') \in \Sigma \times \Sigma$ is labeled by input and output vectors $\mathbf{b} \in \mathbb{F}_2^k$ and $\mathbf{c} \in \mathbb{F}_2^n$, corresponding to the state transition between $\mathbf{s}$ and $\mathbf{s}'$.

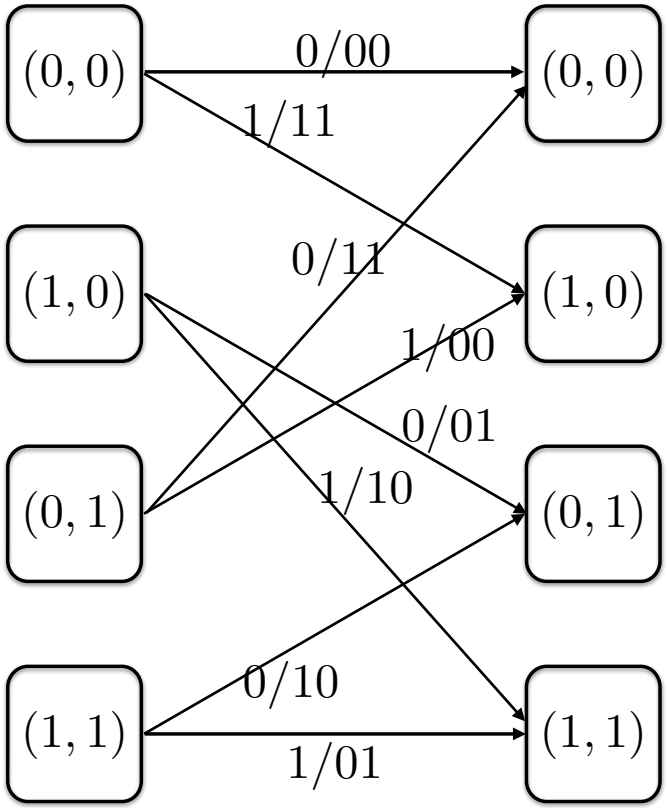
Example: the $(2, 1)$ 4-state code



Trellis diagram

- An alternative representation consists of the trellis section, i.e., by a bipartite graph with $|\Sigma|$ state vertices on the left and $|\Sigma|$ state vertices on the right.
- Left vertices represent the possible states at time i, and right vertices represent the possible states at time $i+1$. Edges represent the possible state transitions corresponding to input $\mathbf{u}_i$ and output $\mathbf{c}_i$.
- The trellis representation follows from the state transition diagram by introducing the time axis.
- A trellis diagram for a convolutional code consists of the concatenation of an infinite number of trellis sections.
- Given an initial state at time $i = 0$, an input sequence $\mathbf{u}(D)$ determines an output sequence $\mathbf{c}(D)$ and a state sequence $\mathbf{s}(D)$ that correspond to a path in the trellis.

Example: the $(2, 1)$ 4-state code

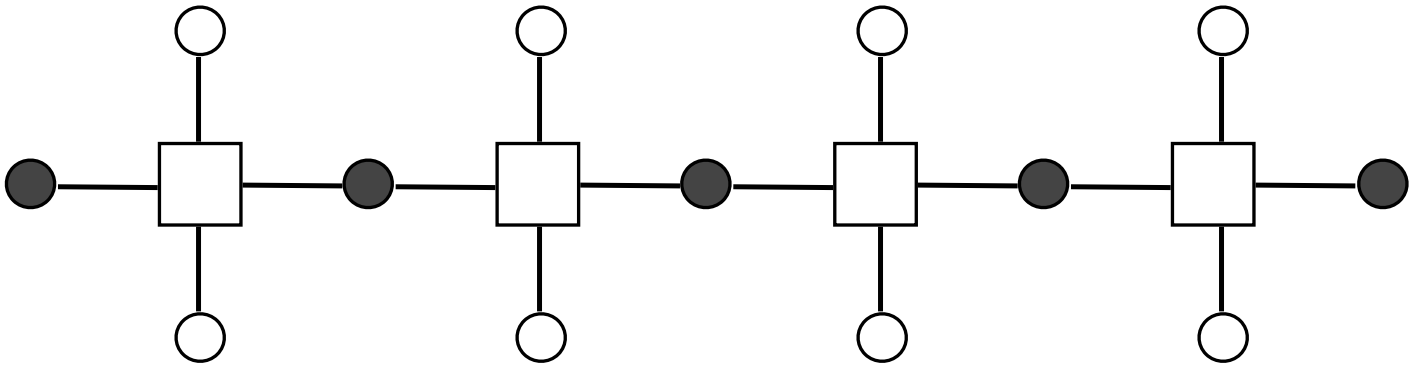


The Factor Graph for a convolutional code

- Three types of variable nodes: information bitnodes $\mathbf{u}_i$, coded bitnodes $\mathbf{c}_i$ and states nodes $\mathbf{s}_i$.

- The function nodes correspond to the state and output mappings

$$\mathbf{s}_{i+1} = \mathbf{s}_i\mathbf{A} + \mathbf{u}_i\mathbf{B}, \qquad \mathbf{c}_i = \mathbf{s}_i\mathbf{C} + \mathbf{u}_i\mathbf{D}$$



End of Lecture 4