Tutorial on

Streaming Codes

P. Vijay Kumar*, Nikhil Krishnan†, Myna Vajha*, Vinayak Ramkumar*

* Electrical Communication Engineering, Indian Institute of Science, Bengaluru
† International Institute of Information Technology, Bengaluru

National Conference on Communications (NCC 2021)

Indian Institute of Technology Kanpur (Virtual)
July 27

Thanks go out to Prof Adrish Banerjee as well as other members of the organizing committee of NCC 2021:

1. General Co-Chairs: Rajesh Hegde & Sudeb Dasgupta
2. TPC Co-Chairs: Ketan Rajawat, Debashis Ghosh & Rohit Budhiraja
3. Tutorial Committee: R K Bansal, Amalendu Patnaik, Srikrishna Bhashyam

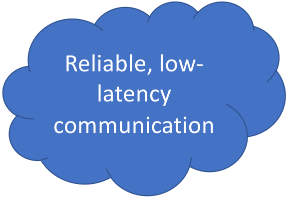
for the invitation...

Motivation - from NCC 2014, IIT Kanpur!

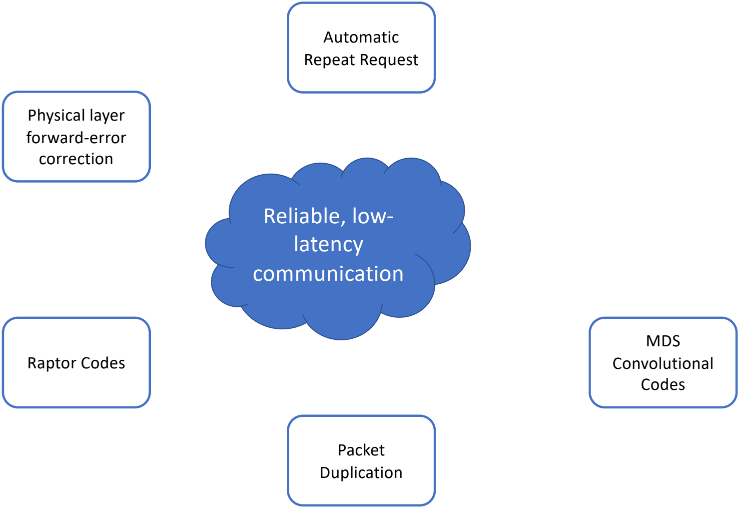
Gerhard Fettweis, "Designing the Physical Layer of 5G for Enabling the Tactile Internet"

- What sets 5G apart from prior generations is the emphasis on latency
- Low-latency communication will revolutionize education, mobility, healthcare, sports, entertainment, gaming,....

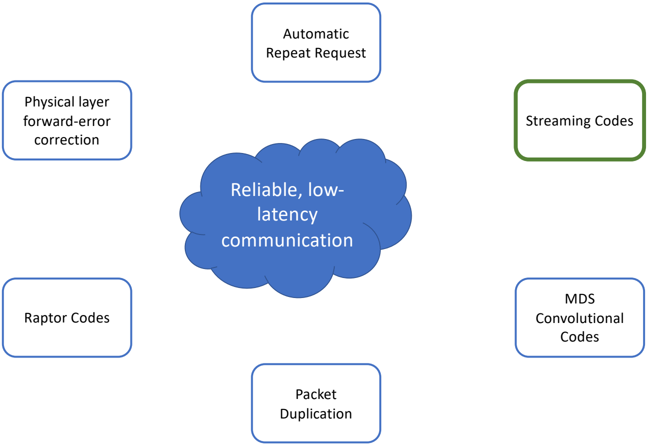
The General Goal



Some Potential Coding Strategies



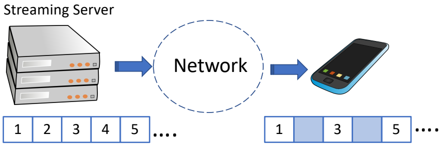
Streaming Codes - An Added Option



ToC

Setting Addressed by Streaming Codes

- Sequence of information-bearing packets $s_0, s_1, \ldots$, sent over an erasure channel
- Packet drops (erasures), shown below in in blue, due to
  - network congestion,
  - a degraded wireless link, or
  - a packet that arrives too late.



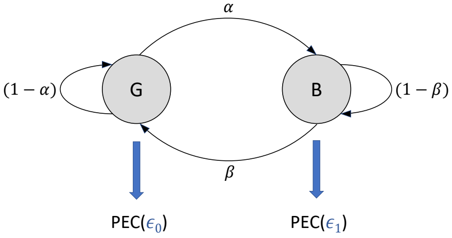
(blue rectangles denote erased packets)

Goal: use packet-level FEC to ensure best-possible tradeoff between rate and reliability

- under decoding-delay constraint of τ packets

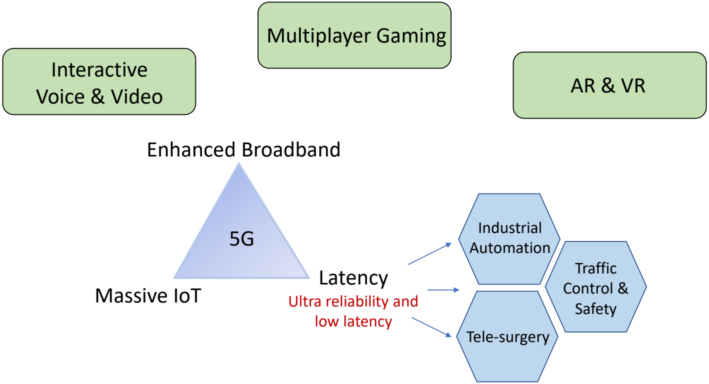
The GE Channel Model

Packet Erasure Model: The Gilbert-Elliott Channel



- Gilbert Elliott (GE) is a two-state channel model
  - **G** $\equiv$ Good State, **B** $\equiv$ Bad State
  - PEC is a packet-level erasure channel
    - ϵ_0 is the probability of packet erasure in good state
    - ϵ_1 is the probability of packet erasure in bad state
- capable of generating the random and burst erasures that one might encounter in practice

Potential Applications of Streaming Codes



- Commonality: Underlying each of these applications is:
  - a stream of packets: voice, video or data
  - the need for reliable, low-latency communication

The Sliding Window Approximation

Tractable Approximation to the GE Channel

Challenging to design codes for the GE channel to ensure a desired reliability level

Our approach (following work of Prof Ashish Khisti's group [1]):

- Replace the GE channel with a tractable approximation
- a sliding-window channel model that introduces burst and random erasures

[1] Badr et al., "Layered Constructions for Low-Delay Streaming Codes," *IEEE Trans. Inf. Theory*, 2017.

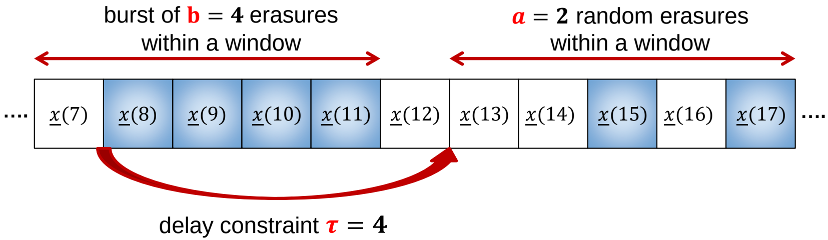
The Sliding-Window (SW) Channel Model

An **admissible erasure pattern** is one in which, within each sliding window of w-packet duration, there are

- either $\leq a$ random erasures,
- or else, a burst of $\leq b$ erasures

Eg.

$$(a = 2,\ b = 4,\ w = 5,\ \tau = 4)$$



The DCSW Channel Model

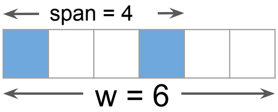
Combining the

- sliding-window channel model with the
- τ-packet decoding-delay constraint,

we arrive at the Delay-Constrained, Sliding Window (DCSW) channel model.

Span and Weight of an Erasure Pattern

- Consider a window of size $w = 6$.
- 2^6 erasure patterns possible.
- The example erasure pattern below has
  - (Hamming) weight 2 and
  - span 4.



GE Channel: Empirical Probabilities of Erasure Patterns

span	b = 6			0.0098%	0.0005%	0.0000%	0.0000%	0.0002%
	b = 5			0.020%	0.0008%	0.0001%	0.0005%	
	b = 4			0.029%	0.0008%	0.0017%		
	b = 3			0.04%	0.01%			
	b = 2			0.06%				
	b = 1		5.75%					
	b = 0	94.08%						
		a=0	a=1	a=2	a=3	a=4	a=5	a=6
		weight						

- Table shows empirical probabilities of
  - erasure patterns with weight a
  - and span b
  - over window of length $\tau + 1 = 6$
- of an example GE Channel: $(\ \alpha = 10^{-4},\ \beta = 0.6,\ \epsilon_0 = 0.01,\ \epsilon_1 = 1\ )$

- **Erasure probabilities in red add to $< 10^{-5}$.**

Our Consequent, Packet-Level FEC Approach

Given

- a GE channel model,
- a delay constraint τ, and
- a desired probability P_e of unrecoverable packets

our approach is to:

- select an $\{a, b, w, \tau\}$ DCSW channel approximation of highest rate achieving desired probability P_e of unrecoverable packets

- then employ a rate-optimal packet-level FEC for that $\{a, b, w, \tau\}$ channel

ToC

Linear Codes

Linear Codes

We begin with some background on linear codes

- An $[n, k]$ code $\mathcal{C}$ denotes a linear code of length n and dimension k
- By which we mean a subspace $\mathcal{C}$ of $\mathbb{F}_q^n$ having dimension k
- Code symbols are drawn from a finite field $\mathbb{F}_q$ of size q
- Rate R of the code is $\frac{k}{n}$

Linear Codes and Erasure Recovery

Let $[n, k]$ linear code $\mathcal{C}$ have parity-check matrix H. Then

$$\underline{c} \in \mathcal{C} \quad \text{iff} \quad H\underline{c} = \underline{0}.$$

If for example $n = 7$ and

$$H = \left[\; \underline{h}_1 \quad \underline{h}_2 \quad \underline{h}_3 \quad \underline{h}_4 \quad \underline{h}_5 \quad \underline{h}_6 \quad \underline{h}_7 \; \right],$$

and there are erasures in positions $1, 3, 6$, then

- the code can recover from all three erasures iff
  - $\{\underline{h}_1, \; \underline{h}_3, \; \underline{h}_6\}$ are linearly independent,

- on the other hand, to recover c_1 alone, it suffices if
  - $\underline{h}_1$ is linearly independent of $\{\underline{h}_3, \underline{h}_6\}$.

Hamming Code Recovery as an Example

- $[7, 4]$ Hamming code

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

- Suppose code symbols 1,3,6 are erased.

$$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} \quad \text{are linearly independent,}$$

- code can recover all three erased symbols

Hamming Code Recovery as an Example

- Suppose code symbols 1,2,3,6 are erased.

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

$$\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

- can not recover all four erased symbols together

$$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \text{ is linearly independent of } \left\{ \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} \right\}$$

- can recover c_1

MDS and Reed-Solomon Codes

Maximum Distance Separable (MDS) Codes

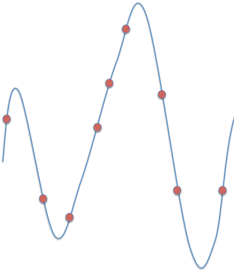
An $[n, k]$ MDS code is a linear block code over a finite field $\mathbb{F}_q$ that possess the 'any k of n' property

1. the message symbols can all be recovered if one has access to any k of n symbols
2. this class includes Reed-Solomon codes
3. the minimum Hamming distance $d_{\min}$ between a pair of distinct codewords in an MDS code satisfies the Singleton bound

$$d_{\min} \leq n - k + 1,$$

with equality and the codes are hence said to be maximum distance separable.

The Underlying Principle of Reed-Solomon (RS) Codes



- Assume that this is the plot of a polynomial of degree 5
- then its values at any 6 of the 9 points shown are sufficient to determine its values everywhere else
- thus the code can recover from 3 symbol erasures
- Here $(n = 9, k = 6, d_{\min} = 4)$, so

$$k = n - d_{\min} + 1.$$

All Arithmetic Takes Place in A Finite Field

$$\mathbb{F}_3 \quad \triangleq \quad \{0, 1, 2\}$$

Table: Addition

	0	1	2
0	0	1	2
1	1	2	0
2	2	0	1

Table: Multiplication

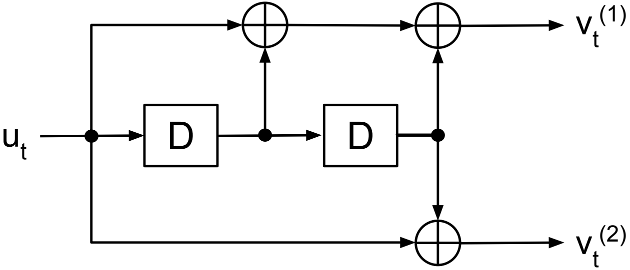
	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

(an example finite field of 3 elements under modulo-3 arithmetic)

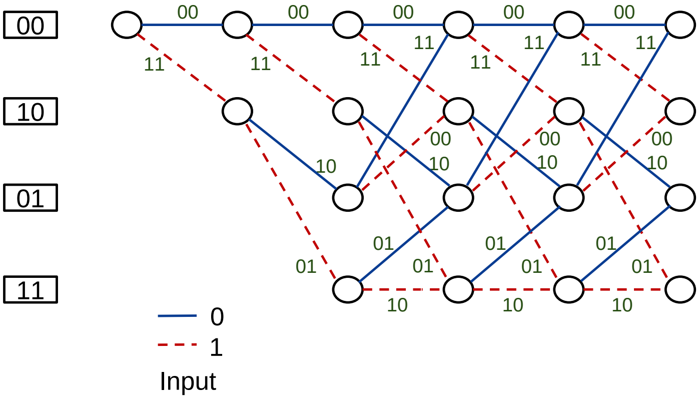
Convolutional Codes

Convolutional Code Example

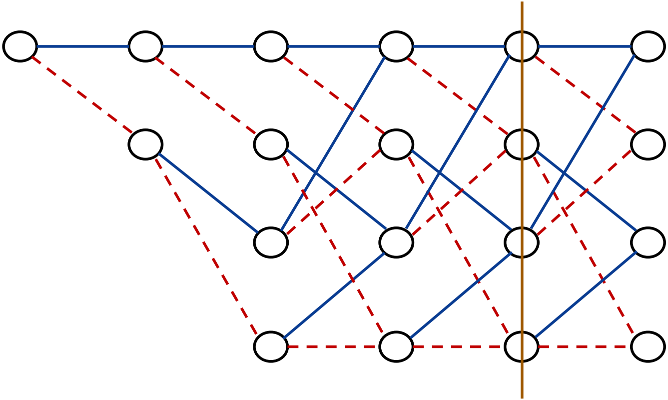
- rate $\frac{1}{2}$ code
- $k = 1, n = 2$
- Polynomial generator matrix $G(D) = \begin{bmatrix} 1 + D + D^2 & 1 + D^2 \end{bmatrix}$



Trellis Diagram



MDS Convolutional Code



Puncturing the convolutional code to this depth yields an MDS code

Return to the Streaming Code Setting

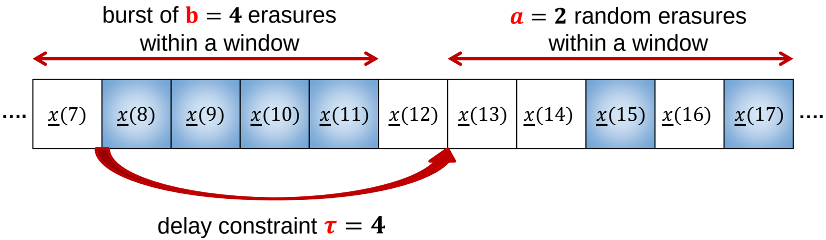
Recall: The Sliding-Window (SW) Channel Model

An **admissible erasure pattern** is one in which, within each sliding window of w-packet duration, there are

- either $\leq a$ random erasures,
- or else, a burst of $\leq b$ erasures

Eg.

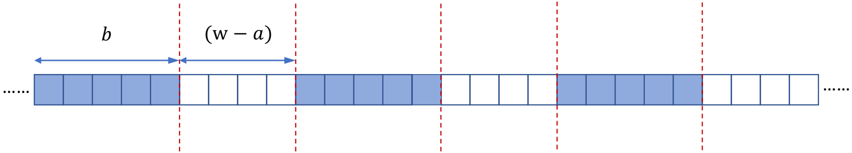
$$(a = 2,\ b = 4,\ w = 5,\ \tau = 4)$$



Key Upper Bound on Code Rate

Upper Bound on Code Rate R for the DCSW Channel

The erasure pattern below needs to be handled:

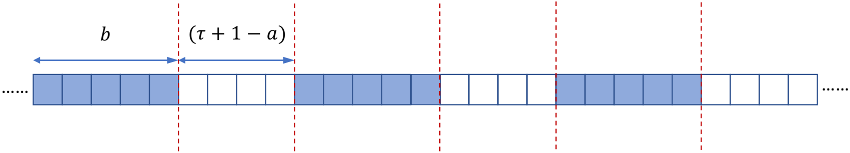


It follows that

$$R = \frac{k}{n} \leq \frac{w - a}{(w - a) + b}$$

Upper Bound on Code Rate R for the DCSW Channel

The erasure pattern below also needs to be handled:

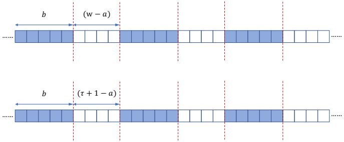


It follows that

$$R = \frac{k}{n} \leq \frac{\tau + 1 - a}{(\tau + 1 - a) + b}$$

Upper Bound on Code Rate R for the SW Channel

Thus both erasure patterns below need to be handled:



It follows that

$$R = \frac{k}{n} \ \leq \ \frac{\tau_{\text{eff}} + 1 - a}{(\tau_{\text{eff}} + 1 - a) \ + \ b}; \qquad \tau_{\text{eff}} + 1 = \min\{w, \tau + 1\}$$

But this is achievable with $w = \tau + 1$. Hence we assume WOLOG that

$$w \ = \ \tau + 1$$

and dispense with w. We will henceforth speak of an $\{a, b, \tau\}$ DCSW channel.

Badr et al., "Layered Constructions for Low-Delay Streaming Codes," *IEEE Trans. Inf. Theory*, 2017.

Sub-optimality of MDS Codes

MDS Codes are Sub-Optimal

Recall the rate upper bound for Sliding-Window Channel:

$$R \leq \frac{\tau + 1 - a}{(\tau + 1 - a) \ + \ b} \ := \ R_{\text{opt}}.$$

Let $\mathcal{C}$ be an $[n, k]$ MDS code. Then necessarily

$$\begin{aligned} n &\leq \tau + 1 \quad \text{(from the latency requirement)} \\ n - k &\geq b \quad \text{(to handle a burst of } b \text{ erasures).} \end{aligned}$$

Hence rate

$$\begin{aligned} R_{\text{MDS}} &= 1 - \frac{(n-k)}{n} \leq 1 - \frac{b}{\tau + 1} = \frac{\tau + 1 - b}{\tau + 1} \\ &< \frac{\tau + 1 - b \ + (\mathrm{b} - \mathrm{a})}{\tau + 1 \ + (\mathrm{b} - \mathrm{a})} = R_{\text{opt}}, \end{aligned}$$

if $a < b$. Thus MDS codes do not achieve R_{opt} for $a < b$.

Note on Minimum Possible Block Length

Recall the rate upper bound for Sliding-Window Channel:

$$R \leq \frac{\tau + 1 - a}{(\tau + 1 - a) \ + \ b} \ := \ R_{\text{opt}}.$$

- If numerator and denominator of R_{opt} expression are relatively prime, then $n = \tau + 1 - a + b$ is the minimum possible block length of rate-optimal code.

(we will indirectly exploit this fact when we bring back MDS codes later into the picture)

ToC

Block or Convolutional Coding

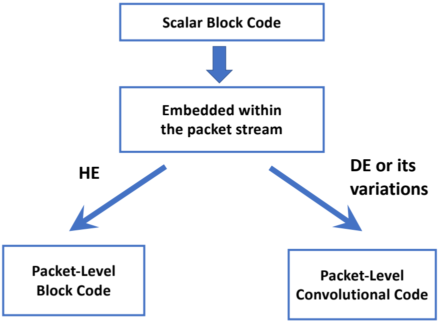
One could employ codes that at the packet-level are either

- block or
- convolutional codes

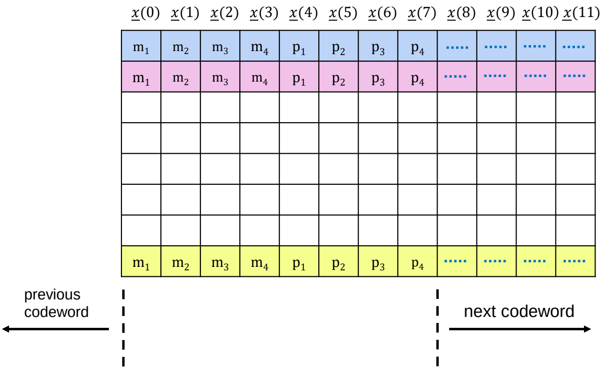
Our Approach

- Construct packet-level codes that are built out of scalar block codes using one of:

1. Horizontal Embedding (HE)
2. Diagonal Embedding (DE) or its variations



Horizontal-Embedding



- each column is a packet
- each row is a distinct codeword of an example $[8, 4]$ code $\mathcal{C}$
- leads to a packet-level block code
- we are in effect inserting redundant packets

Diagonal-Embedding (DE)

$\underline{x}(0)$	$\underline{x}(1)$	$\underline{x}(2)$	$\underline{x}(3)$	$\underline{x}(4)$	$\underline{x}(5)$	$\underline{x}(6)$	$\underline{x}(7)$	$\underline{x}(8)$	$\underline{x}(9)$	$\underline{x}(10)$	$\underline{x}(11)$
m_1	m_1	m_1									
	m_2	m_2	m_2								
		m_3	m_3	m_3							
			m_4	m_4	m_4						
				p_1	p_1	p_1					
					p_2	p_2	p_2				
						p_3	p_3	p_3			
							p_4	p_4	p_4		

- each column is a packet
- each diagonal is a distinct codeword of an example $[8, 4]$ code $\mathcal{C}$
- we are in effect, expanding each individual packet

Martinian and Trott, "Delay-Optimal Burst Erasure Code Construction," *ISIT*, 2007.

Diagonal-Embedding

$\underline{x}(0)$	$\underline{x}(1)$	$\underline{x}(2)$	$\underline{x}(3)$	$\underline{x}(4)$	$\underline{x}(5)$	$\underline{x}(6)$	$\underline{x}(7)$	$\underline{x}(8)$	$\underline{x}(9)$	$\underline{x}(10)$	$\underline{x}(11)$
m_1	m_1	m_1									
	m_2	m_2	m_2								
		m_3	m_3	m_3							
			m_4	m_4	m_4						
				p_1	p_1	p_1					
					p_2	p_2	p_2				
						p_3	p_3	p_3			
							p_4	p_4	p_4		

- a parity symbol in a packet is a function of message symbols in prior packets
- this is thus an instance of convolutional encoding

Staggered Diagonal Embedding (variant of DE)

- Codewords are embedded diagonally with **gaps** in the packet stream.
- The tiling of the 2D grid shown below under SDE, may be regarded as a kind of interleaving
- Such interleaving is not possible in general, with horizontal embedding

$\underline{x}(0)$	$\underline{x}(1)$	$\underline{x}(2)$	$\underline{x}(3)$	$\underline{x}(4)$	$\underline{x}(5)$	$\underline{x}(6)$	$\underline{x}(7)$	$\underline{x}(8)$	$\underline{x}(9)$	$\underline{x}(10)$	$\underline{x}(11)$
m_1	m_1	m_1									
	m_2	m_2	m_2								
		m_3	m_3	m_3							
			m_4	m_4	m_4						
						p_1	p_1	p_1			
							p_2	p_2	p_2		
								p_3	p_3	p_3	
									p_4	p_4	p_4

Dispersion of Code Symbols under SDE

- Dispersion span N: number of consecutive packets across which codewords are spread.

$\underline{x}(0)$	$\underline{x}(1)$	$\underline{x}(2)$	$\underline{x}(3)$	$\underline{x}(4)$	$\underline{x}(5)$	$\underline{x}(6)$	$\underline{x}(7)$	$\underline{x}(8)$	$\underline{x}(9)$	$\underline{x}(10)$	$\underline{x}(11)$
m_1	m_1	m_1									
	m_2	m_2	m_2								
		m_3	m_3	m_3							
			m_4	m_4	m_4						
						p_1	p_1	p_1			
							p_2	p_2	p_2		
								p_3	p_3	p_3	
									p_4	p_4	p_4

Dispersion Span N = 10

Generalized Diagonal Embedding (a second variant of DE)

- Similar to DE except that each codeword may have more than one symbol per packet

$\underline{x}(0)$	$\underline{x}(1)$	$\underline{x}(2)$	$\underline{x}(3)$	$\underline{x}(4)$	$\underline{x}(5)$	$\underline{x}(6)$	$\underline{x}(7)$	$\underline{x}(8)$	$\underline{x}(9)$
c_1	c_1	c_1	c_1	c_1					
c_2	c_2	c_2	c_2	c_2					
c_3	c_3	c_3	c_3	c_3					
	c_4	c_4	c_4	c_4	c_4				
		c_5	c_5	c_5	c_5	c_5			
			c_6	c_6	c_6	c_6	c_6		
				c_7	c_7	c_7	c_7	c_7	
					c_8	c_8	c_8	c_8	c_8
					c_9	c_9	c_9	c_9	c_9
					c_{10}	c_{10}	c_{10}	c_{10}	c_{10}

Streaming Codes

By Streaming Code, we will here mean

- a packet-level erasure-recovery code that is designed to efficiently communicate over the Sliding-Window channel
- while permitting decoding under decoding-delay constraint τ

ToC

Our Results

- Rate-Optimal Streaming Code Constructions:
  - Jigsaw Code
  - The Explicit Jigsaw Code
  - Simple Streaming (SS) Codes
  - Long SDE Code

- Near-rate-Optimal Streaming Code Constructions Having Low Field Size

- Performance Evaluation over the GE Channel
- An Experimental Attempt at Channel Adaptation

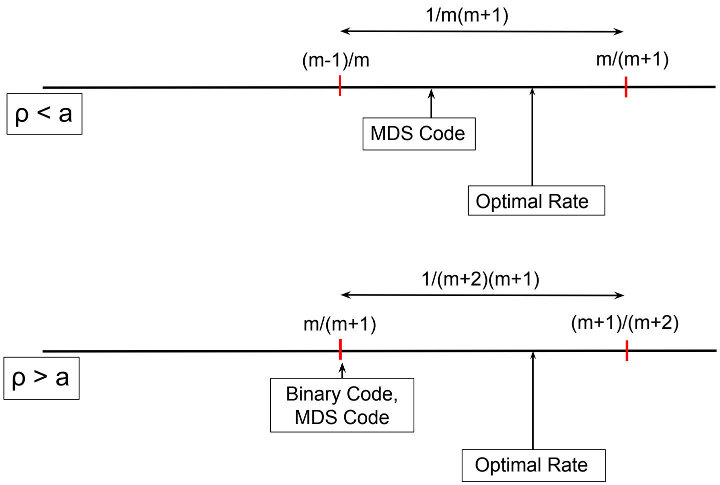
Rate-Optimal Constructions of Streaming Codes

Name	Embed-ding	Dispersion Span	Field Size	Based on MDS Codes ?	Parameter Range Covered
Jigsaw*	DE	$> (\tau + 1)$	Quadratic	No	all $\{a, b, \tau\}$
Dominovitz et al	DE	$> (\tau + 1)$	Quadratic	No	all $\{a, b, \tau\}$
Jigsaw (explicit)	DE	$> (\tau + 1)$	Quadratic	No	all $\{a, b, \tau\}$
SS codes	SDE	$= (\tau + 1)$	Linear	Yes	$\tau + 1 = a \pmod{b}$
Long SDE	SDE	$> (\tau + 1)$	Linear	No	$(\tau + 1 - a, b) = g$ $a \leq g < b$

* not fully explicit

Near Rate-Optimality of Simple Streaming Codes

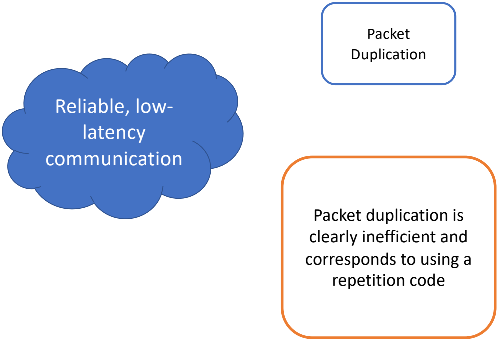
- $\tau + 1 = mb + \rho$
- Rate-optimal when $\rho = a$



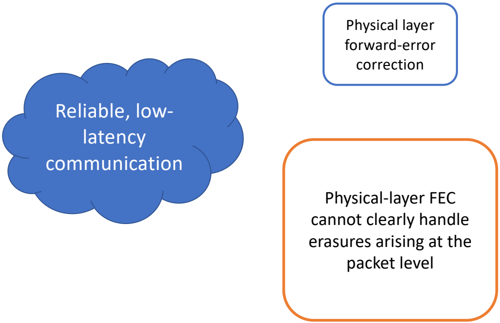
Other Approaches to

Reliable Low-Latency Communication

Other Approaches: Packet Duplication



Other Approaches: Physical-Layer FEC



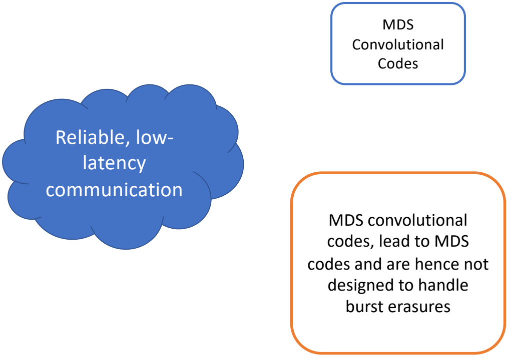
Other Approaches: ARQ

Reliable, low-latency communication

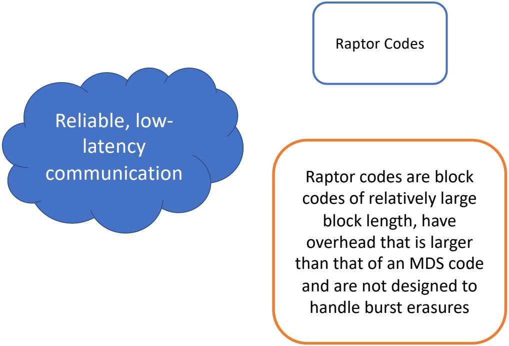
Automatic Repeat Request

ARQ schemes incur an undesirable round-trip transmission delay

Other Approaches: MDS Convolutional Codes



Other Approaches: Raptor Codes



ToC

Streaming Codes

By Streaming Code, we will here mean

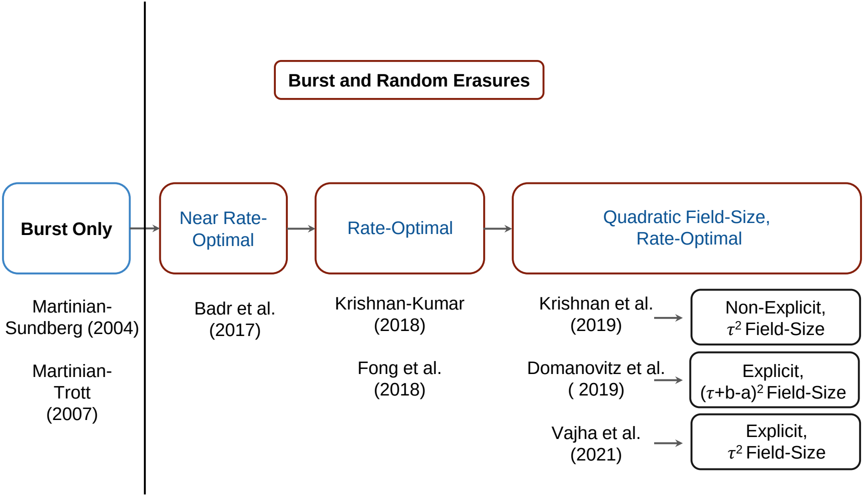
- a packet-level erasure-recovery code that is designed to efficiently communicate over the Sliding-Window channel
- while permitting decoding under decoding-delay constraint τ

Work on streaming codes

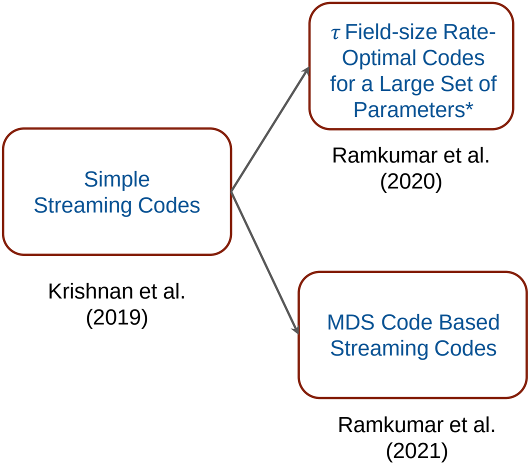
- originated at MIT
- continued at the University of Toronto (Prof Ashish Khisti's group)
- continued at the Indian Institute of Science (our group)

Literature on Streaming Codes

Literature on Streaming Codes



Low-Complexity Streaming Codes



$*\ \gcd(b, \tau + 1 - a) \in \{a, a+1, a+2, \cdots, b\}$

Streaming Codes for Other Settings

Multicast Over Burst Erasure Channels	Variable Message Packet Size	3 Node Relay Network
Badr et al. (2011, 2015)	Rudow-Rashmi (2018)	Fong et al. (2020)

Construction Requirements

Diagonal Embedding

$\underline{x}(0)$	$\underline{x}(1)$	$\underline{x}(2)$	$\underline{x}(3)$	$\underline{x}(4)$	$\underline{x}(5)$	$\underline{x}(6)$	$\underline{x}(7)$	$\underline{x}(8)$	$\underline{x}(9)$	$\underline{x}(10)$	$\underline{x}(11)$
m_1	m_1	m_1									
	m_2	m_2	m_2								
		m_3	m_3	m_3							
			m_4	m_4	m_4						
				p_1	p_1	p_1					
					p_2	p_2	p_2				
						p_3	p_3	p_3			
							p_4	p_4	p_4		

- Focus on diagonal embedding of scalar block code.

Requirement of a Rate-Optimal Code

Basic idea:

- design the code so that one can recover from erasure patterns given **only** the next τ code symbols.

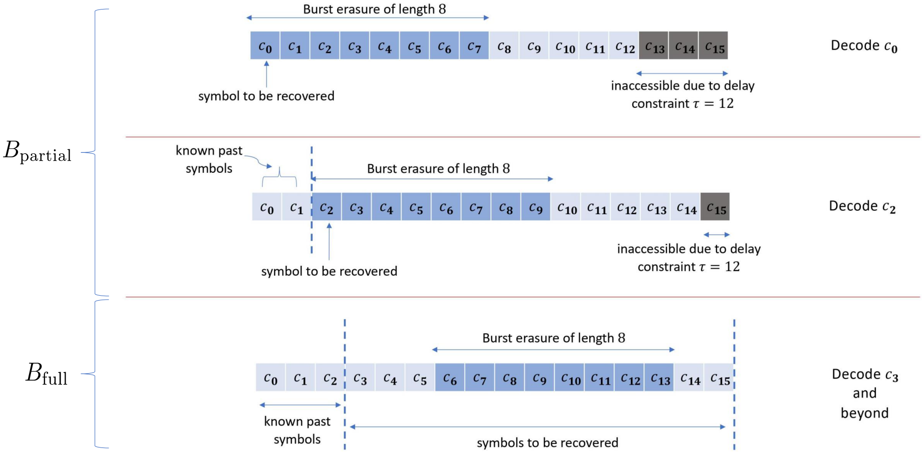
We illustrate with an example:

- $\{a, b, \tau\} = \{5, 8, 12\}$

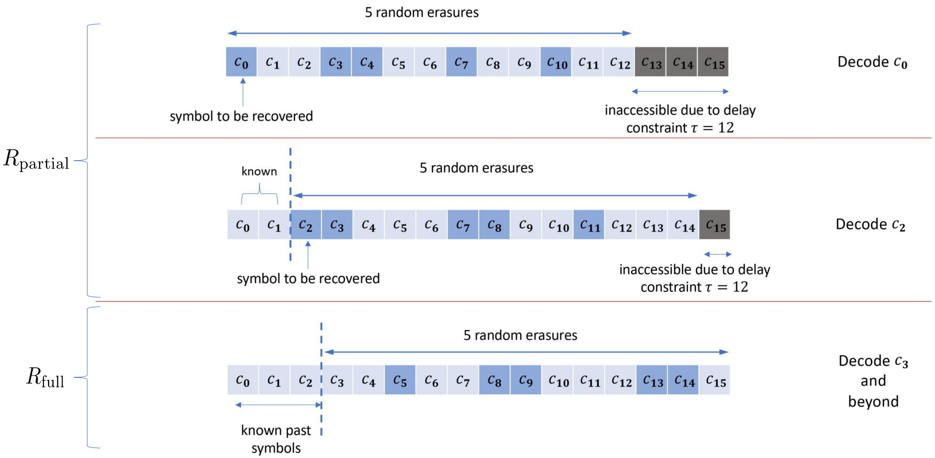
- $R_{\text{opt}} \;=\; \frac{\tau+1-a}{\tau+1-a+b} = \frac{8}{16} = 0.5$

- Consider an $[n = 16, k = 8]$ code $\mathcal{C}$.

Burst Erasure Correction - Requirements



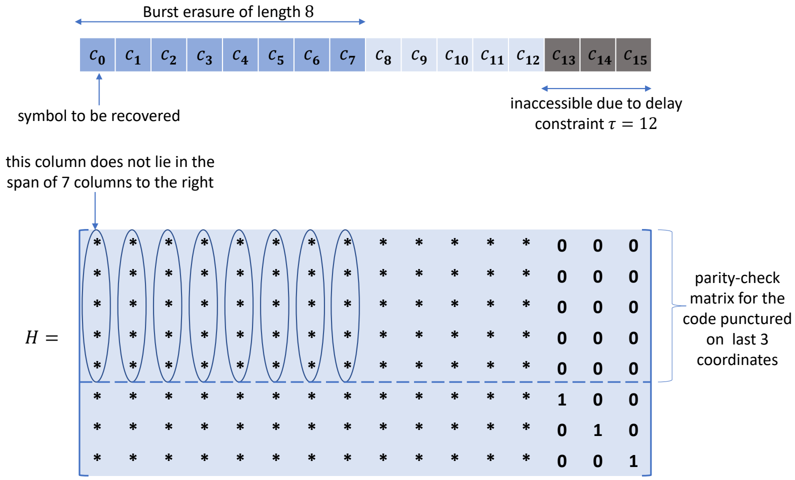
Random Erasure Correction - Requirements



Implications of Requirements on the Parity-Check Matrix

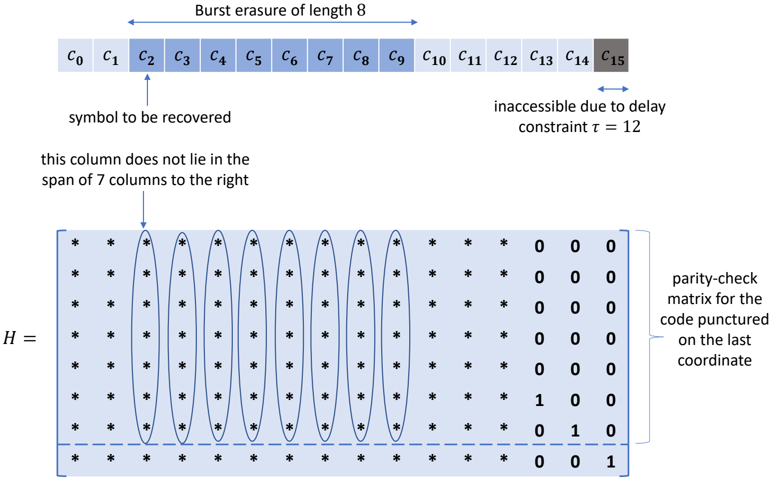
Decoding c_0 in the Presence of a Burst (B_{partial})

- $\{a, b, \tau\} = \{5, 8, 12\}$
- Goal: recover c_0 while c_{13}, c_{14}, c_{15} are inaccessible



Decoding c_2 in the Presence of a Burst (B_{partial})

- $\{a, b, \tau\} = \{5, 8, 12\}$
- Goal: recover c_2 while c_{15} is inaccessible



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

Decoding $(c_6, c_7, \cdots, c_{13})$ in the Presence of a Burst (B_{full})

- $\{a, b, \tau\} = \{5, 8, 12\}$

Burst erasure of length 8

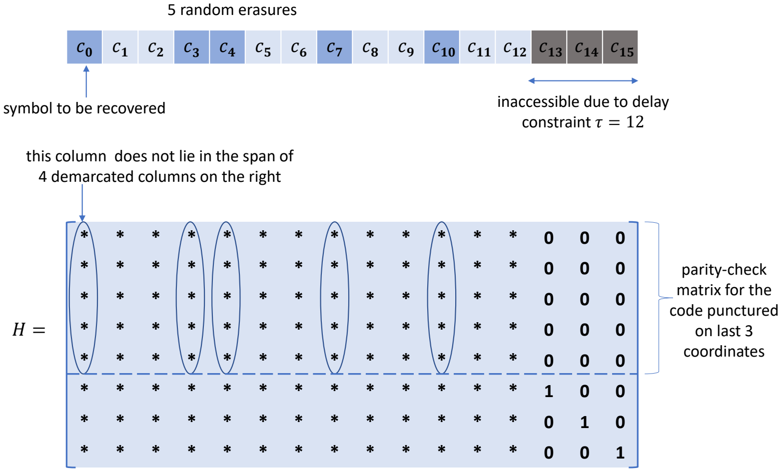
c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}

matrix is non-singular

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

Decoding c_0 in the Presence of Random Erasures (R_{partial})

- $\{a, b, \tau\} = \{5, 8, 12\}$
- Goal: recover c_0 while c_{13}, c_{14}, c_{15} are inaccessible



Decoding $(c_5, c_8, c_9, c_{13}, c_{14})$ in the Presence of Random Erasures (R_{full})

- $\{a, b, \tau\} = \{5, 8, 12\}$

5 random erasures

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}

the five marked columns are independent

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

The Jigsaw Code

M. N. Krishnan, D. Shukla, and P. V. Kumar, "Low Field-size, Rate-Optimal Streaming Codes for Channels With Burst and Random Erasures," *IEEE Trans. IT*, 2020.

An Example Construction

- $\{a, b, \tau\} = \{5, 8, 12\}$

- We construct an $[n = 16, k = 8]$ code $\mathcal{C}$.

An Example Construction: $\{a = 5, b = 8, \tau = 12\}$

- parity-check matrix H shown below
- provides codes for all $\{a, b, \tau\}$
- built up in segments like a jigsaw puzzle
- field size $q \in O(\tau^2)$

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0	0	0	0	*	*	*	*	*	*	*	*			
0	1	0	0	0	*	*	*	*	*	*	*	*			
0	0	1	0	0	*	*	*	*	*	*	*	*			
0	0	0	1	0	*	*	*	*	*	*	*	*			
0	0	0	0	1	*	*	*	*	*	*	*	*			

Any 5 columns are linearly independent

An Example Construction: $\{a = 5, b = 8, \tau = 12\}$

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	0	0	0	0	*	*	*	*	*	*	*	*			
0	1	0	0	0	*	*	*	*	*	*	*	*			
0	0	1	0	0	*	*	*	*	*	*	*	*			
0	0	0	1	0	*	*	*	*	*	*	*	*			
0	0	0	0	1	*	*	*	*	*	*	*	*			

Any 5 columns are linearly independent

An Example Construction: $\{a = 5, b = 8, \tau = 12\}$

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
			0	0	0	0	0						0	0	0
			0	0	0	0	0	0						0	0
			0	0	0	0	0	0	0						0
1	0	0	0	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	0	0	
0	1	0	0	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	0	0	0
0	0	1	0	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	0	0	0
0	0	0	1	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	0	0	0
0	0	0	0	1	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	0	0	0

Any 5 columns are linearly independent

An Example Construction: $\{a = 5, b = 8, \tau = 12\}$

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
			0	0	0	0	0						0	0	0
			0	0	0	0	0	0						0	0
			0	0	0	0	0	0	0						0
1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Any 5 columns are linearly independent

An Example Construction: $\{a = 5, b = 8, \tau = 12\}$

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
			0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0	0
			0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0	0
			0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	0
1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

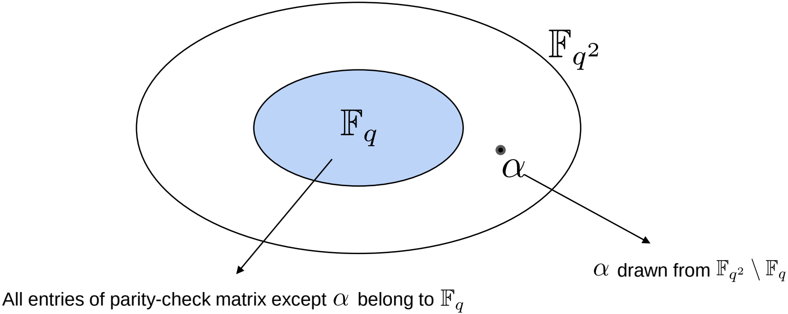
Any 5 columns are linearly independent

An Example Construction: $\{a = 5, b = 8, \tau = 12\}$

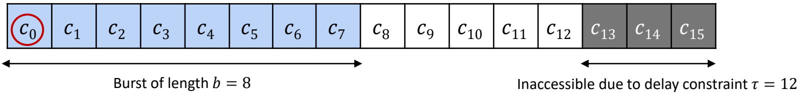
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0	0
0	α	0	0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0	0
0	0	α	0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	0
1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Any 5 columns are linearly independent

A Note on α



Recovery from Burst Erasures



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0	0
1	0	α	0	0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0	0
2	0	0	α	0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

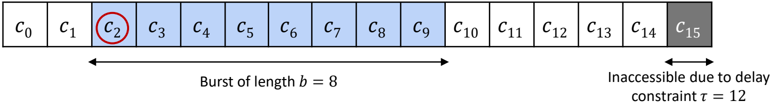
Recovery from Burst Erasures

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}

Burst of length $b = 8$

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4
4	0	1	0	0	0	*	*	*	*	*	*	*	*
5	0	0	1	0	0	*	*	*	*	*	*	*	*
6	0	0	0	1	0	*	*	*	*	*	*	*	*
7	0	0	0	0	1	*	*	*	*	*	*	*	*

Recovery from Burst Erasures



	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0	0
1	0	α	0	0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0	0
2	0	0	α	0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Recovery from Burst Erasures

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}

Burst of length $b = 8$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0
1	0	α	0	0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0
2	0	0	α	0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0

Recovery from Burst Erasures

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}

Burst of length $b = 8$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0	0
1	0	α	0	0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0	0
2	0	0	α	0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Recovery from Burst Erasures

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}

Burst of length $b = 8$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0	0
1	0	α	0	0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0	0
2	0	0	α	0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Recovery from Burst Erasures

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}

Burst of length $b = 8$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0	0
1	0	α	0	0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0	0
2	0	0	α	0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Recovery from Burst Erasures

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}

Burst of length $b = 8$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0	0
1	0	α	0	0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0	0
2	0	0	α	0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Recovery from Burst Erasures

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}

Burst of length $b = 8$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0	0
1	0	α	0	0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0	0
2	0	0	α	0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Recovery from Burst Erasures

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}

Burst of length $b = 8$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0	0
1	0	α	0	0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0	0
2	0	0	α	0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Recovery from Random Erasures

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}

Inaccessible due to delay constraint $\tau = 12$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0	0
1	0	α	0	0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0	0
2	0	0	α	0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Recovery from Random Erasures

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}

	0	1	2	3	4	5	6	7	8	9	10	11	12
0	α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4
4	0	1	0	0	0	*	*	*	*	*	*	*	*
5	0	0	1	0	0	*	*	*	*	*	*	*	*
6	0	0	0	1	0	*	*	*	*	*	*	*	*
7	0	0	0	0	1	*	*	*	*	*	*	*	*

Recovery from Random Erasures

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}

Inaccessible due to delay constraint $\tau = 12$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0	0
1	0	α	0	0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0	0
2	0	0	α	0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Recovery from Random Erasures

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0
1	0	α	0	0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0
2	0	0	α	0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0

Recovery from Random Erasures

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0
1	0	α	0	0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0
2	0	0	α	0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0

coefficients of linear combination over $\mathbb{F}_q$

Recovery from Random Erasures

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
0	α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0
1	0	α	0	0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0
2	0	0	α	0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0

drawn from $\mathbb{F}_{q^2} \setminus \mathbb{F}_q$

coefficients of linear combination over $\mathbb{F}_q$

Recovery from Random Erasures

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0	0
1	0	α	0	0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0	0
2	0	0	α	0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

The Explicit Jigsaw Code

M. Vajha, V. Ramkumar, M. N. Krishnan, and P. V. Kumar, "Explicit Rate-Optimal Streaming Codes with Smaller Field Size," *ISIT* 2021.

An Example

- Returning to the same parameters ($a = 5, b = 8, \tau = 12$).
- We start with parity check of Jigsaw construction

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
α	0	0	0	0	0	0	0	v_0	v_1	v_2	v_3	v_4	0	0	0
0	α	0	0	0	0	0	0	0	v_5	v_6	v_7	v_8	v_9	0	0
0	0	α	0	0	0	0	0	0	0	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	0
1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Any 5 columns are linearly independent

Explicit Jigsaw Code: Parity Check Matrix

- $(a = 5, b = 8, \tau = 12)$.
- Replace some v_i's with matrix P

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0					α	0	0	0
1	0	α	0	0	0	0	0	0		P			0	1	0	0
2	0	0	α	0	0	0	0	0					0	0	1	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Explicit Jigsaw Code: Parity Check Matrix

- $(a = 5, b = 8, \tau = 12)$
- After setting the value for P

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	1	0	0	0	α	0	0	0
1	0	α	0	0	0	0	0	0	0	1	0	0	0	1	0	0
2	0	0	α	0	0	0	0	0	0	0	1	0	0	0	1	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Explicit Jigsaw Code

- $(a = 5, b = 8, \tau = 12)$
- We will see $\mathrm{B}_{\mathrm{full}}$ property where burst starts at index $i \in [3:8]$.
- The properties $\mathrm{R}_{\mathrm{partial}}$, $\mathrm{R}_{\mathrm{full}}$, $\mathrm{B}_{\mathrm{partial}}$ go through due to similar structure as previous construction.

Recovering from Burst at index 3

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}

Burst of length $b = 8$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	1	0	0	0	α	0	0	0
1	0	α	0	0	0	0	0	0	0	1	0	0	0	1	0	0
2	0	0	α	0	0	0	0	0	0	0	1	0	0	0	1	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Recovering from Burst at index 4

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}

Burst of length $b = 8$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	1	0	0	0	α	0	0	0
1	0	α	0	0	0	0	0	0	0	1	0	0	0	1	0	0
2	0	0	α	0	0	0	0	0	0	0	1	0	0	0	1	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

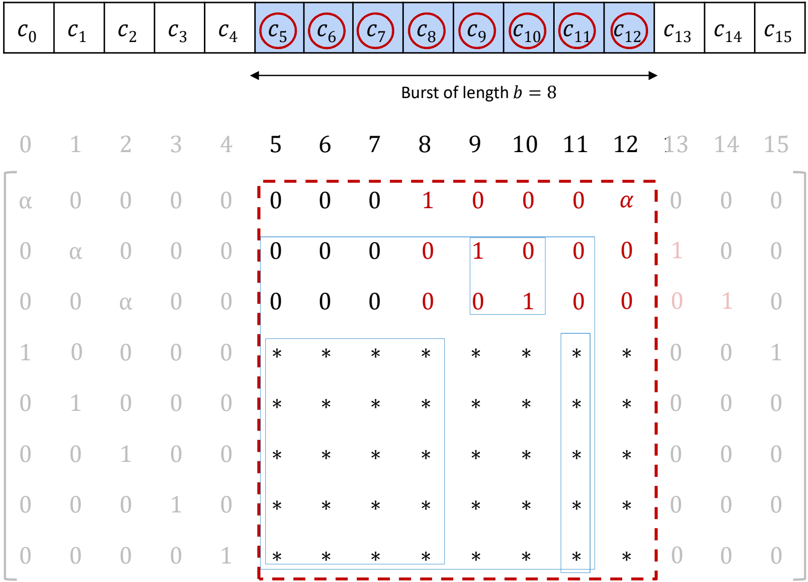
Recovering from Burst at index 5

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}

Burst of length $b = 8$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	1	0	0	0	α	0	0	0
1	0	α	0	0	0	0	0	0	0	1	0	0	0	1	0	0
2	0	0	α	0	0	0	0	0	0	0	1	0	0	0	1	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Recovering from Burst at index 5



Recovering from Burst at index 6

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}

Burst of length $b = 8$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	1	0	0	0	α	0	0	0
1	0	α	0	0	0	0	0	0	0	1	0	0	0	1	0	0
2	0	0	α	0	0	0	0	0	0	0	1	0	0	0	1	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Recovering from Burst at index 7

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}

Burst of length $b = 8$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	1	0	0	0	α	0	0	0
1	0	α	0	0	0	0	0	0	0	1	0	0	0	1	0	0
2	0	0	α	0	0	0	0	0	0	0	1	0	0	0	1	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Recovering from Burst at index 8

c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}

Burst of length $b = 8$

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	1	0	0	0	α	0	0	0
1	0	α	0	0	0	0	0	0	0	1	0	0	0	1	0	0
2	0	0	α	0	0	0	0	0	0	0	1	0	0	0	1	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

Desired Properties of P

- To satisfy $\mathrm{B}_{\mathrm{full}}$ property, intuitively we want the first δ rows of the $b \times b$ sub-matrix of parity check matrix to be comprised of
  - a zero columns and
  - $\delta \times \delta$ invertible matrix.
- Then the Cauchy property of C can be leveraged.

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	α	0	0	0	0	0	0	0	1	0	0	0	α	0	0	0
1	0	α	0	0	0	0	0	0	0	1	0	0	0	1	0	0
2	0	0	α	0	0	0	0	0	0	0	1	0	0	0	1	0
3	1	0	0	0	0	*	*	*	*	*	*	*	*	0	0	1
4	0	1	0	0	0	*	*	*	*	*	*	*	*	0	0	0
5	0	0	1	0	0	*	*	*	*	*	*	*	*	0	0	0
6	0	0	0	1	0	*	*	*	*	*	*	*	*	0	0	0
7	0	0	0	0	1	*	*	*	*	*	*	*	*	0	0	0

The Assignment of P

- $P = P^{a}_{\delta,\tau-b}$
- Recursive construction of $(u \times v)$ matrix $P^{a}_{u,v}$.

$$
\mathrm{P}^{a}_{u,v} = \begin{cases}
\left[\begin{array}{ccc} I_u & \underbrace{0}_{(u\times a)} & \mathrm{P}^{a}_{u,v-u-a} \end{array}\right] & u+a<v \\
\left[\begin{array}{cc} I_u & \underbrace{0}_{(u\times(v-u))} \end{array}\right] & u \le v \le u+a \\
\left[\begin{array}{c} I_v \\ \mathrm{P}^{a}_{u-v,v} \end{array}\right] & v<u
\end{cases}
$$

For example:

$$
P^{2}_{5,3} = \left[\begin{array}{c} I_3 \\ P^{2}_{2,3} \end{array}\right]
= \left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \hline 1 & 0 & 0 \\ 0 & 1 & 0 \end{array}\right]
$$

$$
P^{1}_{3,1} = \left[\begin{array}{c} 1 \\ 1 \\ 1 \end{array}\right]
$$

Staggered Diagonal Embedding

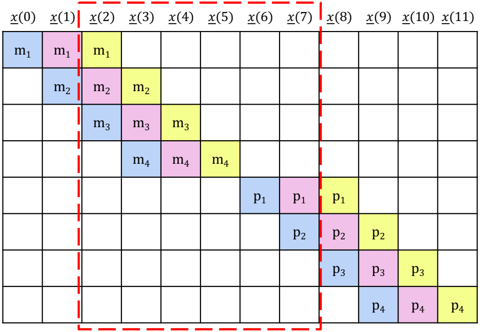
Recall: The Diagonal-Embedding Approach

$\underline{x}(0)$	$\underline{x}(1)$	$\underline{x}(2)$	$\underline{x}(3)$	$\underline{x}(4)$	$\underline{x}(5)$	$\underline{x}(6)$	$\underline{x}(7)$	$\underline{x}(8)$	$\underline{x}(9)$	$\underline{x}(10)$	$\underline{x}(11)$
m_1	m_1	m_1									
	m_2	m_2	m_2								
		m_3	m_3	m_3							
			m_4	m_4	m_4						
				p_1	p_1	p_1					
					p_2	p_2	p_2				
						p_3	p_3	p_3			
							p_4	p_4	p_4		

- Each diagonal is a distinct codeword in $[n = 8, k = 4]$ code $\mathcal{C}$ here.

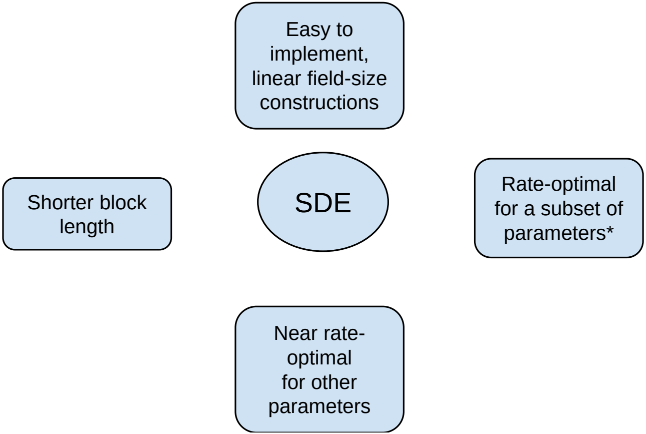
Variation: Staggered Diagonal Embedding (SDE)

- Codewords are embedded diagonally with **gaps** in the packet stream.
- This is in effect a form of interleaving made possible by diagonal embedding
- Reduces a burst of 6 erasures to a burst of length 4 in the example shown below.



M. N. Krishnan, V. Ramkumar, M. Vajha, and P. V. Kumar, "Simple streaming codes for reliable, low-latency communication," *IEEE Comm. Letters*, 2020.

Why Staggered Diagonal Embedding?



$*$ $\gcd(b, \tau + 1 - a) \in \{a, a+1, a+2, \cdots, b\}$

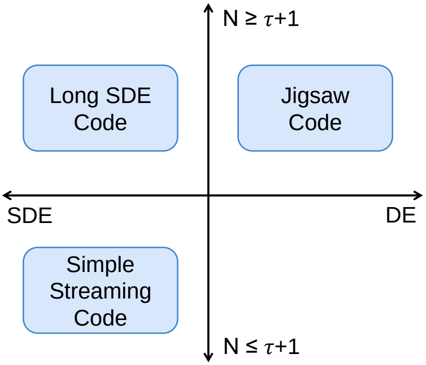
Dispersion of Code Symbols under SDE

- Base code: $[n, k]$ scalar block code
- Dispersion span N: number of consecutive packets across which codewords are spread.

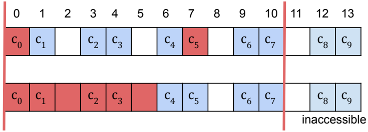
$\underline{x}(0)$	$\underline{x}(1)$	$\underline{x}(2)$	$\underline{x}(3)$	$\underline{x}(4)$	$\underline{x}(5)$	$\underline{x}(6)$	$\underline{x}(7)$	$\underline{x}(8)$	$\underline{x}(9)$	$\underline{x}(10)$	$\underline{x}(11)$
m_1	m_1	m_1									
	m_2	m_2	m_2								
		m_3	m_3	m_3							
			m_4	m_4	m_4						
						p_1	p_1	p_1			
							p_2	p_2	p_2		
								p_3	p_3	p_3	
									p_4	p_4	p_4

Dispersion Span N = 10

SDE: Two Regimes



- $N > \tau + 1 \implies$ partial knowledge decoding is needed.



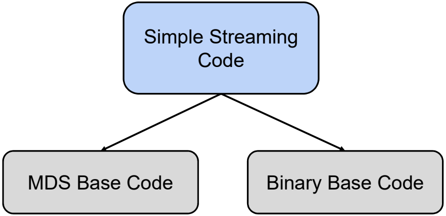
Partial knowledge recovery of c_0

Simple Streaming Codes

M. N. Krishnan, V. Ramkumar, M. Vajha, and P. V. Kumar, "Simple streaming codes for reliable, low-latency communication," *IEEE Comm. Letters*, 2020.

Simple Streaming Codes: SDE with $N \leq \tau + 1$

- Setting $N \leq \tau + 1$, ensures that no partial knowledge decoding is needed.
- Constituent base codes are MDS or binary cyclic codes.

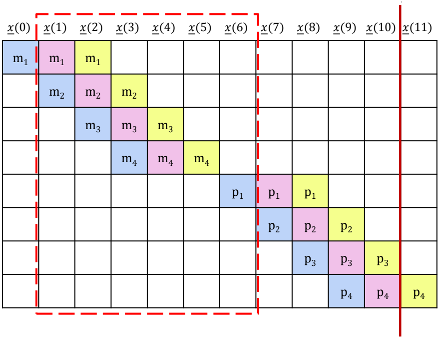


- Rate optimal with smaller block length and linear field size for

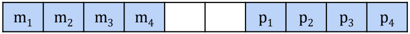
$$\tau + 1 \quad = \quad a \pmod{b}.$$

- Near optimal in terms of rate for other parameter sets.

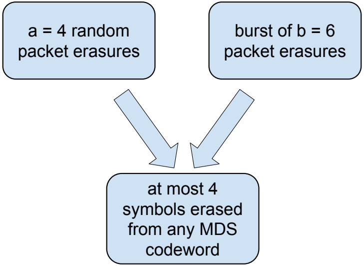
MDS-Code-Based Simple Streaming Codes: An Example



$(a = 4, b = 6, \tau = 9)$ streaming code

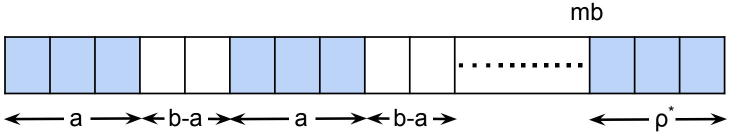


- SDE of $[8, 4]$ MDS code
- $N = 10 = \tau + 1$
  $\implies$ delay-constraint satisfied
- Rate $= \frac{1}{2} = R_{\text{opt}}$



MDS-Code-Based Simple Streaming Codes

- $\tau + 1 = mb + \rho, \; 0 \leq \rho < b$
- $\rho^* = min\{\rho, a\}$.
- Pick $[n = ma + \rho^*, \; k = (m-1)a + \rho^*]$ MDS code



- burst of b packet erasures $\implies$ erasure of $(n-k) = a$ consecutive code symbols

Binary-Code-Based Simple Streaming Codes for $\rho > a$

- $[7, 4]$ binary Hamming code

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

- The above code can recover from any burst of 3 erasures.
- $[n, k]$ cyclic codes have $(n - k)$ burst erasure recovery capability.
- This property is utilized to come up with binary-code-based simple streaming codes.

Binary-Code-Based Simple Streaming Codes for $\rho > a$

- $[7, 4]$ binary Hamming code

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

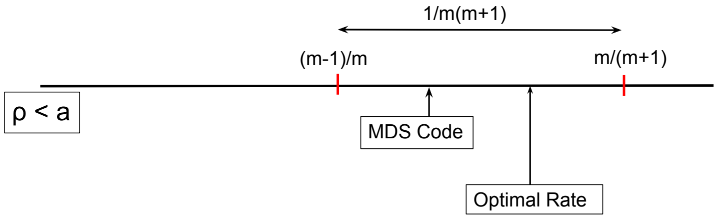
- The above code can recover from any burst of 3 erasures.

- $[n, k]$ cyclic codes have $(n - k)$ burst erasure recovery capability.

- This property is utilized to come up with binary-code-based simple streaming codes.

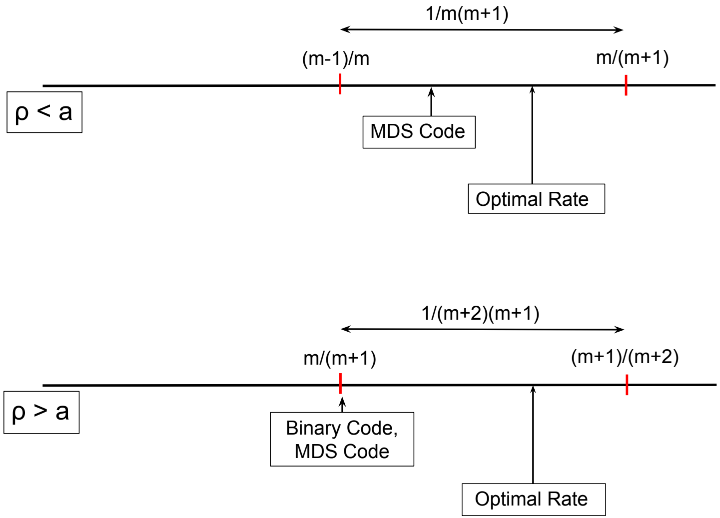
Near Rate-Optimality of Simple Streaming Codes

- $\tau + 1 = mb + \rho$
- Rate-optimal when $\rho = a$



Near Rate-Optimality of Simple Streaming Codes

- $\tau + 1 = mb + \rho$
- Rate-optimal when $\rho = a$



Rate Comparison for Some Parameters: MDS Base Code

a	b	τ	N	$n-k$	n	k	R_{MDS}	R_{opt}
2	3	3	4	2	3	1	0.333	0.4
2	3	4	5	2	4	2	0.5	0.5
2	3	5	6	2	4	2	0.5	0.571
3	5	5	6	3	4	1	0.25	0.375
3	5	11	12	3	8	5	0.625	0.6429
3	5	12	13	3	9	6	0.666	0.666
3	5	13	13	3	9	6	0.666	0.6875

- For some parameters it is possible to get better rate than this while retaining the simplicity of employing MDS codes.

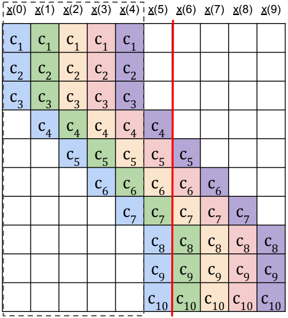
Rate Comparison for Some Parameters: MDS Base Code

a	b	τ	N	$n-k$	n	k	R_{MDS}	R_{opt}
2	3	3	4	2	3	1	0.333	0.4
2	3	4	5	2	4	2	0.5	0.5
2	3	5	6	2	4	2	0.5	0.571
3	5	5	6	3	4	1	0.25	0.375
3	5	11	12	3	8	5	0.625	0.6429
3	5	12	13	3	9	6	0.666	0.666
3	5	13	13	3	9	6	0.666	0.6875

- For some parameters it is possible to get better rate than this while retaining the simplicity of employing MDS codes.

Generalized Diagonal Embedding

- Allows embedding of more than one symbol of an MDS codeword within a single coded packet.
- Rate improvement possible for some cases, by exploiting increase in block length.

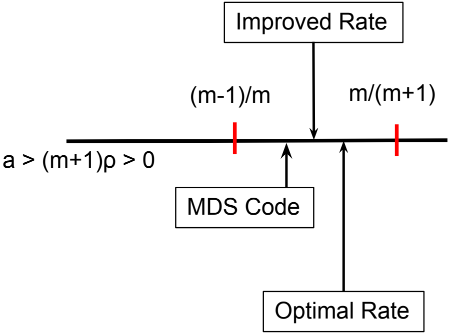


$(a = 3, b = 5, \tau = 5)$ Example

- $[10, 3]$ MDS code as base code
- $N = 6 = \tau + 1$
- No more than 7 code symbols erased from any MDS codeword
- Rate $= 0.3 > 0.25 = R_{\text{MDS}}$

Rate Improvement

- Rate increase happens if $b > a > (m+1)\rho > 0$.



V. Ramkumar, M. Vajha, and P. V. Kumar, "Generalized Simple Streaming Codes from MDS Codes," *ISIT* 2021.

Rate Comparison for Some Parameters: Binary Base Code

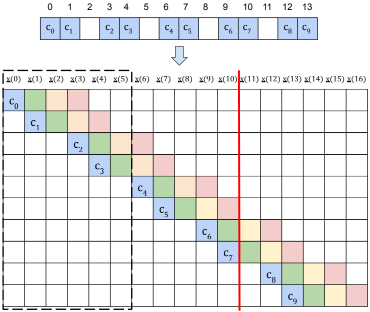
a	b	τ	N	r	n	k	R_{binary}	R_{opt}
3	6	9	10	4	8	4	0.5	0.538
3	7	10	11	4	8	4	0.5	0.533
3	6	16	17	5	15	10	0.666	0.7
3	7	19	19	5	15	10	0.666	0.708
3	8	21	21	5	15	10	0.666	0.703
3	9	22	23	5	15	10	0.666	0.689

Long SDE Code

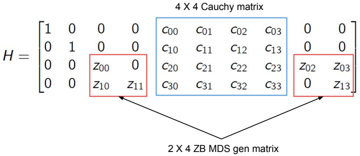
V. Ramkumar, M. Vajha, M. N. Krishnan, and P. V. Kumar, "Staggered Diagonal Embedding Based Linear Field Size Streaming Codes," *ISIT* 2020.

Long SDE Code: SDE with $N > \tau + 1$

- $N > \tau + 1 \implies$ partial-knowledge decoding is required



SDE of $[10, 6]$ scalar code with $N = 14$ to construct $(a = 2, b = 6, \tau = 10)$ streaming code



Parity check matrix of $[10, 6]$ scalar code

- $O(\tau)$ field-size scalar code, but not MDS code

- This rate-optimal construction works provided:

$$\gcd(b, \tau + 1 - a) \in \{a, a+1, a+2, \cdots, b-1\}.$$

ToC

Block Erasure Probability of Streaming Code Over GE Channel

M. Vajha, V. Ramkumar, M. Jhamtani, P. V. Kumar, "On Sliding Window Approximation of Gilbert-Elliott Channel for Delay Constrained Setting," *arXiv* 2020.

Guaranteeing Reliability

Given

- a GE channel model,
- a delay constraint τ, and
- a desired block erasure probability (BEP) P_e

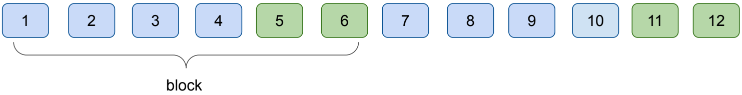
- select best rate (by choosing a, b), $\{a, b, \tau\}$ streaming code such that BEP$\leq P_e$

Computing BEP Using Probability of Admissible Erasure Patterns

- Rate-optimal streaming codes with following parameters exist for any (a, b, τ):

$$(n = \tau + 1 + b - a, k = n - b)$$

- Horizontal embedding results in a block streaming code

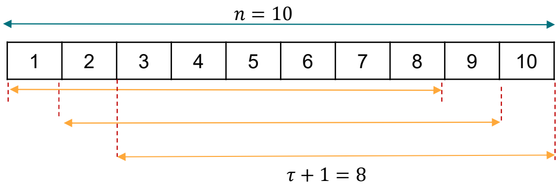


- BEP over $GE(\alpha, \beta, \epsilon_0, \epsilon_1)$

$$BEP(n, a, b, \tau) \quad = \quad 1 - P(\text{AEP})$$

AEP: set of admissible erasure patterns of an (a, b, τ) DCSW channel over a length n.

Admissible Erasure Patterns



$$\text{AEP} \quad = \quad \cap_{i=1}^{n-\tau}(A_i \cup B_i)$$

- A_i is the set of erasure patterns that have weight $\leq a$ in window $[i : i + \tau]$
- B_i is the set of erasure patterns that have span $\leq b$ in window $[i : i + \tau]$

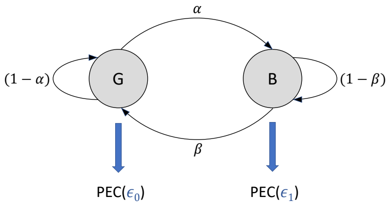
Goal: To get a handle on the $P(\text{AEP})$

What is Known for GE Channels ?

Computing $P(\text{AEP})$

- Closed form expression for $P(A_i)$ and $P(B_i)$ known.
- We provide an expression for $P(A_i \cup B_i)$
- Characterising $P(\text{AEP}) = P(\cap_{i=1}^{n-\tau}(A_i \cup B_i))$ is hard.
- We come up with bounds for $P(\text{AEP})$.

Computing Probability of an Erasure Pattern



Can show that

$$P(E_1^n = e_1^n) = 1^T \Psi(e_n) \cdots \Psi(e_1) \underline{\pi}$$

- $\underline{\pi} = [\frac{\beta}{\alpha+\beta} \quad \frac{\alpha}{\alpha+\beta}]$ is the stationary probability vector
- Ψ is defined as below:

$$\Psi(e) = \begin{cases} \Lambda S & e = 1 \\ (I - \Lambda) S & e = 0 \end{cases}$$

- $S = \underbrace{\begin{bmatrix} 1-\alpha & \beta \\ \alpha & 1-\beta \end{bmatrix}}_{\text{transitional probability matrix}}$ and $\Lambda = \begin{bmatrix} \epsilon_0 & \\ & \epsilon_1 \end{bmatrix}$.
- Notice that $\Psi(0) + \Psi(1) = S$

Computing Random Erasure Probabilities

- Let A be the set of erasures whose weight is atmost a in window of length n.

$$P(A) = \sum_{i=0}^{a} \underbrace{P(w(E_1^n) = i)}_{\text{closed form expression known}}$$

- BEP of an $[n, k = n - a]$ MDS code when used over GE channel is given by $1 - P(A)$.

C. Pimentel and I. F. Blake, "Enumeration of markov chains and burst error statistics for finite state channel models," IEEE Transactions on Vehicular Technology, 1999.

Computing Burst Erasure Probabilities

- Let B be the set of erasures whose span is atmost b in window of length n.
- Let q_i be the probability of erasures where the first erasure appears at index i and the span $\leq b$.

1	…	$i-1$	i	$i+1$	…	$i+b'-1$	$i+b'$	…	n
0	…	0	1	X	…	X	0	…	0

don't care

$$P(B) = P(E_1^n = \underline{0}) + \sum_{i=1}^{n} q_i$$

$$q_i = 1^T \Psi(0)^{n-i-b'+1} S^{b'-1} \Psi(1) \Psi(0)^{i-1} \underline{\pi}$$

where $b' = \min\{b, n-i+1\}$.

- Any cyclic code with parameters $[n, k = n-b]$ has BEP upper bounded by $1 - P(B)$.

G. Haßlinger and O. Hohlfeld, "Analysis of random and burst error codes in 2-state markov channels," in 34th International Conference on Telecommunications and Signal Processing (TSP 2011).

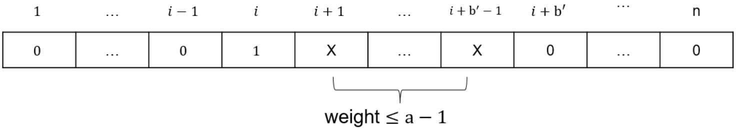
What is New ?

Computing $P(A \cup B)$

- $A \cup B$ is the set of erasure patterns either have weight atmost a or span atmost b in window of length n.

$$P(A \cup B) \quad = \quad P(A) + P(B) - P(A \cap B) \triangleq P_{ws}(n, a, b)$$

- Let q_i be the probability of erasures where the first erasure appears at index i and the span $\leq b$ and weight $\leq a$.



$$\begin{aligned} P(A \cap B) &= P(E_1^n = \underline{0}) + \sum_{i=1}^{n} q_i \\ q_i &= 1^T \Psi(0)^{n-i-b'+1} Q(b'-1, a-1) \Psi(1) \Psi(0)^{i-1} \underline{\pi} \end{aligned}$$

where $b' = \min\{b, n-i+1\}$

Bounding $P(\text{AEP})$

$$P(\text{AEP}) = P(\cap_{i=1}^{n-\tau}(A_i \cup B_i))$$

- $A \cup B$ is the set of erasure patterns that either have weight atmost a or span atmost b in a window $[1 : n]$.

- $A_i \cup B_i$ is the set of erasure patterns that either have weight atmost a or span atmost b in a window $[i : \tau + i]$.

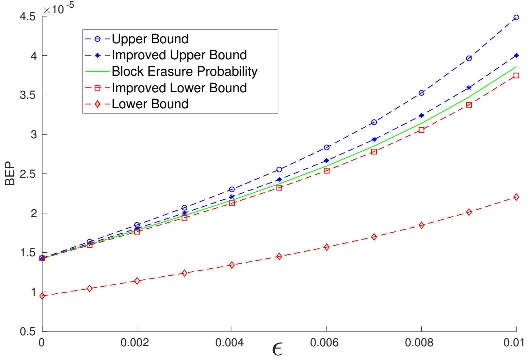
$$\begin{array}{rcl} (A \cup B) \subseteq & \text{AEP} & \subseteq (A_1 \cup B_1) \\ P_{ws}(n, a, b) \leq & P(\text{AEP}) & \leq P_{ws}(\tau + 1, a, b) \end{array}$$

Bounds on BEP of streaming code

- Improved the bounds by coming up with tractable sets L, U such that:

$$\begin{aligned} (A \cup B) \subseteq L \subseteq \quad \text{AEP} \quad \subseteq U \subseteq (A_1 \cup B_1) \\ 1 - P(U) \leq \quad \text{BEP} \quad \leq 1 - P(L) \end{aligned}$$

$(a = 3, b = 6, \tau = 10)$ streaming code

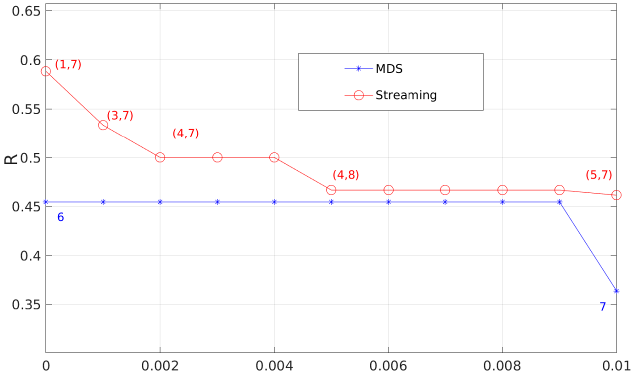


$\text{GE}(\alpha = 10^{-4}, \beta = 0.5, \epsilon_0 = \epsilon, \epsilon_1 = 1)$

Choosing a, b Using BEP Upper Bound

- (a, b) is picked to give best rate while meeting BEP$\leq P_e$ requirement for $(n = \tau + 1 + b - a, k = n - b)$ streaming code.

- For $[\tau + 1, \tau + 1 - a]$ MDS codes minimal value of a is picked to satisfy BEP requirement.



GE($\alpha = 10^{-4}, \beta = 0.5, \epsilon_0 = \epsilon, \epsilon_1 = 1$), $\tau = 10$ and $P_e = 10^{-5}$

ToC

An Experimental Attempt at Channel Adaptation

Parameters of Interest for Any Streaming Application

- End to End Delay (Δ)
  - V2X requires $\Delta \leq 100$ms and,
  - Telesurgery Camera Flow requires $\Delta \leq 150$ms

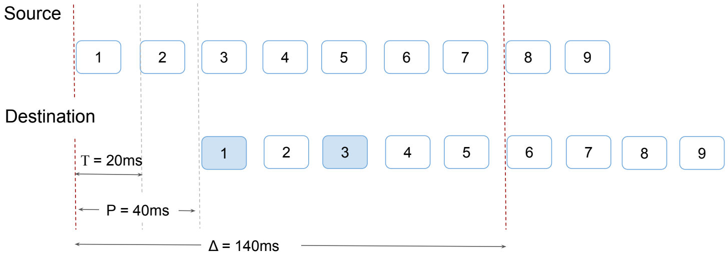
- Reliability (P_e)
  - Packet Erasure Probability (PEP) $\leq P_e$

    OR

  - Block Erasure Probability (BEP) $\leq P_e$.

- ETSI TS 122 185 V14.3.0. LTE;Service requirements for V2X services. 2017.
- 5G Americas. 5G Services Innovation. 2019.

Breakup of E2E Delay



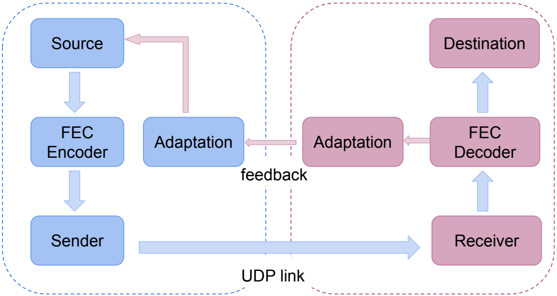
T: Inter packet delay, P: Propagation Delay

- Pick τ such that:

$$P + \mathrm{T} * (\tau + 1) \quad \leq \quad \Delta$$

- For the example above $\tau = 4$ should suffice.

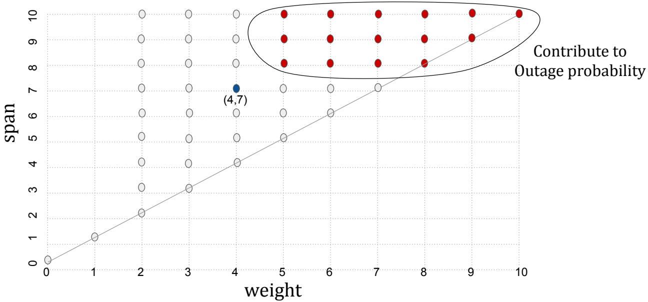
System Architecture for Rate Adaptation



- Source uses VP8 encoder to compress video frames.
- Compressed frame is divided into equal sized packets and sent over UDP link
- We introduce erasures in the UDP link using GE channel model
- $\{a, b\}$ parameters obtained by adaptation algorithm that has access to packet erasure patterns.
- FEC encoder uses simple streaming code family implemented using Jerasure library.

Outage Based Rate Adaptation

- M past packets are used to estimate (a, b) parameters.
- Estimation of (a, b) parameters takes place once every L packets.

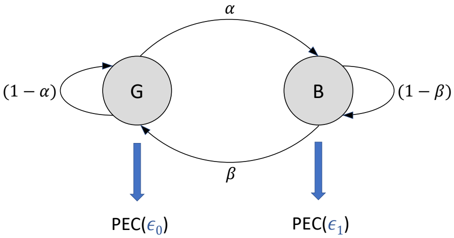


$(a = 4, b = 7, \tau = 9)$ code can recover from all the erasures other than the ones show in red

- empirical probabilities of (span, weight) pairs are maintained.
- allows for a small nonzero probability (outage) of uncorrectable erasure patterns

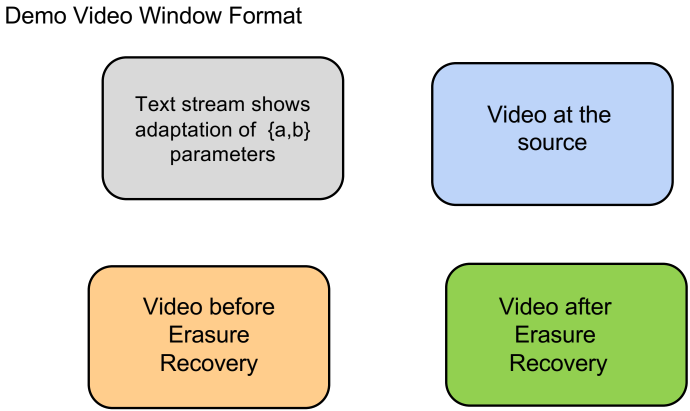
Video Demo Setting

- The Channels:
  - C0: perfect channel (no erasures)
  - C1: GE ($\alpha = 0.01, \beta = 0.5, \epsilon_0 = 0.001, \epsilon_1 = 1$)



- τ is set to 9
- Outage adaptation parameters: $M = 10^5, \quad L = 10^3, \quad P_{out} = 10^{-3}$.
- Video1: We set ($a = 0, b = 0$) at the start of experiment and move from C0 to C1 (we see adaptation taking place)
- Video2: After the adaptation has converged

Description of Four Windows Appearing in The Demo



References

1. E. Martinian and C. W. Sundberg, "Burst erasure correction codes with low decoding delay," *IEEE Trans. Inf. Theory*, 2004.
2. E. Martinian and M. Trott, "Delay-Optimal Burst Erasure Code Construction," *ISIT* 2007.
3. A. Badr, P. Patil, A. Khisti, W. Tan, and J. G. Apostolopoulos, "Layered Constructions for Low-Delay Streaming Codes," *IEEE Trans. Inf. Theory*, 2017.
4. N. Krishnan and P. V. Kumar, "Rate-Optimal Streaming Codes for Channels with Burst and Isolated Erasures," *ISIT* 2018.
5. S. L. Fong, A. Khisti, B. Li, W. Tan, X. Zhu, and J. G. Apostolopoulos, "Optimal streaming codes for channels with burst and arbitrary erasures," *IEEE Trans. Inf. Theory*, 2019.
6. N. Krishnan, D. Shukla and P. V. Kumar, "A Quadratic Field-Size Rate-Optimal Streaming Code for Channels with Burst and Random Erasures," *ISIT* 2019.
   (finalist for IEEE Jack Wolf Student Paper Award).

References

7. N. Krishnan, D. Shukla and P. V. Kumar, "Low Field-size, Rate-Optimal Streaming Codes for Channels with Burst and Random Erasures," *IEEE Trans. Inf. Theory*, 2020.

8. E. Domanovitz, S. L. Fong, and A. Khisti, "An explicit rate-optimal streaming code for channels with burst and arbitrary erasures," *ITW* 2019.

9. N. Krishnan, V. Ramkumar, M. Vajha and P. V. Kumar, "Simple Streaming Codes for Reliable, Low-Latency Communication," *IEEE Comm. Letters*, 2020.

10. V. Ramkumar, M. Vajha, M. N. Krishnan, and P. V. Kumar, "Staggered Diagonal Embedding Based Linear Field Size Streaming Codes," *ISIT* 2020.

11. M. Vajha, V. Ramkumar, M. N. Krishnan, and P. V. Kumar, "Explicit Rate-Optimal Streaming Codes with Smaller Field Size," *ISIT* 2021.

12. V. Ramkumar, M. Vajha, and P. V. Kumar, "Generalized Simple Streaming Codes from MDS Codes," *ISIT* 2021.

References

13. A. Badr, A. Khisti, and E. Martinian, "Diversity Embedded Streaming Erasure Codes (de-sco): Constructions and Optimality," *IEEE J. Sel. Areas Commun.*, 2011.

14. A. Badr, D. Lui, and A. Khisti, "Streaming Codes for Multicast over Burst Erasure Channels," *IEEE Trans. Inf. Theory*, 2015.

15. M. Rudow and K. V. Rashmi, "Streaming Codes for Variable-Size Arrivals," *Allerton* 2018.

16. S. L. Fong, A. Khisti, B. Li, W. Tan, X. Zhu, J. G. Apostolopoulos, "Optimal Streaming Erasure Codes Over the Three-Node Relay Network," *IEEE Trans. Inf. Theory*, 2020.

17. M. Vajha, V. Ramkumar, M. Jhamtani, P. V. Kumar, "On Sliding Window Approximation of Gilbert-Elliott Channel for Delay Constrained Setting," *arXiv* 2020.

Thanks!