Error-Correcting codes: Application of convolutional codes to Video Streaming

Diego Napp

Department of Mathematics, Universidad of Aveiro, Portugal

CIDMA

CENTRO DE I&D EM MATEMÁTICA E APLICAÇÕES

CENTER FOR R&D IN MATHEMATICS AND APPLICATIONS

July 22, 2016

Overview

Error Correcting Codes

Basic Problem:

- want to store bits on magnetic storage device
- or send a message (sequence of zeros/ones)

Error Correcting Codes

Basic Problem:

- want to store bits on magnetic storage device
- or send a message (sequence of zeros/ones)
- Bits get corrupted, $0 \to 1$ or $1 \to 0$, but rarely.

Error Correcting Codes

Basic Problem:

- want to store bits on magnetic storage device
- or send a message (sequence of zeros/ones)
- Bits get corrupted, $0 \to 1$ or $1 \to 0$, but rarely.

What happens when we store/send information and errors occur?

Error Correcting Codes

Basic Problem:

- want to store bits on magnetic storage device
- or send a message (sequence of zeros/ones)
- Bits get corrupted, $0 \to 1$ or $1 \to 0$, but rarely.

What happens when we store/send information and errors occur?

can we detect them? correct?

The International Standard Book Number (ISBN)

It can be proved that all possible valid ISBN-10's have at least two digits different from each other.
ISBN-10:

$$x_1 - x_2x_3x_4 - x_5x_6x_7x_8x_9 - x_{10}$$

satisfy

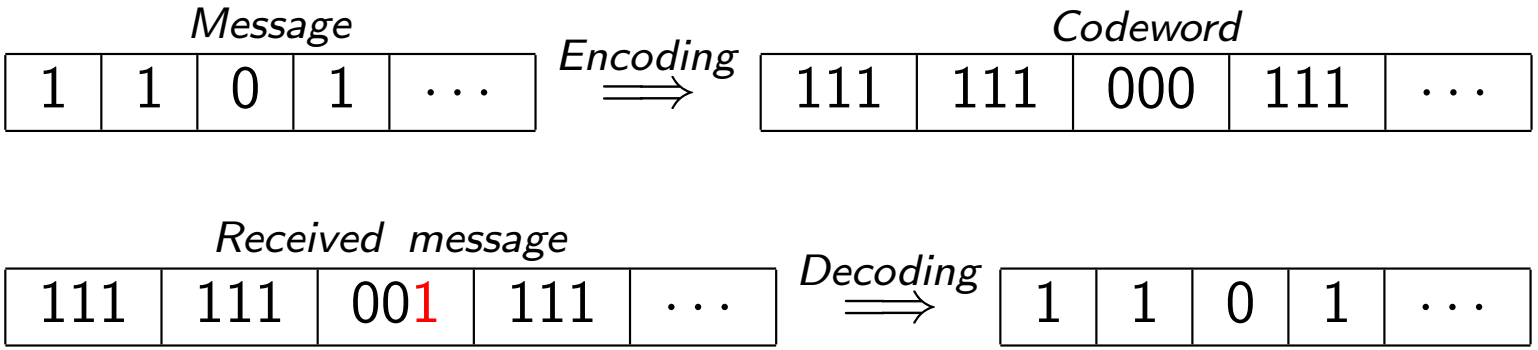
$$\sum_{i=1}^{10} ix_i = 0 \mod 11$$

For example, for an ISBN-10 of 0-306-40615-2:

$$\begin{aligned} s &= (0 \times 10) + (3 \times 9) + (0 \times 8) + (6 \times 7) + \\ &+ (4 \times 6) + (0 \times 5) + (6 \times 4) + (1 \times 3) + (5 \times 2) + (2 \times 1) \\ &= 0 + 27 + 0 + 42 + 24 + 0 + 24 + 3 + 10 + 2 \\ &= 132 = 12 \times 11 \end{aligned}$$

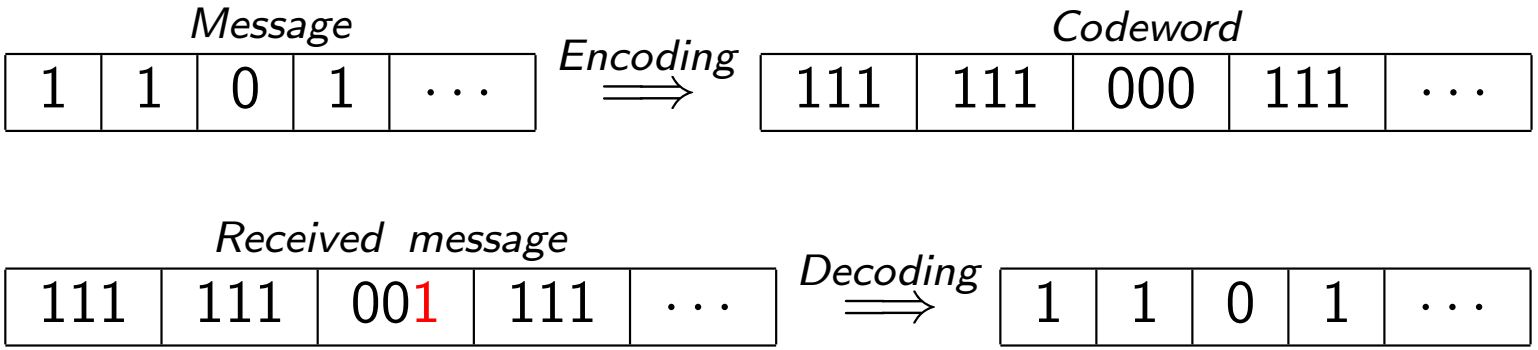
Sending/storing information: Naive solution

Repeat every bit three times



Sending/storing information: Naive solution

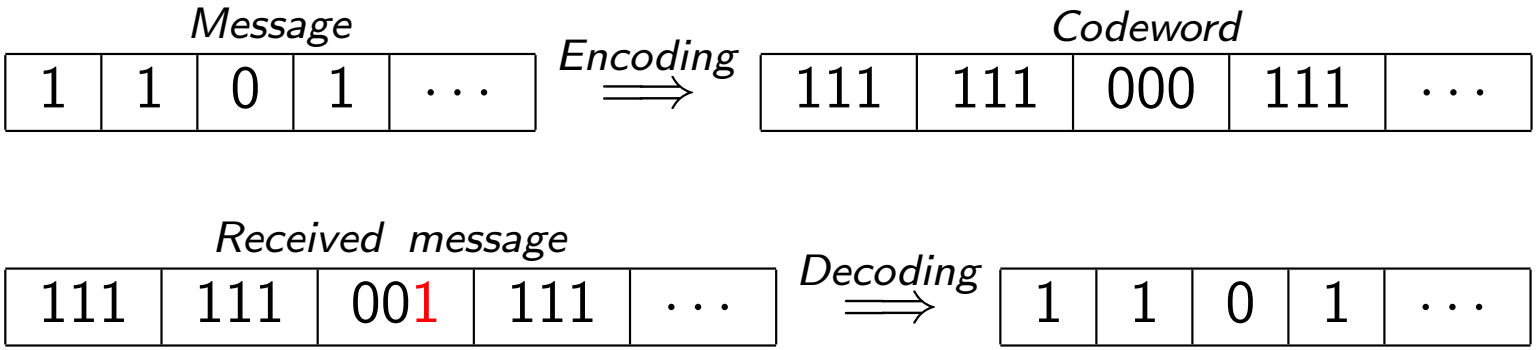
Repeat every bit three times



- Good: Very easy Encoding / decoding
- Bad: Rate 1/3

Sending/storing information: Naive solution

Repeat every bit three times



- Good: Very easy Encoding / decoding
- Bad: Rate 1/3

Can we do better???

Another Solution

- Break the message into 2 bits blocks $m = \boxed{u_1 \mid u_2} \in \mathbb{F}^2$
- Encode each block as follows:

$$u \longrightarrow uG$$

where

$$G = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix};$$

$$(u_1,\ u_2)\begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} = \boxed{u_1 \mid u_2 \mid u_1 + u_2}$$

Another Solution

- Break the message into 2 bits blocks $m = \boxed{u_1 \mid u_2} \in \mathbb{F}^2$
- Encode each block as follows:

$$u \longrightarrow uG$$

where

$$G = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix};$$

$$(u_1,\ u_2)\begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix} = \boxed{u_1 \mid u_2 \mid u_1 + u_2}$$

- Better Rate 2/3
- I can detect 1 error...but I cannot correct.

- Break the message into 3 bits blocks $m = \boxed{1}\boxed{1}\boxed{0} \in \mathbb{F}^3$
- Encode each block as follows:

$$u \longrightarrow uG \qquad G = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix};$$

- Break the message into 3 bits blocks $m = \boxed{1}\boxed{1}\boxed{0} \in \mathbb{F}^3$
- Encode each block as follows:

$$u \longrightarrow uG \qquad G = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix};$$

For example

$$(1,\ 1,\ 0)\begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} = (1,\ 1,\ 0,\ 0,\ 1,\ 1);$$

$$(1,\ 0,\ 1)\begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} = (1,\ 0,\ 1,\ 1,\ 0,\ 1);$$

etc...

- Rate $3/6 = 1/2$, better than before $(1/3)$.
- Only 2^3 codewords in $\mathbb{F}^6$

$$\mathcal{C} = \{(1,0,0,1,1,0),(0,1,0,1,0,1),(0,0,1,0,1,1),(1,1,0,0,1,1),$$

$$(1,0,1,1,0,1),(0,1,1,1,1,0),(1,1,1,0,0,0),(0,0,0,0,0,0)\}$$

- In $\mathbb{F}^6$ we have 2^6 possible vectors

- Rate $3/6 = 1/2$, better than before $(1/3)$.
- Only 2^3 codewords in $\mathbb{F}^6$

$$\mathcal{C} = \{(1,0,0,1,1,0),(0,1,0,1,0,1),(0,0,1,0,1,1),(1,1,0,0,1,1),$$

$$(1,0,1,1,0,1),(0,1,1,1,1,0),(1,1,1,0,0,0),(0,0,0,0,0,0)\}$$

- In $\mathbb{F}^6$ we have 2^6 possible vectors

- Any two codewords differ at least in 3 coordinates. I can detect and correct 1 error!!!

Hamming distance

The intuitive concept of "closeness" of two words is well formalized through Hamming distance $h(x, y)$ of words x, y. For two words x, y

$$h(x, y) = \text{the number of symbols } x \text{ and } y \text{ differ.}$$

A code $\mathcal{C}$ is a subset of $\mathbb{F}^n$, $\mathbb{F}$ a finite field. An important parameter of $\mathcal{C}$ is its minimal distance.

$$dist(\mathcal{C}) = \min\{h(x, y) \mid x, y \in \mathcal{C}, x \neq y\},$$

Hamming distance

The intuitive concept of "closeness" of two words is well formalized through Hamming distance $h(x, y)$ of words x, y. For two words x, y

$$h(x, y) = \text{the number of symbols } x \text{ and } y \text{ differ.}$$

A code $\mathcal{C}$ is a subset of $\mathbb{F}^n$, $\mathbb{F}$ a finite field. An important parameter of $\mathcal{C}$ is its minimal distance.

$$dist(\mathcal{C}) = \min\{h(x, y) \mid x, y \in \mathcal{C}, x \neq y\},$$

Theorem (Basic error correcting theorem)

1. *A code $\mathcal{C}$ can detected up to s errors if $dist(\mathcal{C}) \geq s + 1$.*
2. *A code $\mathcal{C}$ can correct up to t errors if $dist(\mathcal{C}) \geq 2t + 1$.*

Definition

An (n, k) block code $\mathcal{C}$ is a k-dimensional subspace of $\mathbb{F}^n$ and the rows of G form a basis of $\mathcal{C}$

$$\mathcal{C} \quad = \quad \mathrm{Im}_{\mathbb{F}} G = \left\{ uG : \; u \in \mathbb{F}^k \right\} \tag{1}$$

Definition

An (n, k) block code $\mathcal{C}$ is a k-dimensional subspace of $\mathbb{F}^n$ and the rows of G form a basis of $\mathcal{C}$

$$\mathcal{C} \quad = \quad \mathrm{Im}_{\mathbb{F}} G = \left\{ uG : \ u \in \mathbb{F}^k \right\} \tag{1}$$

Main coding theory problem

1. Construct codes that can correct a maximal number of errors while using a minimal amount of redundancy (rate)
2. Construct codes (as above) with efficient encoding and decoding procedures

- Coding theory develops methods to protect information against errors.
- Cryptography develops methods how to protect information against an enemy (or an unauthorized user).
- Coding theory - theory of error correcting codes - is one of the most interesting and applied part of mathematics and informatics.
- All real systems that work with digitally represented data, as CD players, TV, fax machines, internet, satelites, mobiles, require to use error correcting codes because all real channels are, to some extent, noisy.
- Coding theory methods are often elegant applications of very basic concepts and methods of (abstract) algebra.

Topics we are investigating

- (Convolutional) Codes over finite rings ($\mathbb{Z}_{p^r}$).

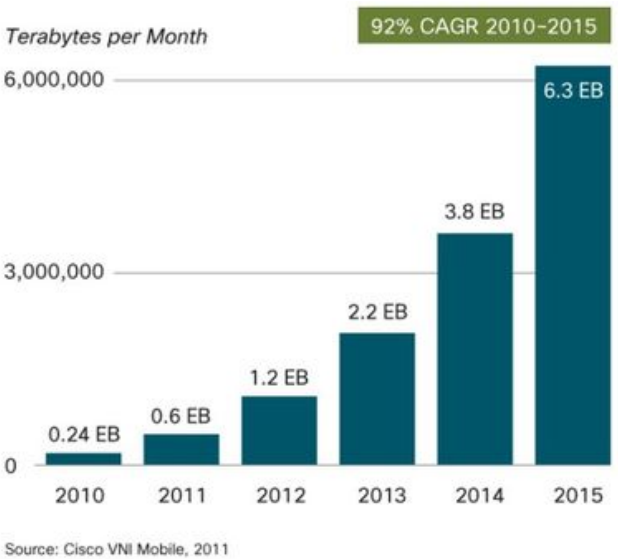
Topics we are investigating

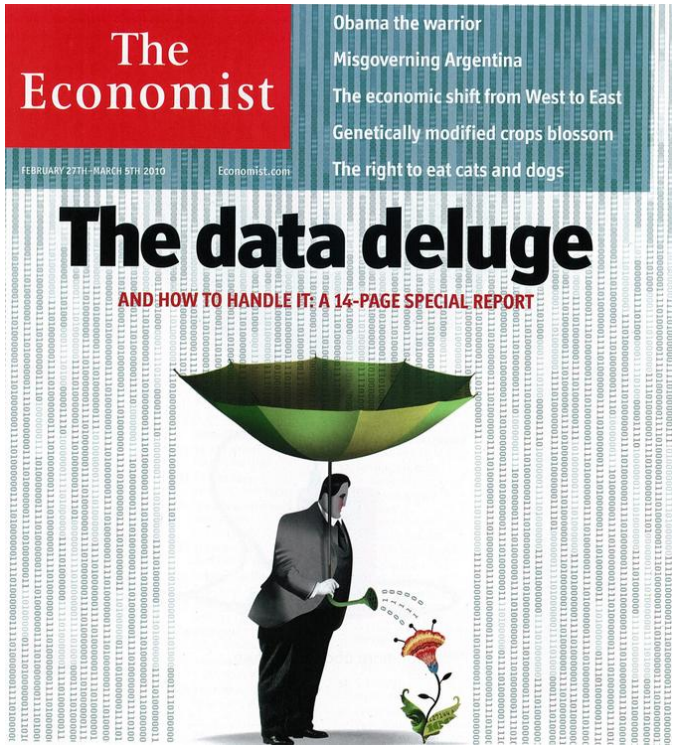
- (Convolutional) Codes over finite rings ($\mathbb{Z}_{p^r}$).
- Application of convolutional codes to Distributed Storage Systems.

Topics we are investigating

- (Convolutional) Codes over finite rings ($\mathbb{Z}_{p^r}$).
- Application of convolutional codes to Distributed Storage Systems.
- Application of convolutional codes Network Coding, in particular to Video Streaming.

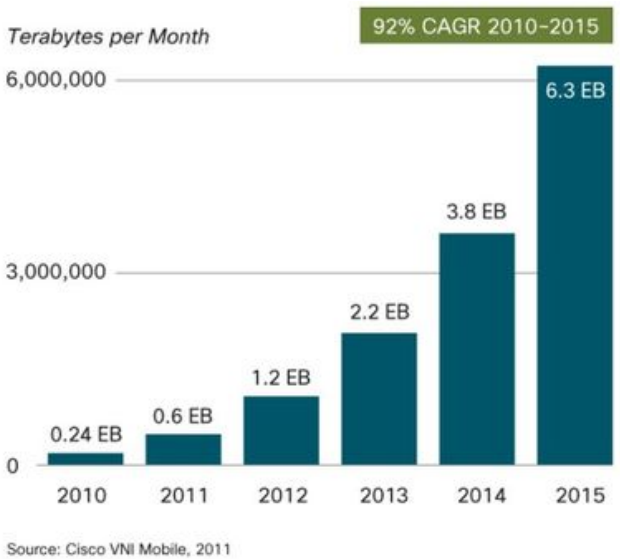
Distributed storage systems

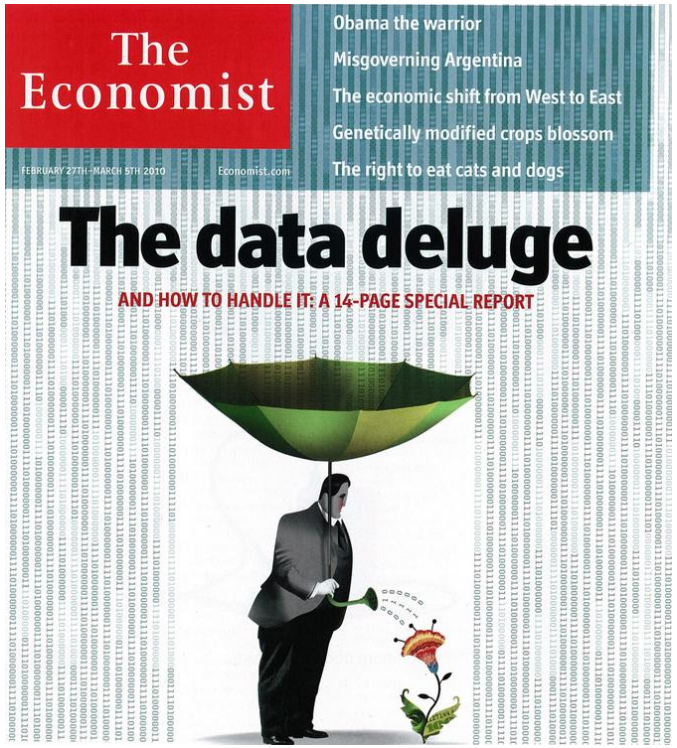




- Fast-growing demand for large-scale data storage.

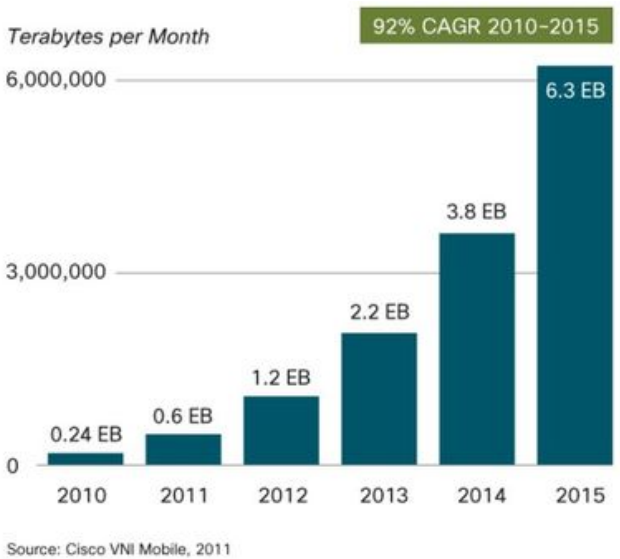
Distributed storage systems

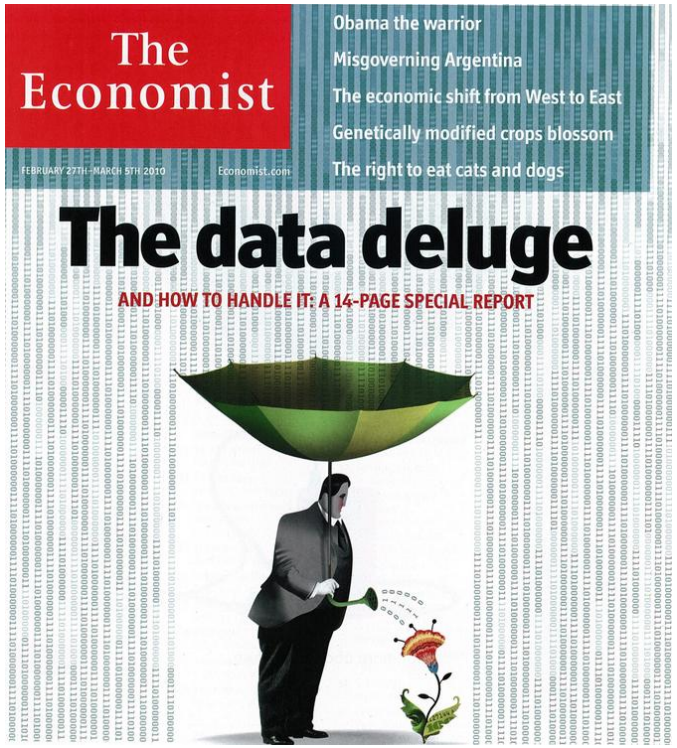




- Fast-growing demand for large-scale data storage.
- Failures occur: Redundancy is needed to ensure resilience $\Rightarrow$ Coding theory.

Distributed storage systems





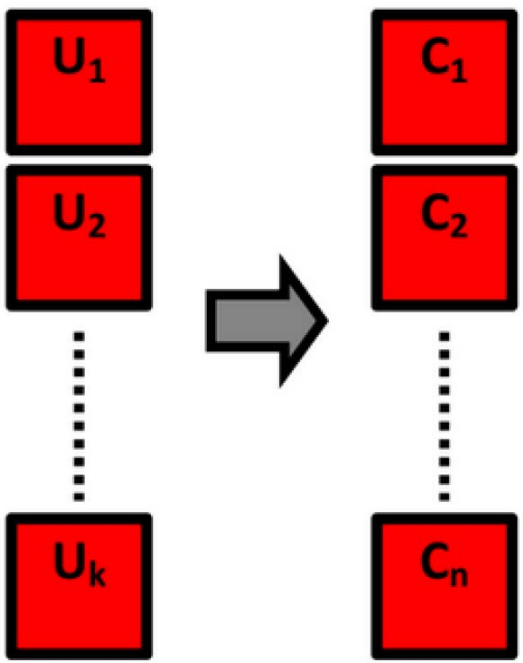
- Fast-growing demand for large-scale data storage.
- Failures occur: Redundancy is needed to ensure resilience $\Rightarrow$ Coding theory.
- Data is stored over a network of nodes: Peer-to-peer or data centers $\Rightarrow$ Distributed storage systems.

Coding Theory

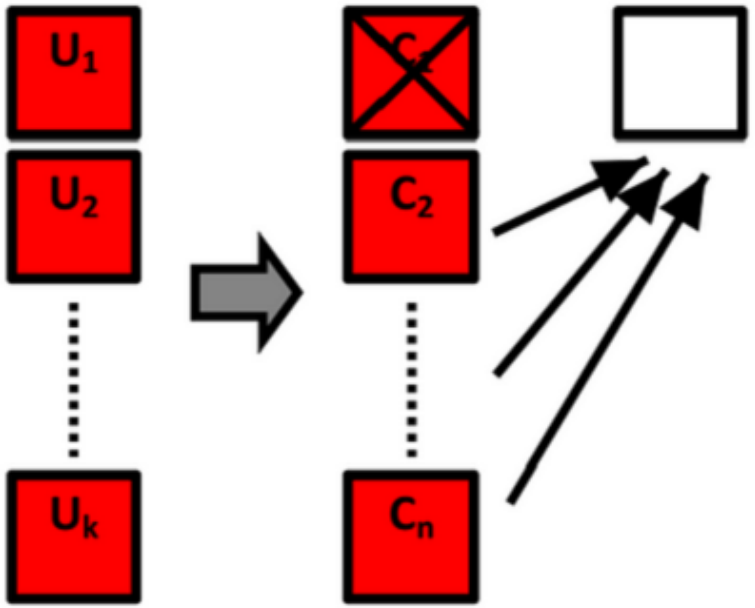
- A file $u = (u_1, \ldots, u_k) \in \mathbb{F}_q^k$ is redundantly stored across n nodes

$$v = (v_1, \ldots, v_n) = uG,$$

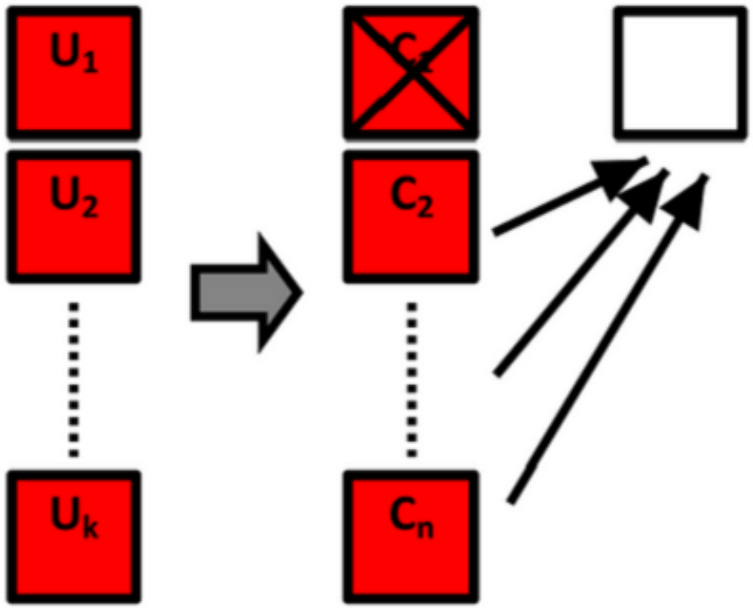
where G is the generator matrix of an (n, k)-code $\mathcal{C}$.



- What happens when nodes fail? The repair problem.



- What happens when nodes fail? The repair problem.



- Metrics:
  - Repair bandwidth
  - Storage cost
  - Locality

Locality

- The locality is the number of nodes necessary to repair one node that fails.

Definition

An (n, k) code has *locality* r if every codeword symbol in a codeword is a linear combination of at most r other symbols in the codeword.

Locality

- The locality is the number of nodes necessary to repair one node that fails.

Definition

An (n, k) code has *locality* r if every codeword symbol in a codeword is a linear combination of at most r other symbols in the codeword.

- Locality ≥ 2 (if the code is not a replication).
- There is a natural trade-off between distance and locality.

Locality

- The locality is the number of nodes necessary to repair one node that fails.

Definition

An (n, k) code has *locality* r if every codeword symbol in a codeword is a linear combination of at most r other symbols in the codeword.

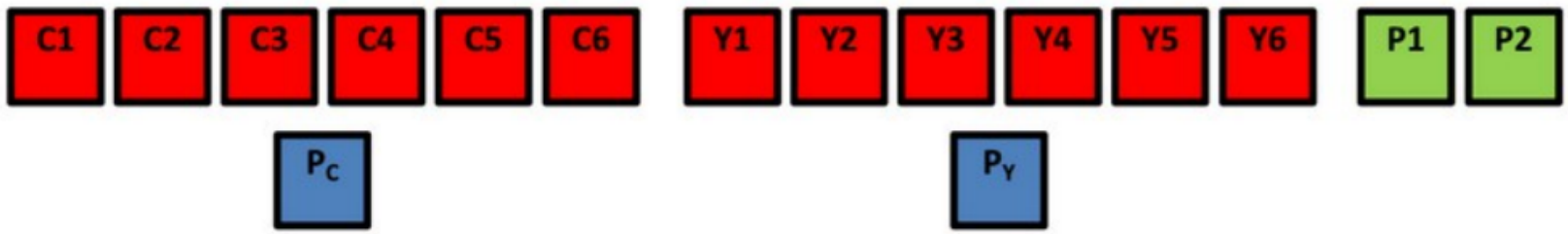
- Locality ≥ 2 (if the code is not a replication).

Theorem (Gopalan et. al,2012)

Let C be an (n, k) linear code with minimum distance d and locality r. Then

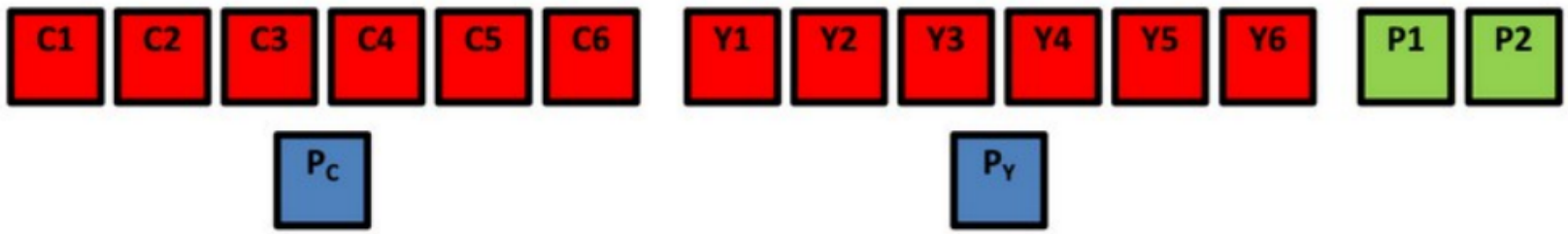
$$n - k + 1 - d \geq \left\lfloor \frac{k-1}{r} \right\rfloor.$$

Pyramid Codes are optimal with respect to this bound



- Pyramid codes were Implemented in Facebook and Windows Azure Storage (release in 2007).
- But if two erasures occur .... how to repair multiple erasures?

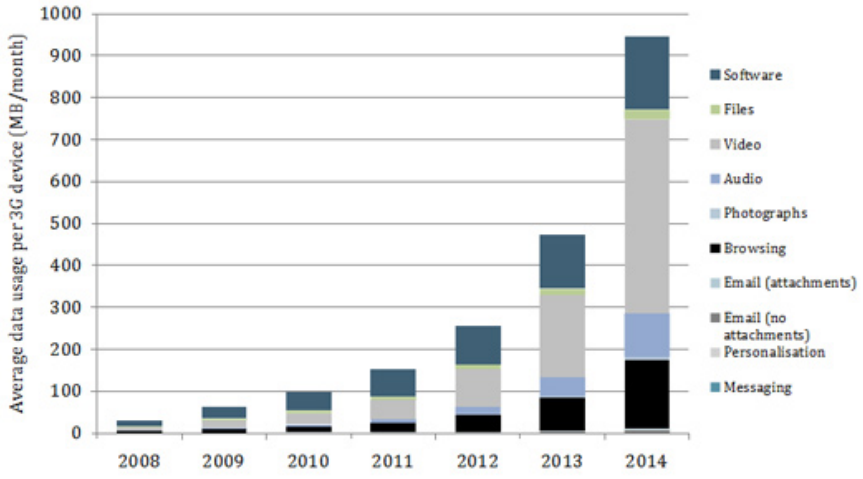
Pyramid Codes are optimal with respect to this bound



- Pyramid codes were Implemented in Facebook and Windows Azure Storage (release in 2007).
- But if two erasures occur .... how to repair multiple erasures?

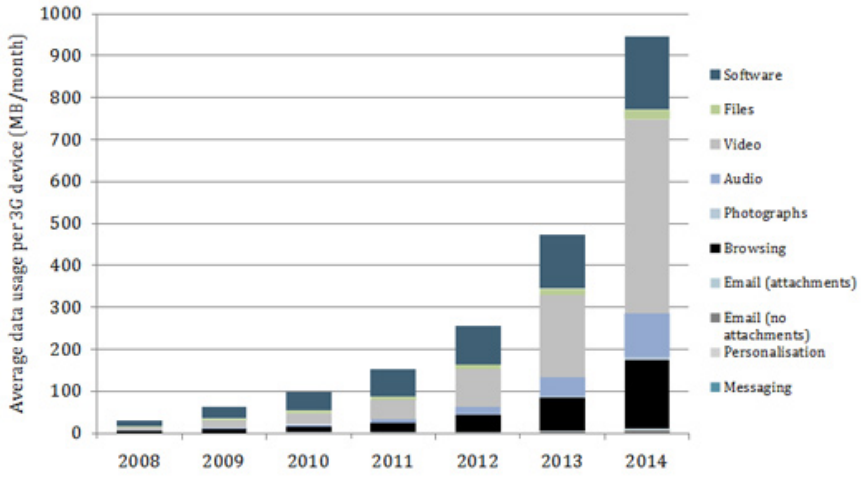
- ....Next time... Today we focus in Video Streaming.

Video Streaming



- Explosive growth of multimedia traffic in general and in video in particular.

Video Streaming

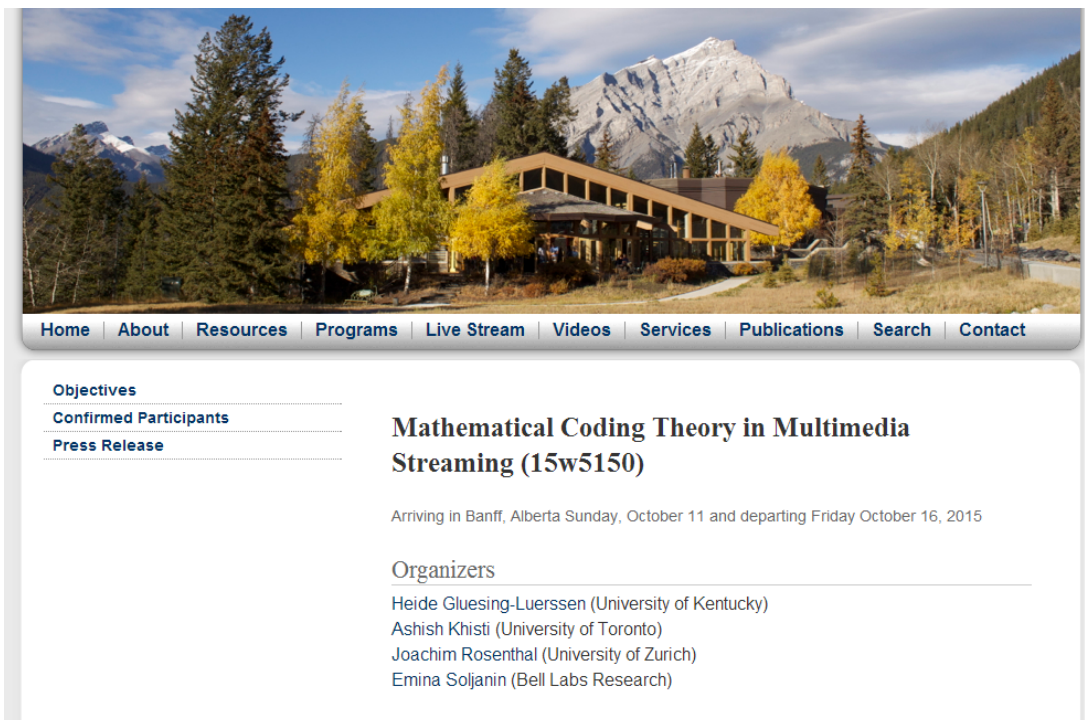


- Explosive growth of multimedia traffic in general and in video in particular.
- Video already accounts for over 50 % of the internet traffic today and mobile video traffic is expected to grow by a factor of more than 20 in the next five years [1].

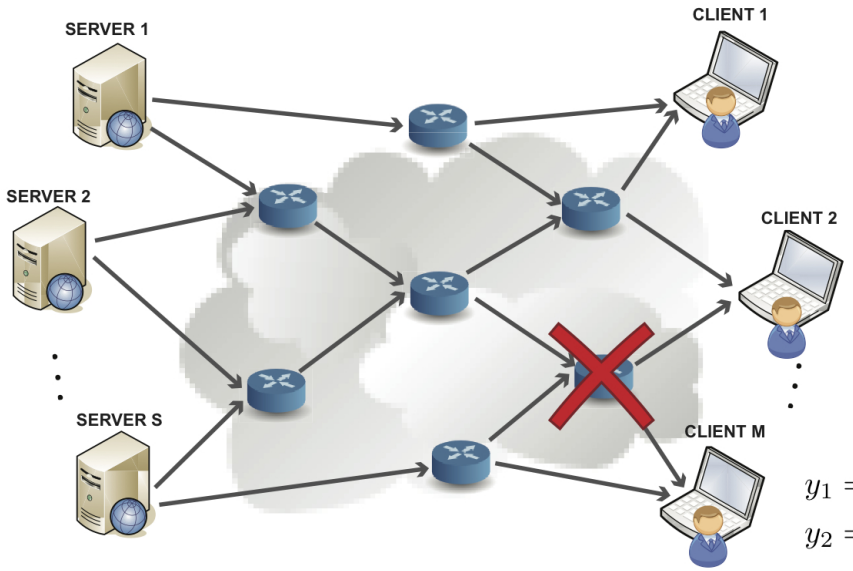
[1] Cisco: Forecast and Methodology 2012-2017.

Video Streaming

- Strong demand for implementing highly efficient approaches for video transmission.



Network Coding



$$y_1 = f_1(x_1, x_2, x_3)$$
$$y_2 = f_2(x_1, x_2, x_3)$$
$$y_3 = f_3(x_1, x_2, x_3)$$

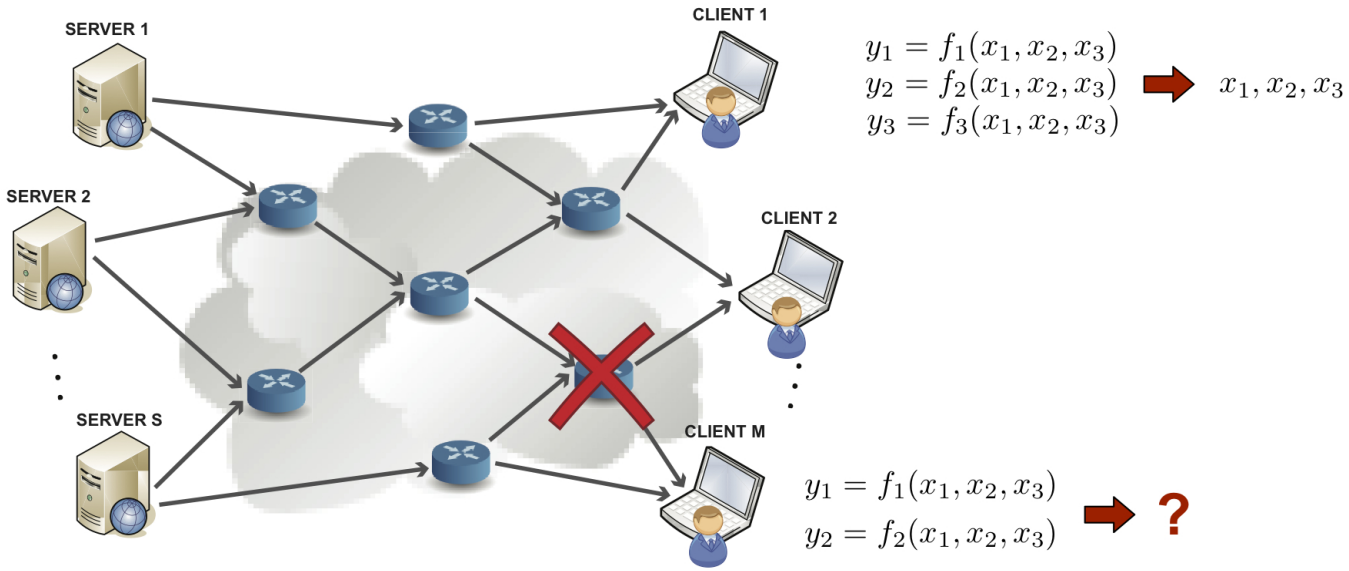
$\Rightarrow x_1, x_2, x_3$

$$y_1 = f_1(x_1, x_2, x_3)$$
$$y_2 = f_2(x_1, x_2, x_3)$$

$\Rightarrow$?

How is the best way to disseminate information over a network?

Network Coding

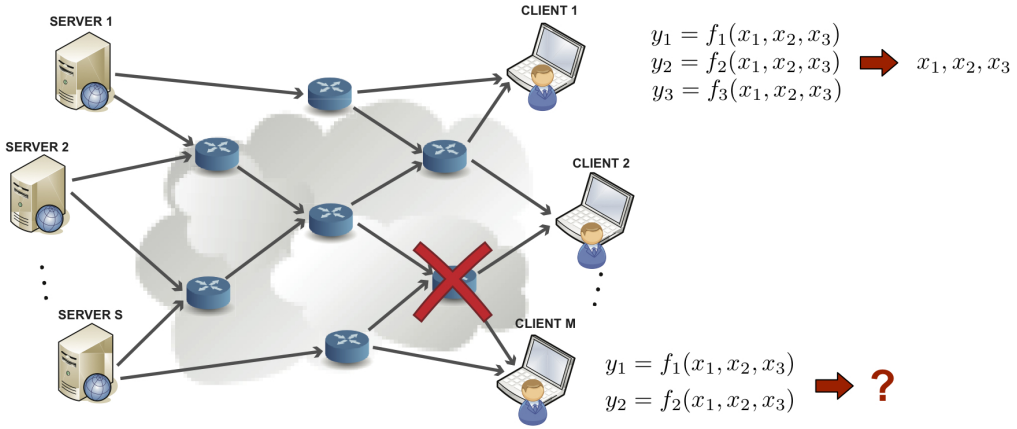


How is the best way to disseminate information over a network?

Linear random network coding

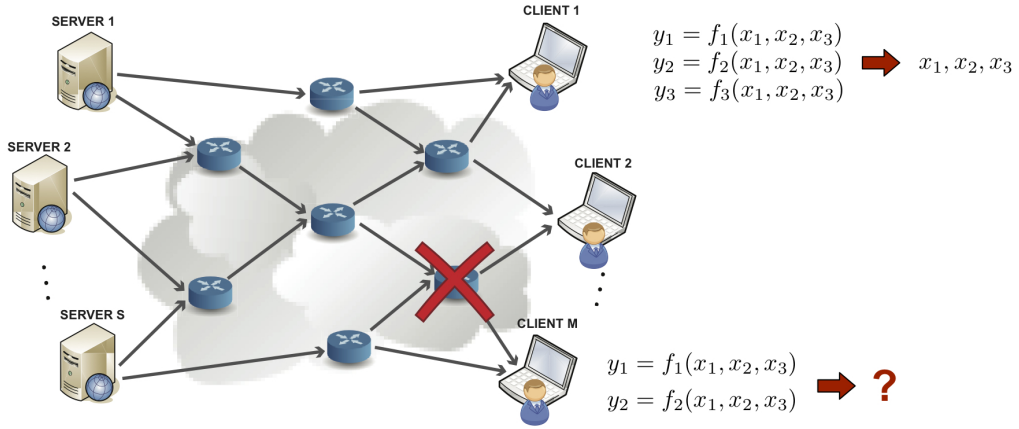
It has been proven that linear coding is enough to achieve the upper bound in multicast problems with one or more sources. It optimizes the throughput.

Linear Network Coding



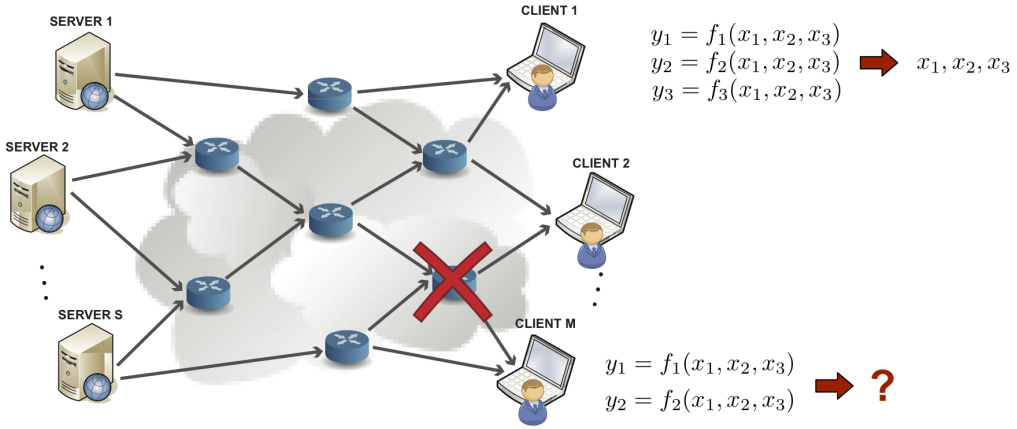
- During one *shot* the transmitter injects a number of packets into the network, each of which may be regarded as a row vector over a finite field $\mathbb{F}_{q^m}$.

Linear Network Coding



- During one *shot* the transmitter injects a number of packets into the network, each of which may be regarded as a row vector over a finite field $\mathbb{F}_{q^m}$.
- These packets propagate through the network. Each node creates a random -linear combination of the packets it has available and transmits this random combination.

Linear Network Coding



- During one *shot* the transmitter injects a number of packets into the network, each of which may be regarded as a row vector over a finite field $\mathbb{F}_{q^m}$.
- These packets propagate through the network. Each node creates a random -linear combination of the packets it has available and transmits this random combination.
- Finally, the receiver collects such randomly generated packets and tries to infer the set of packets injected into the network

Rank metric codes are used in Network Coding

- Rank metric codes are matrix codes $\mathcal{C} \subset \mathbb{F}_q^{m \times n}$, armed with the rank distance

$$d_{\text{rank}}(X, Y) = rank(X - Y), \text{ where } X, Y \in \mathbb{F}_q^{m \times n}.$$

Rank metric codes are used in Network Coding

- Rank metric codes are matrix codes $\mathcal{C} \subset \mathbb{F}_q^{m \times n}$, armed with the rank distance

$$d_{\text{rank}}(X, Y) = rank(X - Y), \text{ where } X, Y \in \mathbb{F}_q^{m \times n}.$$

- For linear (n, k) rank metric codes over $\mathbb{F}_{q^m}$ with $m \geq n$ the following analog of the Singleton bound holds,

$$d_{\text{rank}}(\mathcal{C}) \leq n - k + 1.$$

Rank metric codes are used in Network Coding

- Rank metric codes are matrix codes $\mathcal{C} \subset \mathbb{F}_q^{m \times n}$, armed with the rank distance

$$d_{\text{rank}}(X, Y) = rank(X - Y), \text{ where } X, Y \in \mathbb{F}_q^{m \times n}.$$

- For linear (n, k) rank metric codes over $\mathbb{F}_{q^m}$ with $m \geq n$ the following analog of the Singleton bound holds,

$$d_{\text{rank}}(\mathcal{C}) \leq n - k + 1.$$

- The code that achieves this bound is called Maximum Rank Distance (MRD). Gabidulin codes are a well-known class of MRD codes.

THE IDEA: Multi-shot

- Coding can also be performed over multiple uses of the network, whose internal structure may change at each shot

THE IDEA: Multi-shot

- Coding can also be performed over multiple uses of the network, whose internal structure may change at each shot
- Creating dependencies among the transmitted codewords of different shots can improve the error-correction capabilities

THE IDEA: Multi-shot

- Coding can also be performed over multiple uses of the network, whose internal structure may change at each shot
- Creating dependencies among the transmitted codewords of different shots can improve the error-correction capabilities
- Ideal coding techniques for video streaming must operate under low-latency, sequential encoding and decoding constrains, and as such they must inherently have a convolutional structure.

THE IDEA: Multi-shot

- Coding can also be performed over multiple uses of the network, whose internal structure may change at each shot
- Creating dependencies among the transmitted codewords of different shots can improve the error-correction capabilities
- Ideal coding techniques for video streaming must operate under low-latency, sequential encoding and decoding constrains, and as such they must inherently have a convolutional structure.

- Although the use of convolutional codes is widespread, its application to video streaming (or using the rank metric) is yet unexplored.

THE IDEA: Multi-shot

- Coding can also be performed over multiple uses of the network, whose internal structure may change at each shot
- Creating dependencies among the transmitted codewords of different shots can improve the error-correction capabilities
- Ideal coding techniques for video streaming must operate under low-latency, sequential encoding and decoding constrains, and as such they must inherently have a convolutional structure.

- Although the use of convolutional codes is widespread, its application to video streaming (or using the rank metric) is yet unexplored.
- We propose a novel scheme that add complex dependencies to data streams in a quite simple way

Block codes vs convolutional codes

$$\dots u_2,\ u_1,\ u_0 \xrightarrow{G} \dots v_2 = u_2 G,\ v_1 = u_1 G,\ v_0 = u_0 G$$

represented in a polynomial fashion

$$\cdots + u_2 D^2 + u_1 D + u_0 \xrightarrow{G} \cdots + \underbrace{u_2 G}_{v_2} D^2 + \underbrace{u_1 G}_{v_1} D + \underbrace{u_0 G}_{v_0}$$

substitute G by $G(D) = G_0 + G_1 D + \cdots + G_s D^s$?

$$\dots u_2 D^2 + u_1 D + u_0 \xrightarrow{G(D)} \dots \underbrace{(u_2 G_0 + u_1 G_1 + u_0 G_2)}_{v_2} D^2 + \underbrace{(u_1 G_0 + u_0 G_1)}_{v_1} D + \underbrace{u_0 G_0}_{v_0}$$

Block codes vs convolutional codes

$$\dots u_2,\ u_1,\ u_0 \xrightarrow{G} \dots v_2 = u_2 G,\ v_1 = u_1 G,\ v_0 = u_0 G$$

represented in a polynomial fashion

$$\cdots + u_2 D^2 + u_1 D + u_0 \xrightarrow{G} \cdots + \underbrace{u_2 G}_{v_2} D^2 + \underbrace{u_1 G}_{v_1} D + \underbrace{u_0 G}_{v_0}$$

substitute G by $G(D) = G_0 + G_1 D + \cdots + G_s D^s$?

$$\dots u_2 D^2 + u_1 D + u_0 \xrightarrow{G(D)} \dots \underbrace{(u_2 G_0 + u_1 G_1 + u_0 G_2)}_{v_2} D^2 + \underbrace{(u_1 G_0 + u_0 G_1)}_{v_1} D + \underbrace{u_0 G_0}_{v_0}$$

Block codes: $\mathcal{C} = \{uG\} = \mathrm{Im}_{\mathbb{F}} G \sim \{u(D)G\} = \mathrm{Im}_{\mathbb{F}} G(D)$
Convolutional codes: $\mathcal{C} = \{u(D)G(D)\} = \mathrm{Im}_{\mathbb{F}((D))} G(D)$

Definition

A convolutional code $\mathcal{C}$ is a $\mathbb{F}((D))$-subspace of $\mathbb{F}^n((D))$.

Definition

A convolutional code $\mathcal{C}$ is a $\mathbb{F}((D))$-subspace of $\mathbb{F}^n((D))$.

A matrix $G(D)$ whose rows form a bases for $\mathcal{C}$ is called an encoder.
If $\mathcal{C}$ has rank k then we say the $\mathcal{C}$ has rate k/n.

$$\begin{aligned} \mathcal{C} &= \mathrm{Im}_{\mathbb{F}((D))} G(D) = \left\{ u(D)G(D) : \ u(D) \in \mathbb{F}^k((D)) \right\} \\ &= \mathrm{Ker}_{\mathbb{F}[D]} H(D) = \{ v(D) \in \mathbb{F}^n[D] : \ v(D)H(D) = 0 \} \end{aligned}$$

where $H(D)$ is called the parity-check of $\mathcal{C}$.

Definition

A convolutional code $\mathcal{C}$ is a $\mathbb{F}((D))$-subspace of $\mathbb{F}^n((D))$.

A matrix $G(D)$ whose rows form a bases for $\mathcal{C}$ is called an encoder.
If $\mathcal{C}$ has rank k then we say the $\mathcal{C}$ has rate k/n.

$$\begin{aligned} \mathcal{C} &= \mathrm{Im}_{\mathbb{F}((D))} G(D) = \left\{ u(D)G(D) : \ u(D) \in \mathbb{F}^k((D)) \right\} \\ &= \mathrm{Ker}_{\mathbb{F}[D]} H(D) = \{ v(D) \in \mathbb{F}^n[D] : \ v(D)H(D) = 0 \} \end{aligned}$$

where $H(D)$ is called the parity-check of $\mathcal{C}$.

Remark

One can also consider the ring of polynomials $\mathbb{F}[D]$ instead of Laurent series $\mathbb{F}((D))$ and define $\mathcal{C}$ as a $\mathbb{F}[D]$-module of $\mathbb{F}^n[D]$.

A convolutional encoder is *also* a linear device which maps

$$u(0), u(1), \cdots \longrightarrow v(0), v(1), \ldots$$

In this sense it is the same as block encoders. The difference is that the convolutional encoder has an internal "storage vector" or "memory".

$v(i)$ does not depend only on $u(i)$ but also on the storage vector $x(i)$

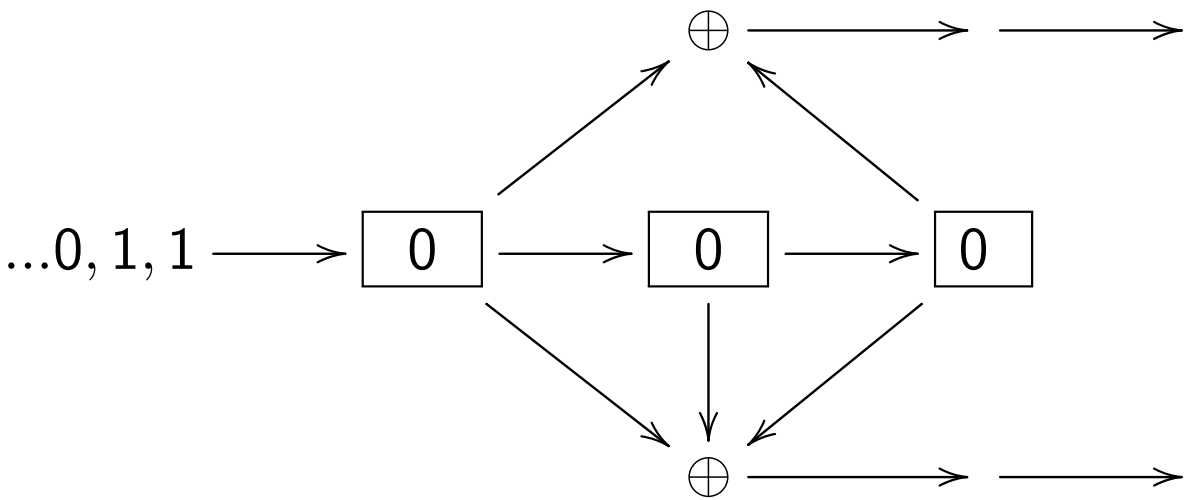
$$\begin{aligned} x(i+1) &= Ax(i) + Bu(i) \\ v(i) &= Cx(i) + Eu(i) \end{aligned} \tag{2}$$

$A \in \mathbb{F}^{\delta \times \delta}, B \in \mathbb{F}^{\delta \times k}, C \in \mathbb{F}^{n \times \delta}, E \in \mathbb{F}^{\delta \times k}$.

The encoder

$$G(D) = \begin{pmatrix} D^2 + 1 \\ D^2 + D + 1 \end{pmatrix}$$

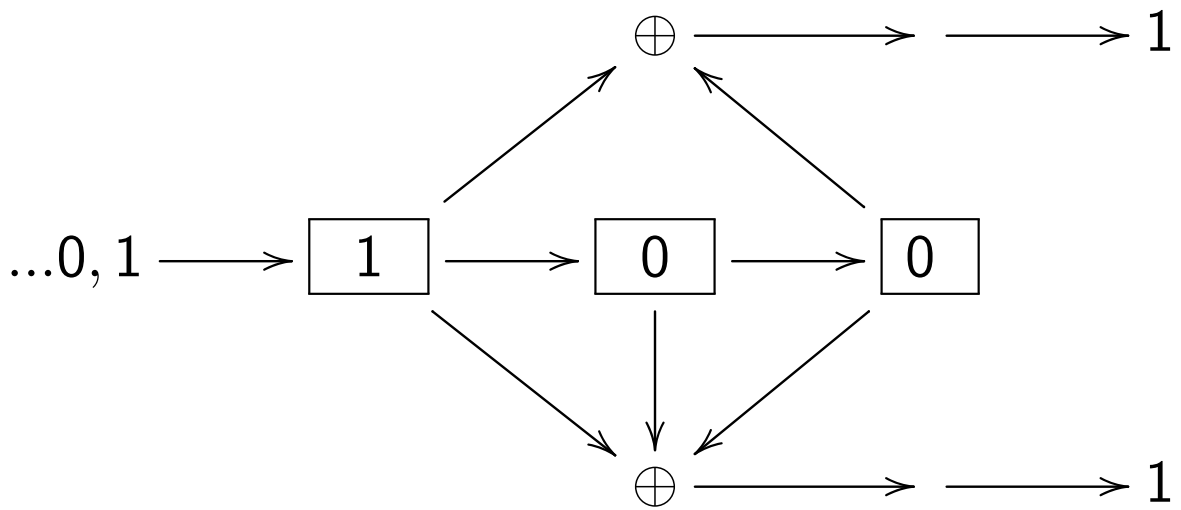
has the following implementation



The encoder

$$G(D) = \begin{pmatrix} D^2 + 1 \\ D^2 + D + 1 \end{pmatrix}$$

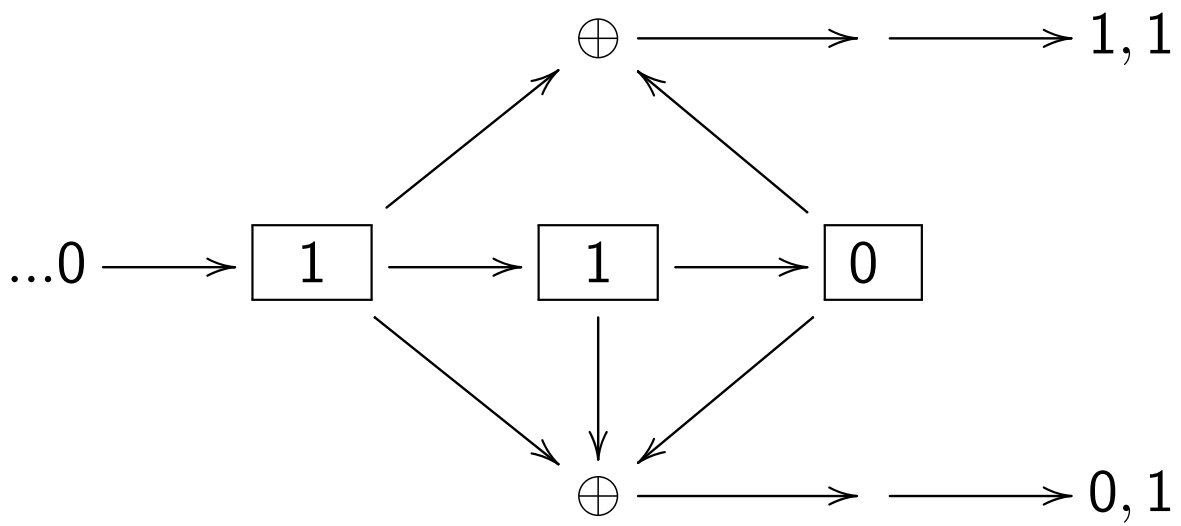
has the following implementation



The encoder

$$G(D) = \begin{pmatrix} D^2 + 1 \\ D^2 + D + 1 \end{pmatrix}$$

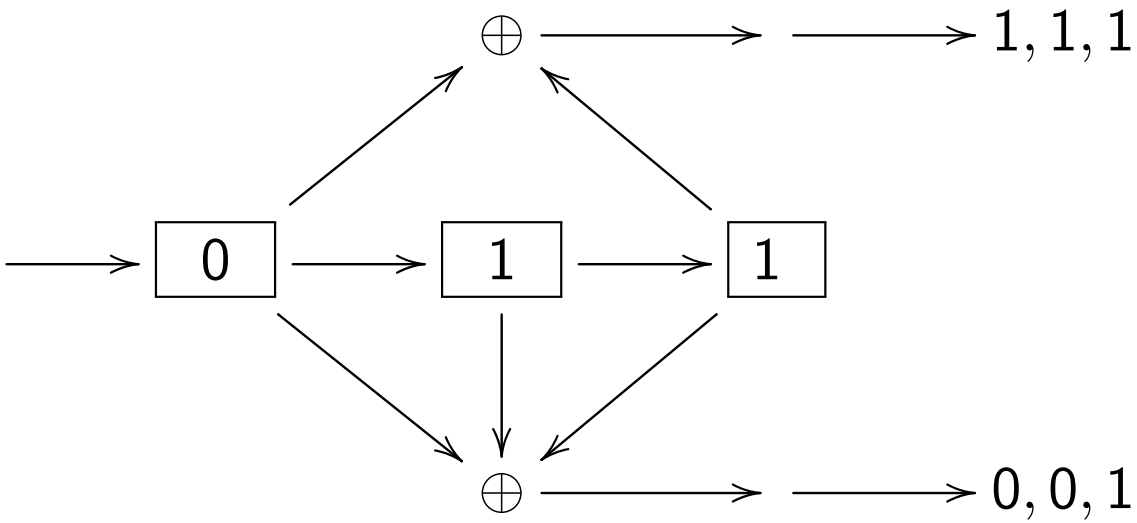
has the following implementation



The encoder

$$G(D) = \begin{pmatrix} D^2 + 1 \\ D^2 + D + 1 \end{pmatrix}$$

has the following implementation



Example

Let the convolutional code be given by matrices

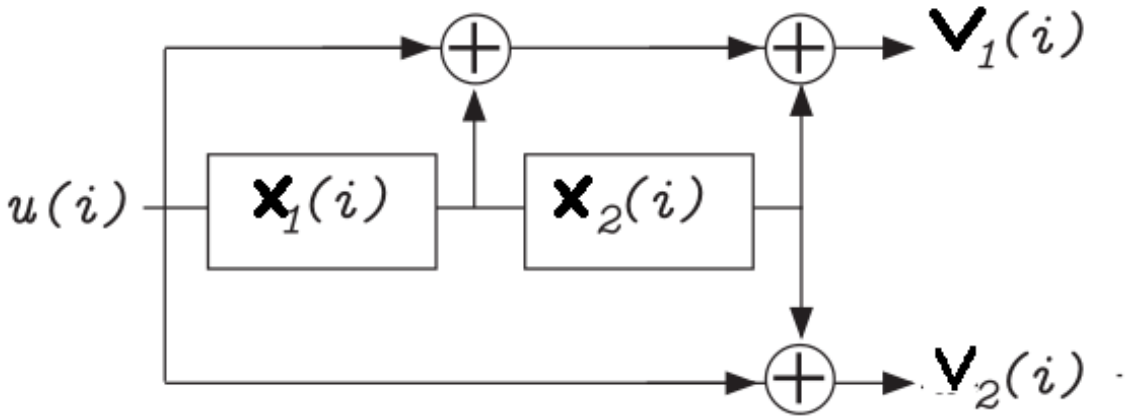
$$\begin{bmatrix} x_1(i+1) \\ x_2(i+1) \end{bmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{bmatrix} x_1(i) \\ x_2(i) \end{bmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix} u(i)$$

$$\begin{bmatrix} v_1(i) \\ v_2(i) \end{bmatrix} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{bmatrix} x_1(i) \\ x_2(i) \end{bmatrix} + \begin{pmatrix} 1 \\ 1 \end{pmatrix} u(i)$$

We can compute an encoder

$$G(D) = E + B(D^{-1}I_m - A)^{-1}C = \begin{pmatrix} 1 + D + D^2 & 1 + D^2 \end{pmatrix}$$

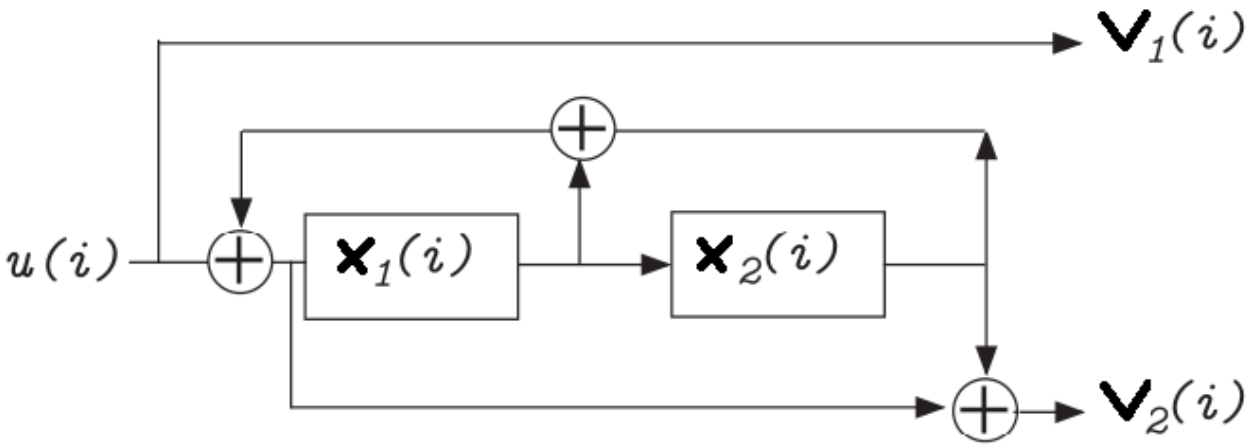
Example



A physical realization for the encoder
$G(D) = \begin{pmatrix} 1+D+D^2 & 1+D^2 \end{pmatrix}$. This encoder has degree 2 and memory 2.
Clearly any matrix which is $\mathbb{F}(D)$-equivalent to $G(D)$ is also an encoder.

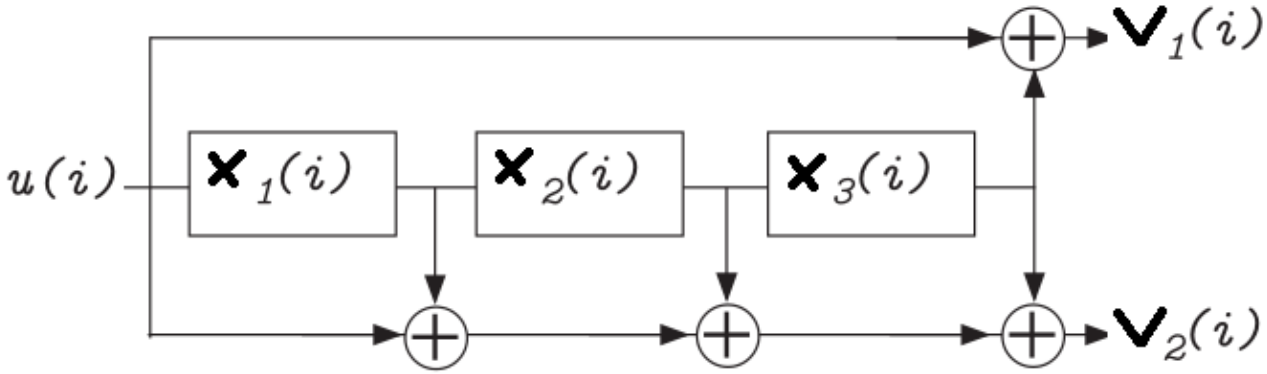
$$G'(D) = \begin{pmatrix} 1 & \frac{1+D^2}{1+D+D^2} \end{pmatrix}$$

Example



A physical realization for the generator matrix $G'(D)$. This encoder has degree 2 and infinite memory.

Example



A physical realization for the catastrophic encoder $G''(D) = \begin{pmatrix} 1+D^3 & 1+D+D^2+D^3 \end{pmatrix}$. This encoder has degree 3 and memory 3.

Polynomial encoders

Two encoders $G(D), G'(D)$ generate the same code if there exist an invertible matrix $U(D)$ such that $G(D) = U(D)G'(D)$.

Polynomial encoders

Two encoders $G(D), G'(D)$ generate the same code if there exist an invertible matrix $U(D)$ such that $G(D) = U(D)G'(D)$.

Definition

A generator matrix $G(D)$ is said to be catastrophic if for every $v(D) = u(D)G(D)$

$$supp(v(D)) \text{ is finite } \Rightarrow supp(u(D)) \text{ is finite}$$

Polynomial encoders

Two encoders $G(D), G'(D)$ generate the same code if there exist an invertible matrix $U(D)$ such that $G(D) = U(D)G'(D)$.

Definition

A generator matrix $G(D)$ is said to be catastrophic if for every $v(D) = u(D)G(D)$

$$supp(v(D)) \text{ is finite } \Rightarrow supp(u(D)) \text{ is finite}$$

Theorem

$G(D)$ is non-catastrophic if $G(D)$ admits a polynomial right inverse.

Example

The encoder

$$G(D) = \begin{pmatrix} 1+D & 0 & 1 & D \\ 1 & D & 1+D & 0 \end{pmatrix}$$

is noncatastrophic as an inverse is

$$H(D) = \begin{pmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ D^{-1} & 1+D^{-1} \end{pmatrix}$$

Historical Remarks

- Convolutional codes were introduced by Elias (1955)
- The theory was imperfectly understood until a series of papers of Forney in the 70's on the algebra of the $k \times n$ matrices over the field of rational functions in the delay operator D.
- Became widespread in practice with the Viterbi decoding. They belong to the most widely implemented codes in (wireless) communications.
- The field is typically $\mathbb{F}_2$ but in the last decade a renewed interest has grown for convolutional codes over large fields trying to fully exploit the potential of convolutional codes.

In Applications

- In block coding it is normally considered n and k large.
- Convolutional codes are typically studied for n and k small and fixed ($n = 2$ and $k = 1$ is common) and for several values of δ.

In Applications

- In block coding it is normally considered n and k large.
- Convolutional codes are typically studied for n and k small and fixed ($n = 2$ and $k = 1$ is common) and for several values of δ.
- Decoding over the symmetric channel is, in general, difficult.

In Applications

- In block coding it is normally considered n and k large.
- Convolutional codes are typically studied for n and k small and fixed ($n = 2$ and $k = 1$ is common) and for several values of δ.
- Decoding over the symmetric channel is, in general, difficult.
- The field is typically $\mathbb{F}_2$. The degree cannot be too large so that the Viterbi decoding algorithm is efficient.

In Applications

- In block coding it is normally considered n and k large.
- Convolutional codes are typically studied for n and k small and fixed ($n = 2$ and $k = 1$ is common) and for several values of δ.
- Decoding over the symmetric channel is, in general, difficult.
- The field is typically $\mathbb{F}_2$. The degree cannot be too large so that the Viterbi decoding algorithm is efficient.
- Convolutional codes over large alphabets have attracted much attention in recent years.

In Applications

- In block coding it is normally considered n and k large.
- Convolutional codes are typically studied for n and k small and fixed ($n = 2$ and $k = 1$ is common) and for several values of δ.
- Decoding over the symmetric channel is, in general, difficult.
- The field is typically $\mathbb{F}_2$. The degree cannot be too large so that the Viterbi decoding algorithm is efficient.
- Convolutional codes over large alphabets have attracted much attention in recent years.
- In [Tomas, Rosenthal, Smarandache 2012]: Decoding over the erasure channel is *easy* and Viterbi is not needed, just linear algebra.

MDS convolutional codes over $\mathbb{F}$

The Hamming weight of a polynomial vector

$$v(D) = \sum_{i \in \mathbb{N}} v_i D^i = v_0 + v_1 D + v_2 D^2 + \cdots + v_\nu D^\nu \in \mathbb{F}[D]^n,$$

defined as $\mathrm{wt}(v(D)) = \sum_{i=0}^{\nu} \mathrm{wt}(v_i)$.

MDS convolutional codes over $\mathbb{F}$

The Hamming weight of a polynomial vector

$$v(D) = \sum_{i \in \mathbb{N}} v_i D^i = v_0 + v_1 D + v_2 D^2 + \cdots + v_\nu D^\nu \in \mathbb{F}[D]^n,$$

defined as $\mathrm{wt}(v(D)) = \sum_{i=0}^{\nu} \mathrm{wt}(v_i)$.

The free distance of a convolutional code $\mathcal{C}$ is given by,

$$d_{\mathrm{free}}(\mathcal{C}) \quad = \quad \min \{\mathrm{wt}(v(D)) \quad | \quad v(D) \in \mathcal{C} \quad \text{and} \quad v(D) \neq 0\}$$

MDS convolutional codes over $\mathbb{F}$

The Hamming weight of a polynomial vector

$$v(D) = \sum_{i \in \mathbb{N}} v_i D^i = v_0 + v_1 D + v_2 D^2 + \cdots + v_\nu D^\nu \in \mathbb{F}[D]^n,$$

defined as $\mathrm{wt}(v(D)) = \sum_{i=0}^{\nu} \mathrm{wt}(v_i)$.

The free distance of a convolutional code $\mathcal{C}$ is given by,

$$d_{\mathrm{free}}(\mathcal{C}) \quad = \quad \min \{\mathrm{wt}(v(D)) \ \mid \ v(D) \in \mathcal{C} \quad \text{and} \quad v(D) \neq 0\}$$

- We are interested in the maximum possible value of $d_{\mathrm{free}}(\mathcal{C})$

MDS convolutional codes over $\mathbb{F}$

The Hamming weight of a polynomial vector

$$v(D) = \sum_{i \in \mathbb{N}} v_i D^i = v_0 + v_1 D + v_2 D^2 + \cdots + v_\nu D^\nu \in \mathbb{F}[D]^n,$$

defined as $\mathrm{wt}(v(D)) = \sum_{i=0}^{\nu} \mathrm{wt}(v_i)$.

The free distance of a convolutional code $\mathcal{C}$ is given by,

$$d_{\mathrm{free}}(\mathcal{C}) \quad = \quad \min \{\mathrm{wt}(v(D)) \ \mid \ v(D) \in \mathcal{C} \quad \text{and} \quad v(D) \neq 0\}$$

- We are interested in the maximum possible value of $d_{\mathrm{free}}(\mathcal{C})$
- For block codes ($\delta = 0$) we know that maximum value is given by the Singleton bound: $n - k + 1$

MDS convolutional codes over $\mathbb{F}$

The Hamming weight of a polynomial vector

$$v(D) = \sum_{i \in \mathbb{N}} v_i D^i = v_0 + v_1 D + v_2 D^2 + \cdots + v_\nu D^\nu \in \mathbb{F}[D]^n,$$

defined as $\mathrm{wt}(v(D)) = \sum_{i=0}^{\nu} \mathrm{wt}(v_i)$.

The free distance of a convolutional code $\mathcal{C}$ is given by,

$$d_{\mathrm{free}}(\mathcal{C}) \quad = \quad \min \{\mathrm{wt}(v(D)) \quad | \quad v(D) \in \mathcal{C} \quad \text{and} \quad v(D) \neq 0\}$$

- We are interested in the maximum possible value of $d_{\mathrm{free}}(\mathcal{C})$
- For block codes ($\delta = 0$) we know that maximum value is given by the Singleton bound: $n - k + 1$
- This bound can be achieve if $|\mathbb{F}| > n$

Theorem

Rosenthal and Smarandache (1999) showed that the free distance of convolutional code of rate k/n and degree δ must be upper bounded by

$$d_{\text{free}}(\mathcal{C}) \leq (n-k)\left(\left\lfloor \frac{\delta}{k} \right\rfloor + 1\right) + \delta + 1. \tag{3}$$

A code achieving (3) is called Maximum Distance Separable (MDS) and if it achieves it "as fast as possible" is called strongly MDS.

Theorem

Rosenthal and Smarandache (1999) showed that the free distance of convolutional code of rate k/n and degree δ must be upper bounded by

$$d_{\text{free}}(\mathcal{C}) \leq (n-k)\left(\left\lfloor \frac{\delta}{k} \right\rfloor + 1\right) + \delta + 1. \tag{3}$$

A code achieving (3) is called Maximum Distance Separable (MDS) and if it achieves it "as fast as possible" is called strongly MDS.

- Allen conjecture (1999) the existence of convolutional codes that are both sMDS and MDP when $k = 1$ and $n = 2$.
- Rosenthal and Smarandache (2001), provided the first concrete construction of MDS convolutional codes
- Gluessing-Luerssen, et. al. (2006), provided the first construction of strongly MDS when $(n-k)|\delta$.
- Napp and Smarandache (2016) provided the first construction of strongly MDS for all rates and degrees.

Definition

Another important distance measure for a convolutional code is the jth column distance $d_j^c(\mathcal{C})$, (introduced by Costello), given by

$$d_j^c(\mathcal{C}) \quad = \quad \min\left\{\mathrm{wt}(v_{[0,j]}(D)) \;\mid\; v(D) \in \mathcal{C} \;\text{ and }\; v_0 \neq 0\right\}$$

where $v_{[0,j]}(D) = v_0 + v_1 D + \ldots + v_j D^j$ represents the j-th truncation of the codeword $v(D) \in \mathcal{C}$.

The column distances satisfy

$$d_0^c \leq d_1^c \leq \cdots \leq \lim_{j\to\infty} d_j^c(\mathcal{C}) = d_{\text{free}}(\mathcal{C}) \leq (n-k)\left(\left\lfloor \frac{\delta}{k} \right\rfloor + 1\right) + \delta + 1.$$

The j-th column distance is upper bounded as following

$$d_j^c(\mathcal{C}) \leq (n-k)(j+1) + 1, \tag{4}$$

The column distances satisfy

$$d_0^c \leq d_1^c \leq \cdots \leq \lim_{j\to\infty} d_j^c(\mathcal{C}) = d_{\text{free}}(\mathcal{C}) \leq (n-k)\left(\left\lfloor \frac{\delta}{k} \right\rfloor + 1\right) + \delta + 1.$$

The j-th column distance is upper bounded as following

$$d_j^c(\mathcal{C}) \leq (n-k)(j+1) + 1, \tag{4}$$

The column distances satisfy

$$d_0^c \leq d_1^c \leq \cdots \leq \lim_{j\to\infty} d_j^c(\mathcal{C}) = d_{\text{free}}(\mathcal{C}) \leq (n-k)\left(\left\lfloor \frac{\delta}{k} \right\rfloor + 1\right) + \delta + 1.$$

The j-th column distance is upper bounded as following

$$d_j^c(\mathcal{C}) \leq (n-k)(j+1) + 1, \tag{4}$$

How do we construct MDP?

The construction of MDP convolutional boils down to the construction of Superregular matrices

LT-Superregular matrices

Definition [Gluesing-Luerssen,Rosenthal,Smadandache (2006)]

A lower triangular matrix

$$B = \begin{pmatrix} a_0 & & & \\ a_1 & a_0 & & \\ \vdots & \vdots & \ddots & \\ a_j & a_{j-1} & \cdots & a_0 \end{pmatrix} \tag{5}$$

is *LT-superregular* if all of its minors, with no zeros in the diagonal, are nonsingular.

Remark

Note that due to such a lower triangular configuration the remaining minors are necessarily zero.

Example

$$\beta^3+\beta+1=0 \Rightarrow \begin{pmatrix} 1 & & & & \\ \beta & 1 & & & \\ \beta^3 & \beta & 1 & & \\ \beta & \beta^3 & \beta & 1 & \\ 1 & \beta & \beta^3 & \beta & 1 \end{pmatrix} \in \mathbb{F}_{2^3}^{5\times 5} \text{ is}$$

LT-superregular

Example

$$\epsilon^5+\epsilon^2+1=0 \Rightarrow \begin{pmatrix} 1 & & & & & & \\ \epsilon & 1 & & & & & \\ \epsilon^6 & \epsilon & 1 & & & & \\ \epsilon^9 & \epsilon^6 & \epsilon & 1 & & & \\ \epsilon^6 & \epsilon^9 & \epsilon^6 & \epsilon & 1 & & \\ \epsilon & \epsilon^6 & \epsilon^9 & \epsilon^6 & \epsilon & 1 & \\ 1 & \epsilon & \epsilon^6 & \epsilon^9 & \epsilon^6 & \epsilon & 1 \end{pmatrix} \in \mathbb{F}_{2^5}^{7\times 7} \text{ is}$$

LT-superregular

Remarks

- Construction of classes of LT-superregular matrices is very difficult due to their triangular configuration.
- Only two classes exist:

1. Rosenthal et al. (2006) presented the first construction. For any n there exists a prime number p such that

$$\begin{pmatrix} \binom{n}{0} & & & \\ \binom{n-1}{1} & \binom{n}{0} & & \\ \vdots & \ddots & \ddots & \\ \binom{n-1}{n-1} & \cdots & \binom{n-1}{1} & \binom{n}{0} \end{pmatrix} \in \mathbb{F}_p^{n \times n}$$

is LT-superregular. Bad news: Requires a field with very large characteristic.

Remarks

2. Almeida, Napp and Pinto (2013) first construction over any characteristic: Let α be a primitive element of a finite field $\mathbb{F}$ of characteristic p. If $|\mathbb{F}| \geq p^{2^M}$ then the following matrix

$$\begin{bmatrix} \alpha^{2^0} & & & & \\ \alpha^{2^1} & \alpha^{2^0} & & & \\ \alpha^{2^2} & \alpha^{2^1} & \alpha^{2^0} & & \\ \vdots & & \ddots & \ddots & \\ \alpha^{2^{M-1}} & \cdots & & \cdots & \alpha^{2^0} \end{bmatrix}.$$

is LT-superregular. Bad news: $|\mathbb{F}|$ very large.

Performance over the erasure channel

Theorem

Let $\mathcal{C}$ be an (n, k, δ) convolutional code and $d^c_{j_0}$ the $j = j_0$ -th column distance. If in any sliding window of length $(j_0 + 1)n$ at most $d^c_{j_0} - 1$ erasures occur then we can recover completely the transmitted sequence.

Performance over the erasure channel

Theorem

Let $\mathcal{C}$ be an (n, k, δ) convolutional code and $d^c_{j_0}$ the $j = j_0$ -th column distance. If in any sliding window of length $(j_0 + 1)n$ at most $d^c_{j_0} - 1$ erasures occur then we can recover completely the transmitted sequence.

Example

$$\cdots vv\big|\overbrace{\star\star\cdots\star\star}^{60}\overbrace{vvv\cdots vv}^{80}\overbrace{\star\star\cdot\star\star}^{60}vv\big|vv\cdots$$

A $[202, 101]$ MDS blok code can correct 101 erasures in a window of 202 symbols (recovering rate $\frac{101}{202}$): $\Rightarrow$ cannot correct this window.

A $(2, 1, 50)$ MDP convolutional code has also 50% error capability. $(L + 1)n = 101 \times 2 = 202$. Take a window of 120 symbols, correct and continue until you correct the whole window.

We have flexibility in choosing the size and position of the sliding window.

Fundamental Open Problems

- Come up with LT superregular matrices over small fields.
- How is the minimum field size needed to construct LT superregular matrices?
- Typically convolutional codes are decoded via the Viterbi decoding algorithm. The complexity of this algorithm grows exponentially with the McMillan degree. New classes of codes coming with more efficient decoding algorithms are needed.
- Good constructions of convolutional codes for rank metric
- Good constructions tailor made to deal with *burst of erasures* (in both Hamming and Rank metric)

References

G.D. Forney, Jr. (1975)
"Minimal bases of rational vector spaces, with applications to multivariable linear systems",
SIAM J. Control 13, 493–520, 1975.

G.D. Forney, Jr. (1970)
"Convolutional codes I: algebraic structure ",
IEEE Trans. Inform. Theory vol. IT-16, pp. 720-738, 1970.

R.J. McEliece (1998)
The Algebraic Theory of Convolutional Codes
Handbook of Coding Theory Vol. 1 , North-Holland, Amsterdam.

Johannesson, Rolf and Zigangirov, K. (1998)
Fundamentals of convolutional coding
IEEE Communications society and IEEE Information theory society and Vehicular Technology Society.

Summary of the basics

- Convolutional codes are block codes with memory. Convolutional codes generalize linear block codes in a natural way.
- Convolutional codes are capable of decoding a large number of errors per time interval require a large free distance and a good distance profile.
- In order to construct good convolutional codes we need to construct superregular matrices over $\mathbb{F}$: Difficult.
- Convolutional codes treat the information as a stream: Great potential for video streaming
- A lots of Mathematics are needed: Linear algebra, systems theory, finite fields, rings/module theory, etc...

Muchas gracias por la invitacion!