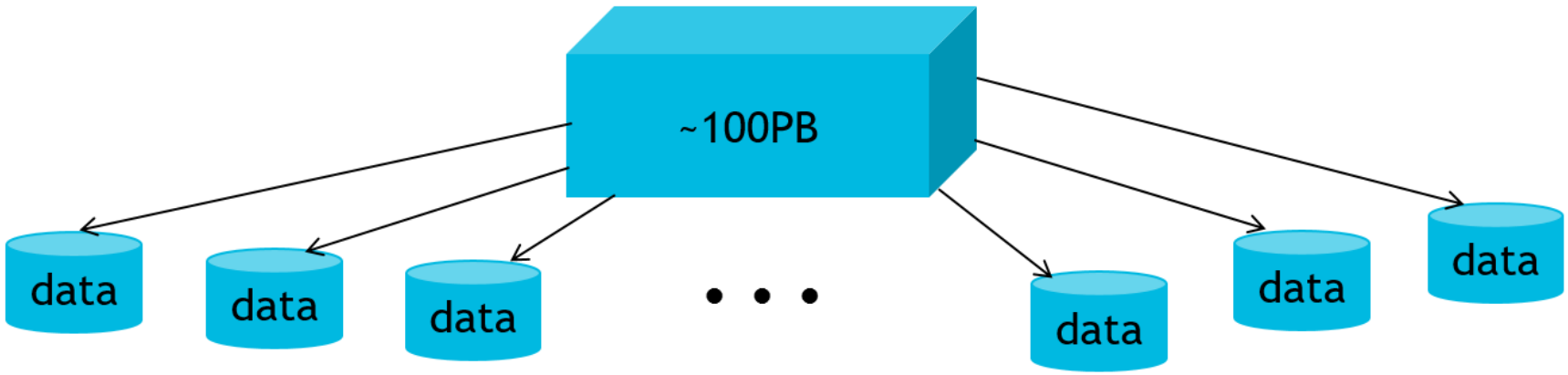
MDS codes with optimal regeneration

Alexander Barg

University of Maryland, College Park

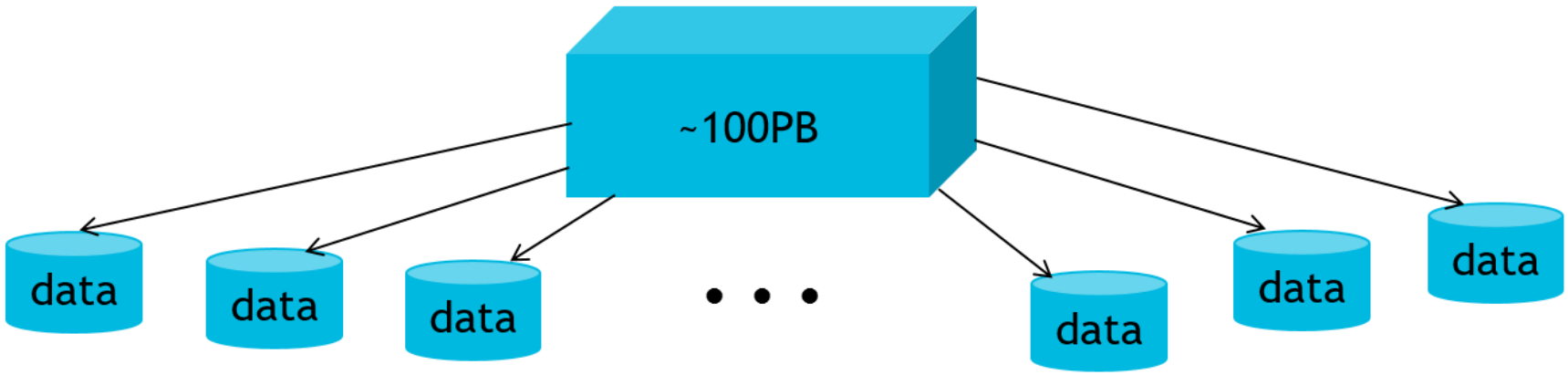
June 21, 2017

Motivation: Distributed Storage Systems (DSS)



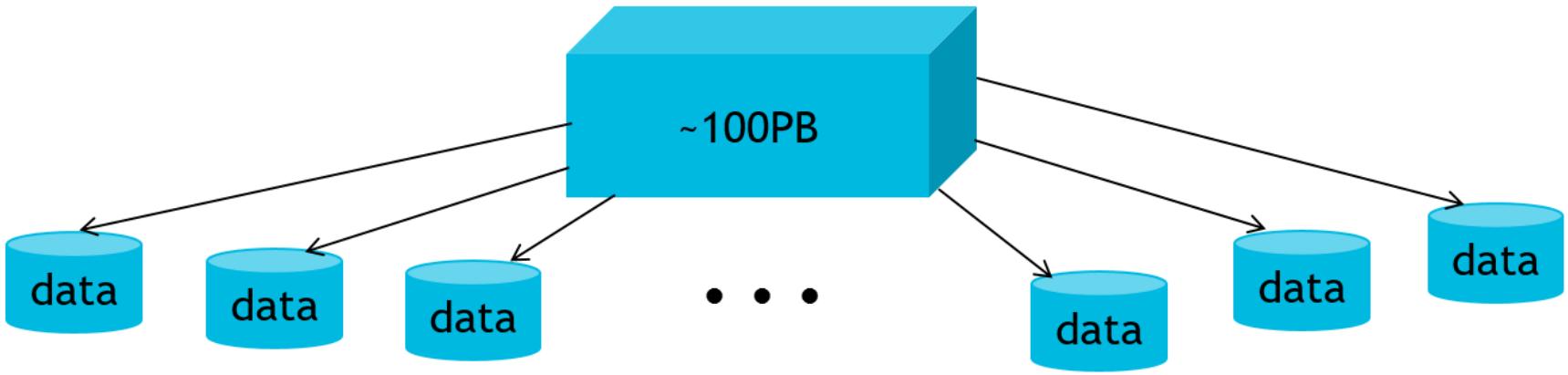
- DSS spread data across thousands of storage nodes

Motivation: Distributed Storage Systems (DSS)



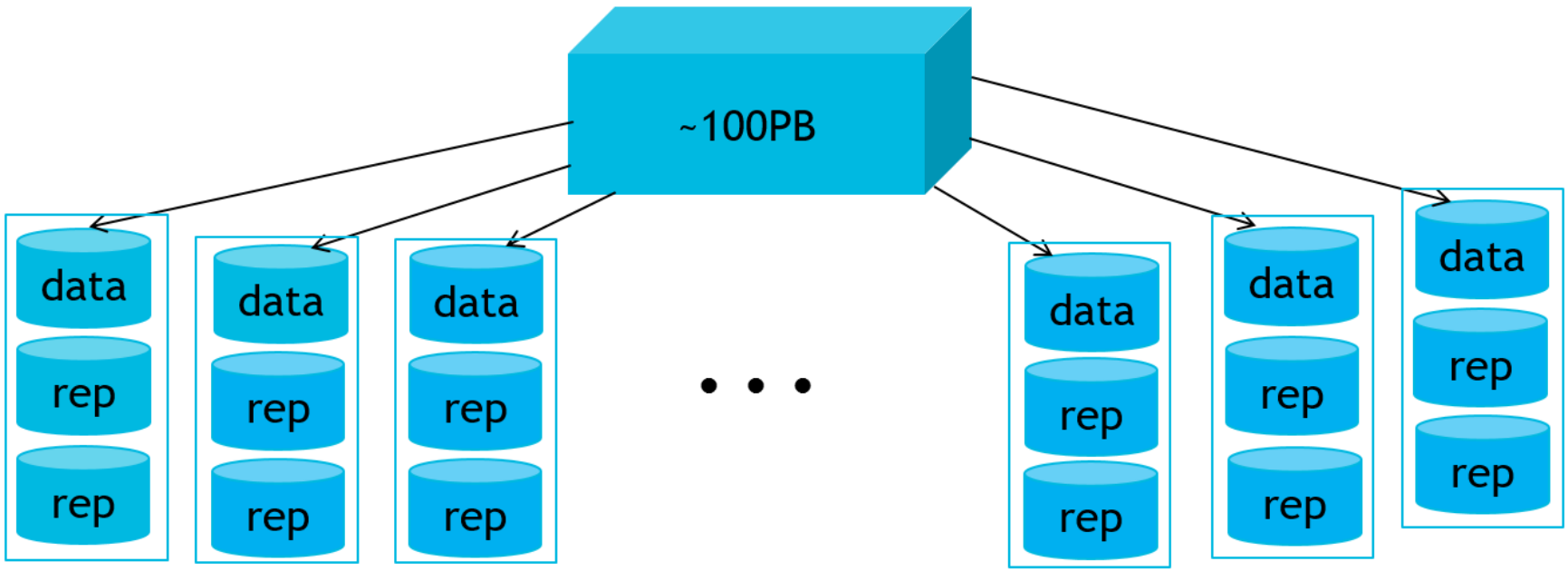
- DSS spread data across thousands of storage nodes
- Individual storage nodes fail frequently

Motivation: Distributed Storage Systems (DSS)



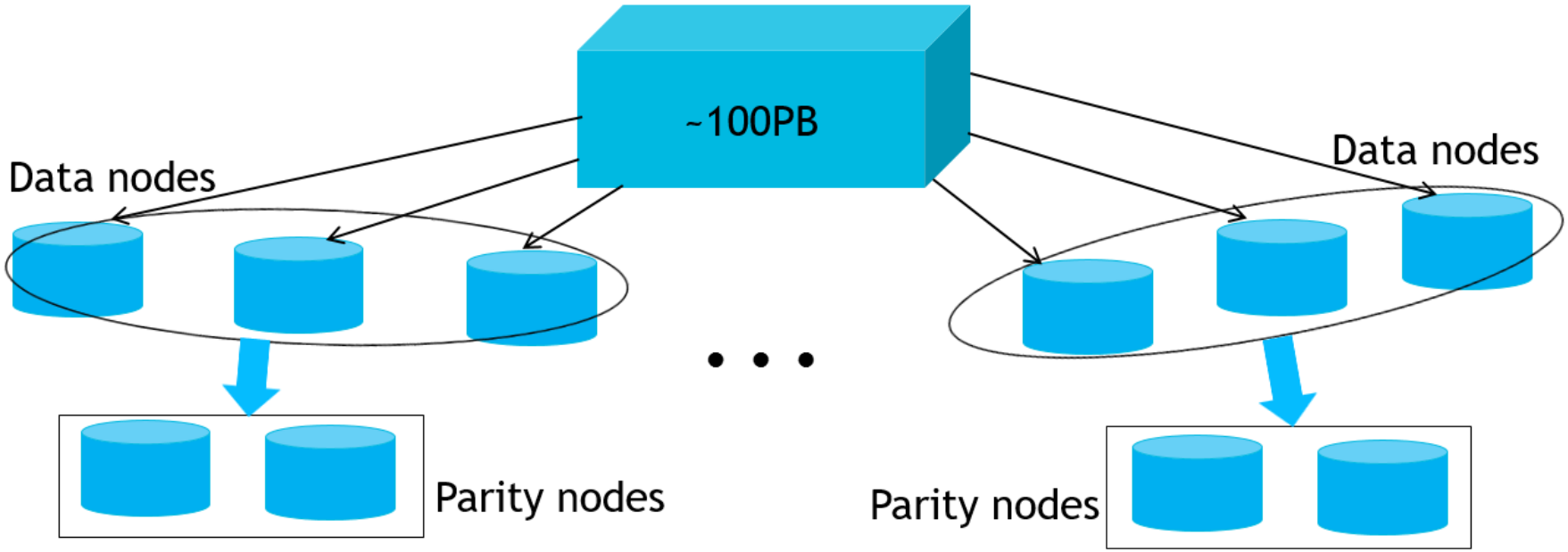
- DSS spread data across thousands of storage nodes
- Individual storage nodes fail frequently
- To protect the data we rely on erasure codes

Replication: large storage overhead



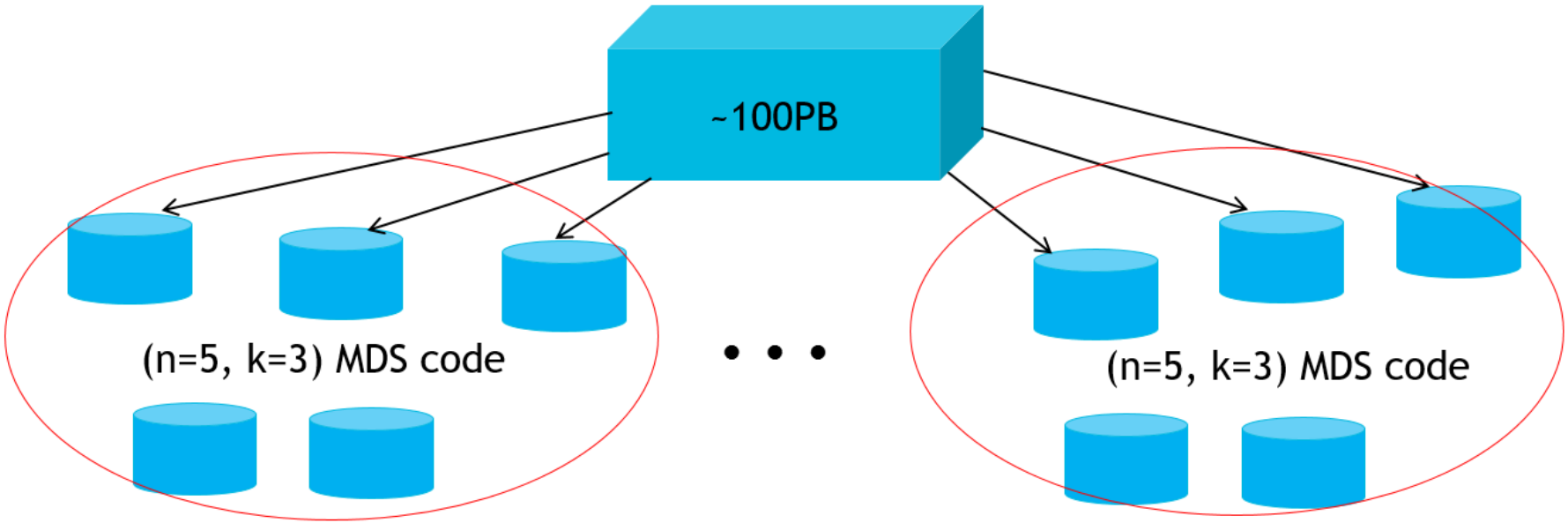
Can tolerate any 2 node failures
Storage overhead $= \quad 3\times$

MDS codes: Optimal storage efficiency



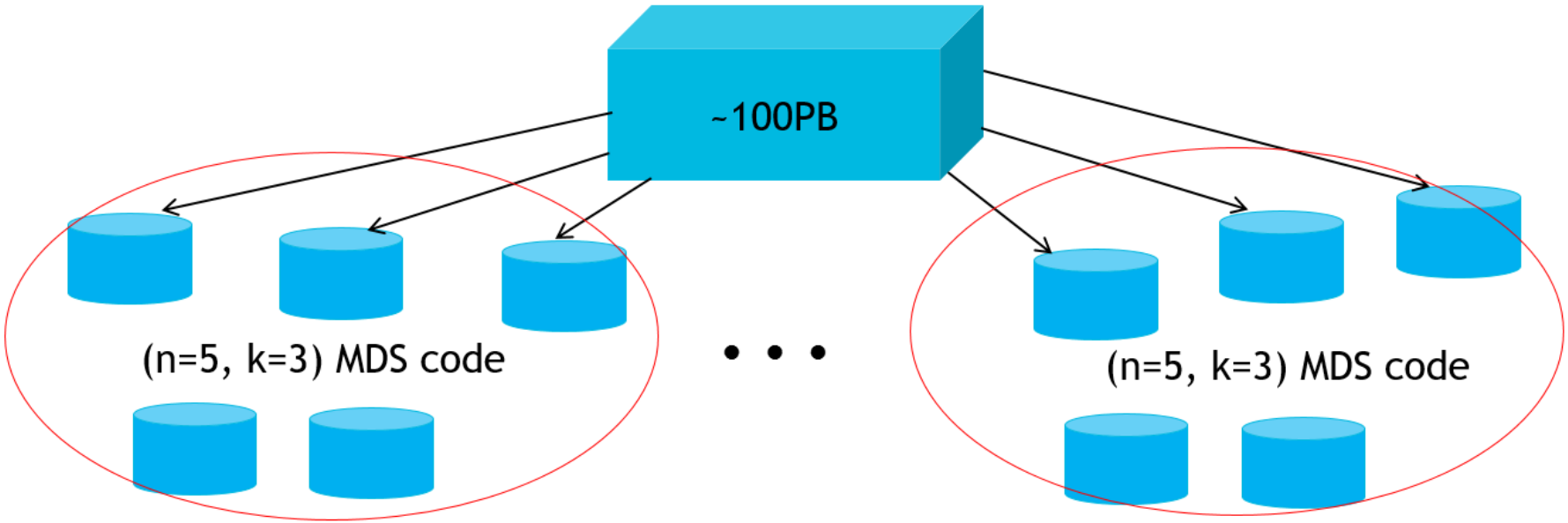
Add 2 parity nodes to every 3 data nodes
Form an $(n = 5, k = 3)$ MDS code

MDS codes: Optimal storage efficiency



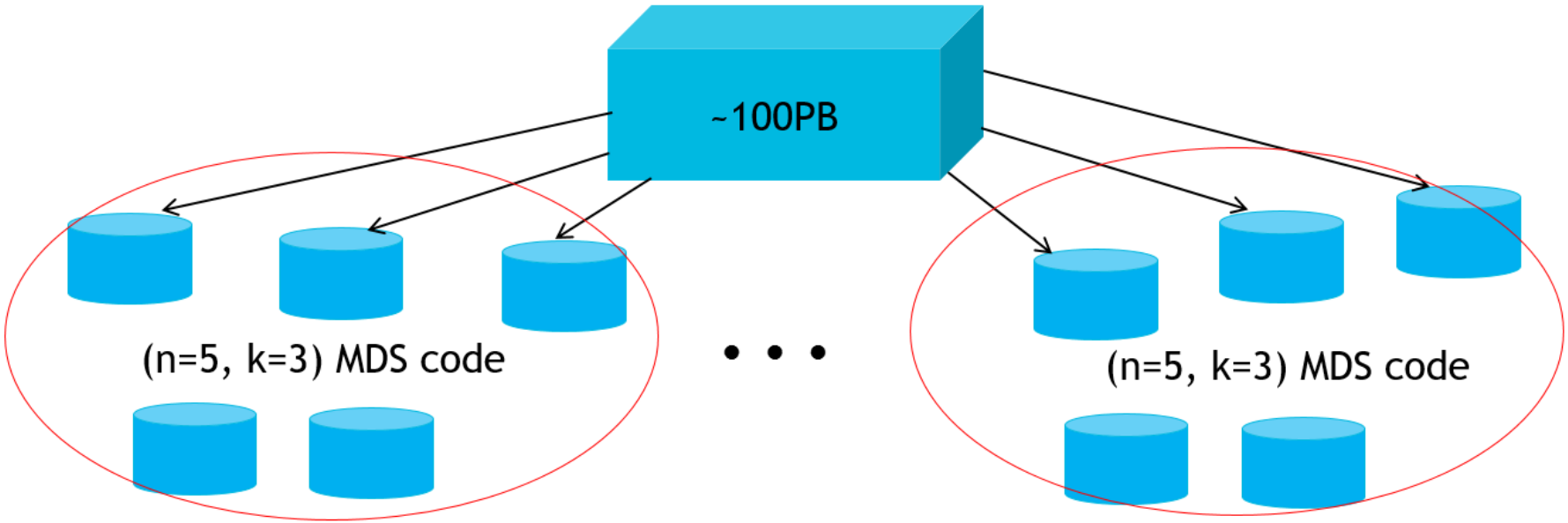
Can tolerate any 2 node failures
Storage overhead $= \quad 1.67\times$

MDS codes: Optimal storage efficiency



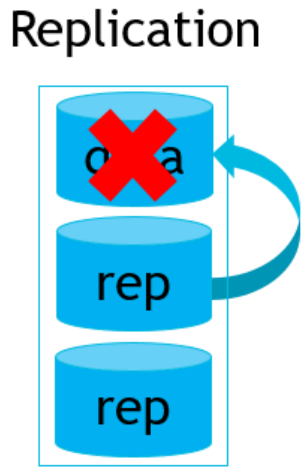
Block-based system model: Data blocks are encoded independently

MDS codes: Optimal storage efficiency

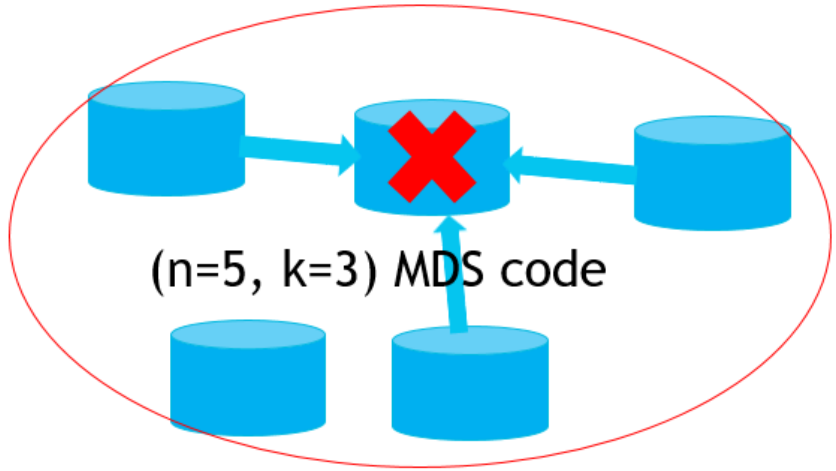


Each node stores l symbols of the block

Network flow incurred by data regeneration



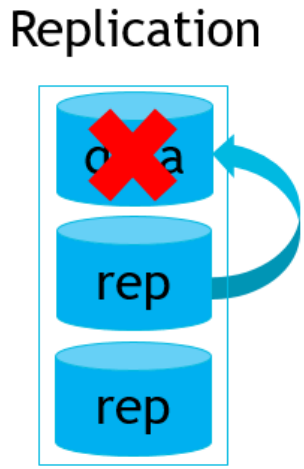
Network flow = node size



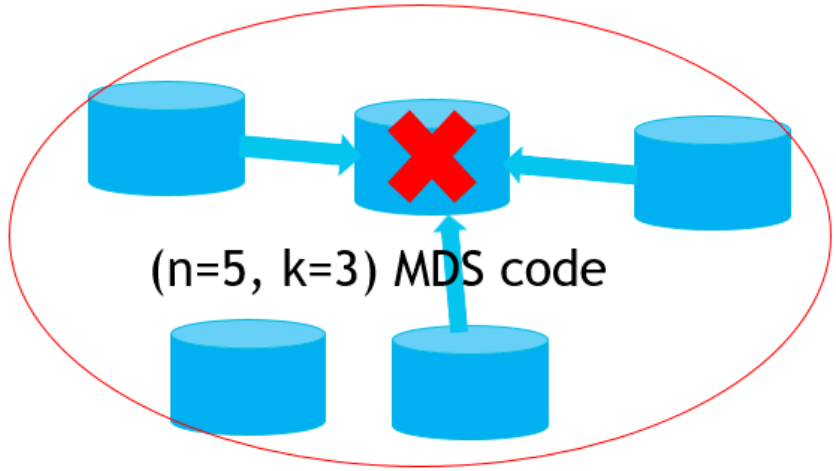
Network flow = 3 x node size

MDS code uses much more network bandwidth during data regeneration

Network flow incurred by data regeneration



Network flow = node size



Network flow = 3 x node size

MDS code uses much more network bandwidth during data regeneration

Goal: To find MDS codes using less bandwidth during data regeneration

Reed-Solomon codes

$F = \mathbb{F}_q$ the finite field of q elements

Example: $q = 8$, $\alpha^3 = \alpha + 1$

$\mathbb{F}_8 = \{0, 1, \alpha, \alpha^2, \alpha^3 = \alpha + 1, \alpha^4 = \alpha^2 + \alpha, \alpha^5 = \alpha^2 + \alpha + 1, \alpha^6 = \alpha^2 + 1\}$

The elements of $\mathbb{F}_8$ can also be written as binary vectors
$(000), (001), (010), (101), (110), (111), (101)$

Encoding example: Let $q = 8$, $(n, k) = (7, 3)$

Suppose that data symbols are $1, \alpha, \alpha$

To encode, form a polynomial $f(x) = 1 + \alpha x + \alpha x^2$ and compute the values of $f(x)$ at the points $1, \alpha, \alpha^2, \ldots, \alpha^6$

x	1	α	α^2	α^3	α^4	α^5	α^6
$f(x)$	1	α^4	α^6	α^4	α	α	α^6

This encodes $k = 3$ data symbols $(1, \alpha, \alpha)$ into $n = 7$ symbols of the codeword

Reed-Solomon codes

Reed-Solomon code $\mathcal{C}(A)$ over F encodes k data symbols into n *code symbols*

Let $f(x) = a_0 + a_1 x + \cdots + a_{k-1} x^{k-1}$ be the data (represented by polynomials) and encode as follows:

$$\begin{aligned} \mathcal{C} : F^k &\to F_q^n \\ f(x) &\mapsto (f(P_1), f(P_2), \ldots, f(P_n)) \end{aligned}$$

where $P_1, \ldots, P_n$ are some n elements of F

Reed-Solomon codes

Reed-Solomon code $\mathcal{C}(A)$ over F encodes k data symbols into n *code symbols*
Let $f(x) = a_0 + a_1 x + \cdots + a_{k-1} x^{k-1}$ be the data (represented by polynomials) and encode as follows:

$$\begin{aligned} \mathcal{C} : F^k &\to F_q^n \\ f(x) &\mapsto (f(P_1), f(P_2), \ldots, f(P_n)) \end{aligned}$$

where $P_1, \ldots, P_n$ are some n elements of F

In other words, $(c_1, \ldots, c_n)$ is an RS codeword if it satisfies the following ***parity-check equations:***

$$c_1 + c_2 \alpha^i + c_3 \alpha^{2i} \cdots + c_n \alpha^{(n-1)i} = 0, \quad i = 1, \ldots, n-k$$

Use of RS codes for node repair

Use of RS codes for node repair

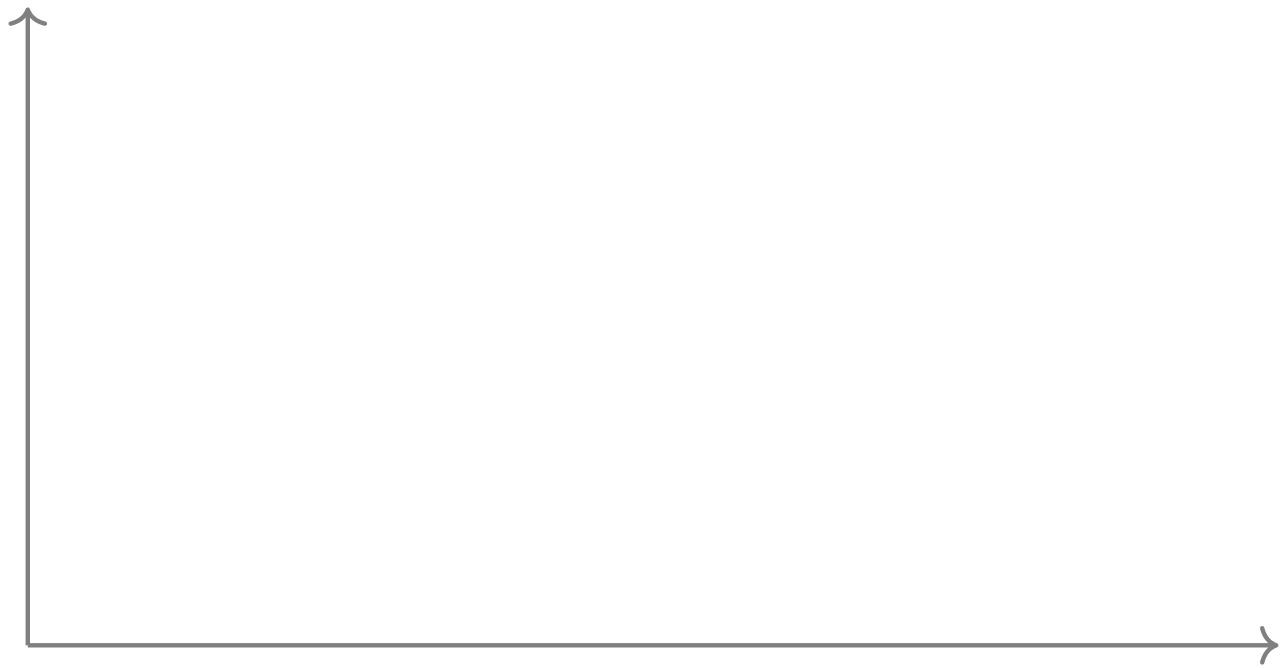
(a_0, a_1, a_2, a_3)

Use of RS codes for node repair

$$(a_0, a_1, a_2, a_3) \rightarrow f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$

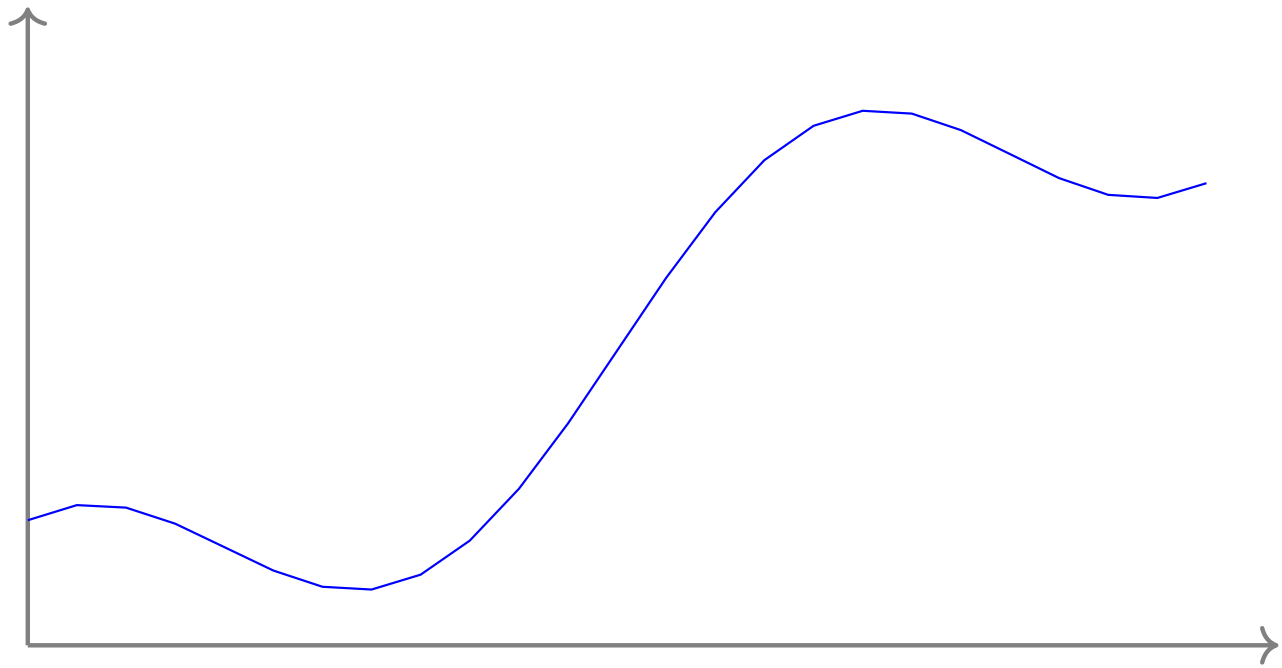
Use of RS codes for node repair

$$(a_0, a_1, a_2, a_3) \rightarrow f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$



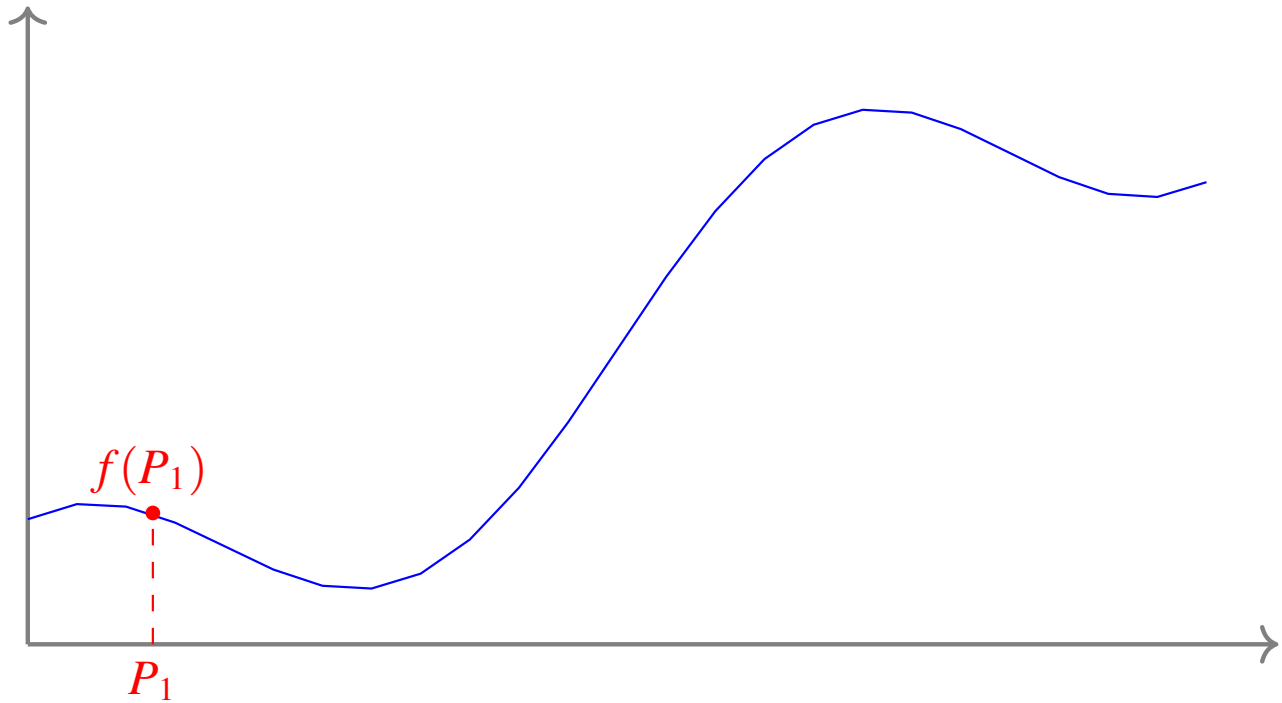
Use of RS codes for node repair

$$(a_0, a_1, a_2, a_3) \rightarrow f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$



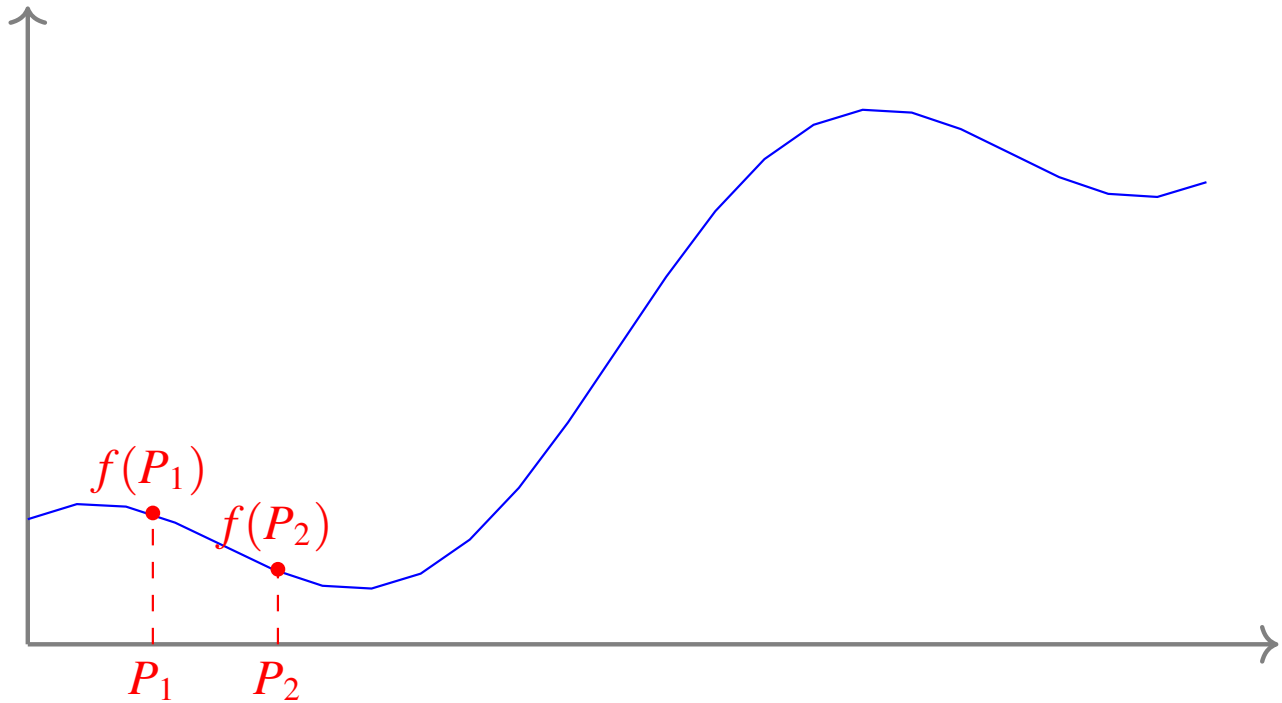
Use of RS codes for node repair

$$(a_0, a_1, a_2, a_3) \rightarrow f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$



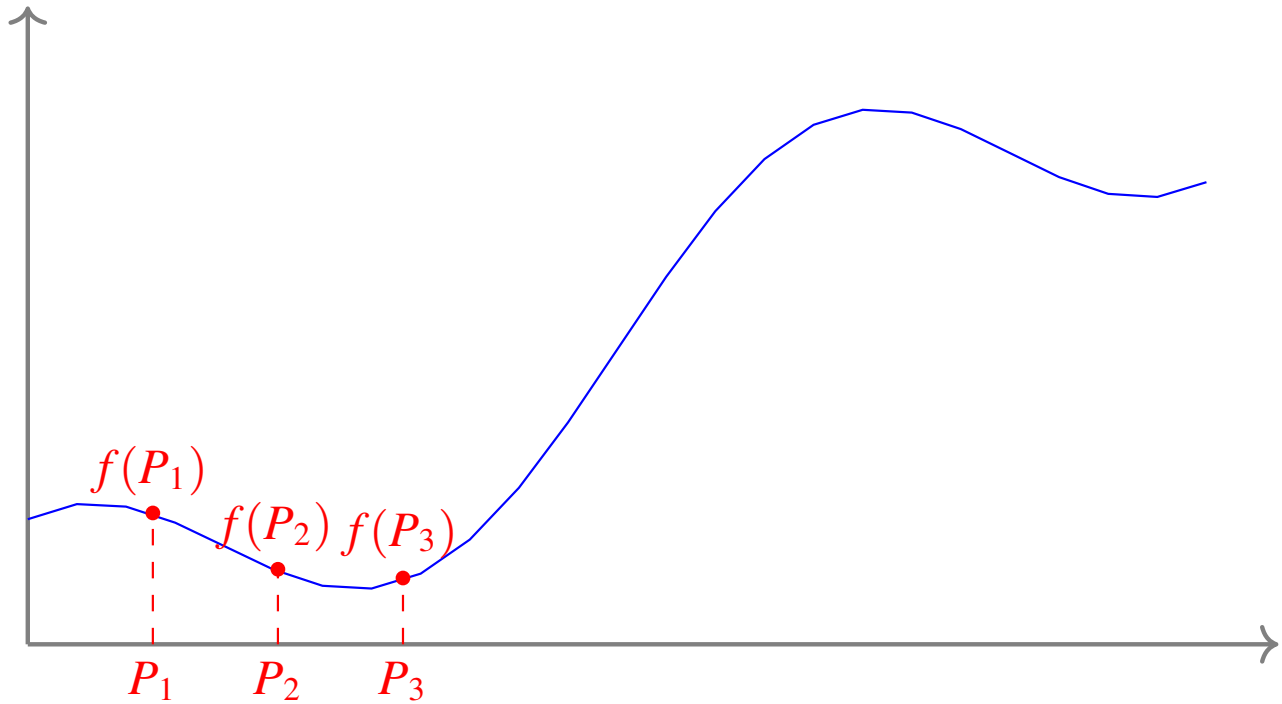
Use of RS codes for node repair

$$(a_0, a_1, a_2, a_3) \rightarrow f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$



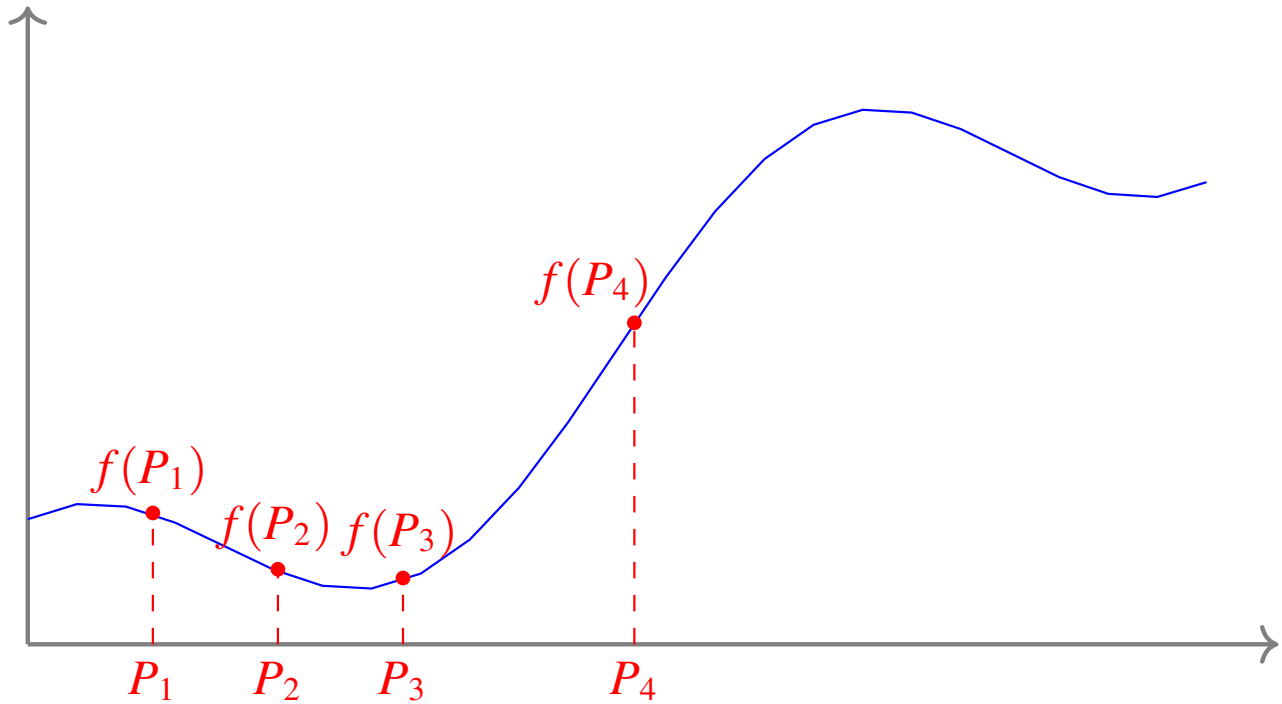
Use of RS codes for node repair

$$(a_0, a_1, a_2, a_3) \rightarrow f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$



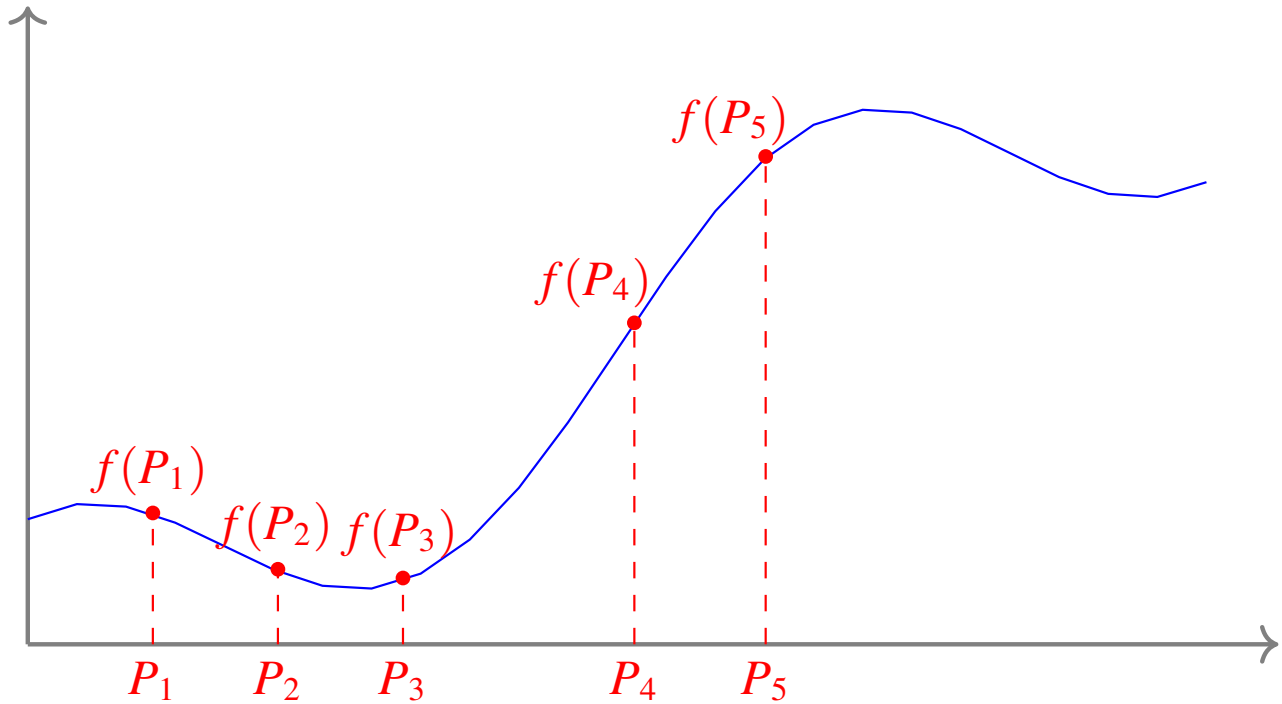
Use of RS codes for node repair

$$(a_0, a_1, a_2, a_3) \rightarrow f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$



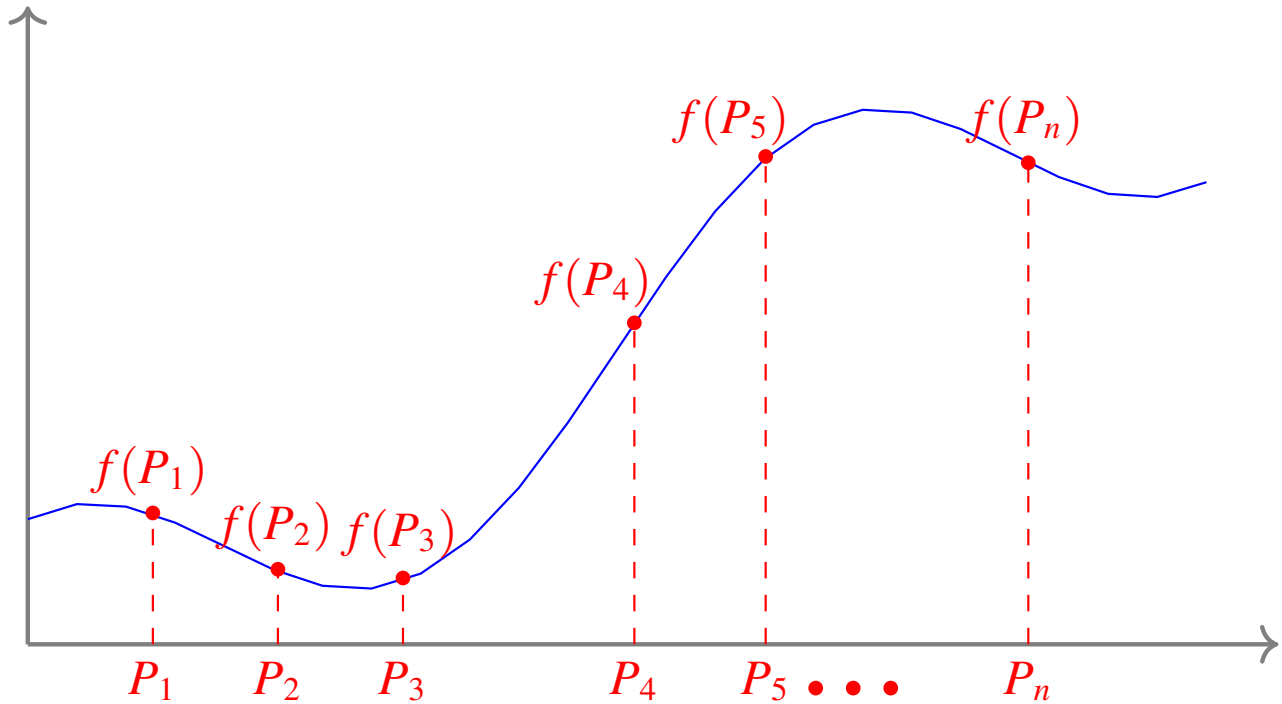
Use of RS codes for node repair

$$(a_0, a_1, a_2, a_3) \rightarrow f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$



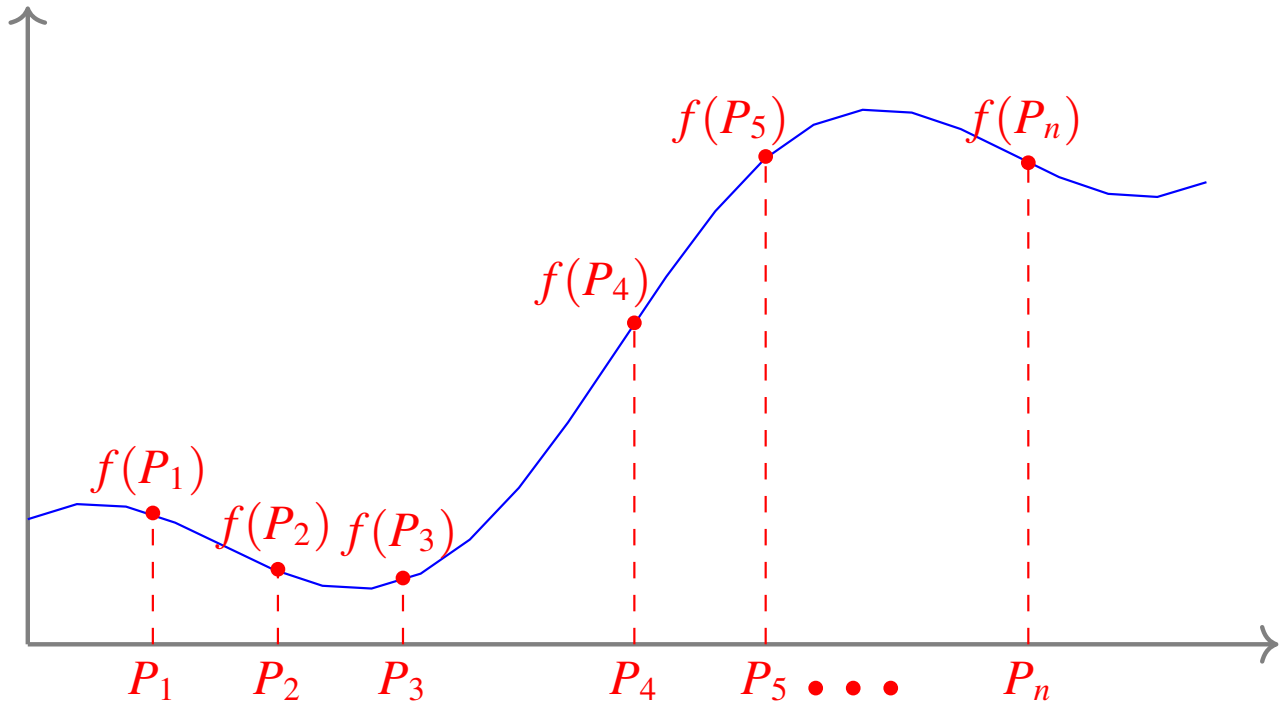
Use of RS codes for node repair

$$(a_0, a_1, a_2, a_3) \rightarrow f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3$$



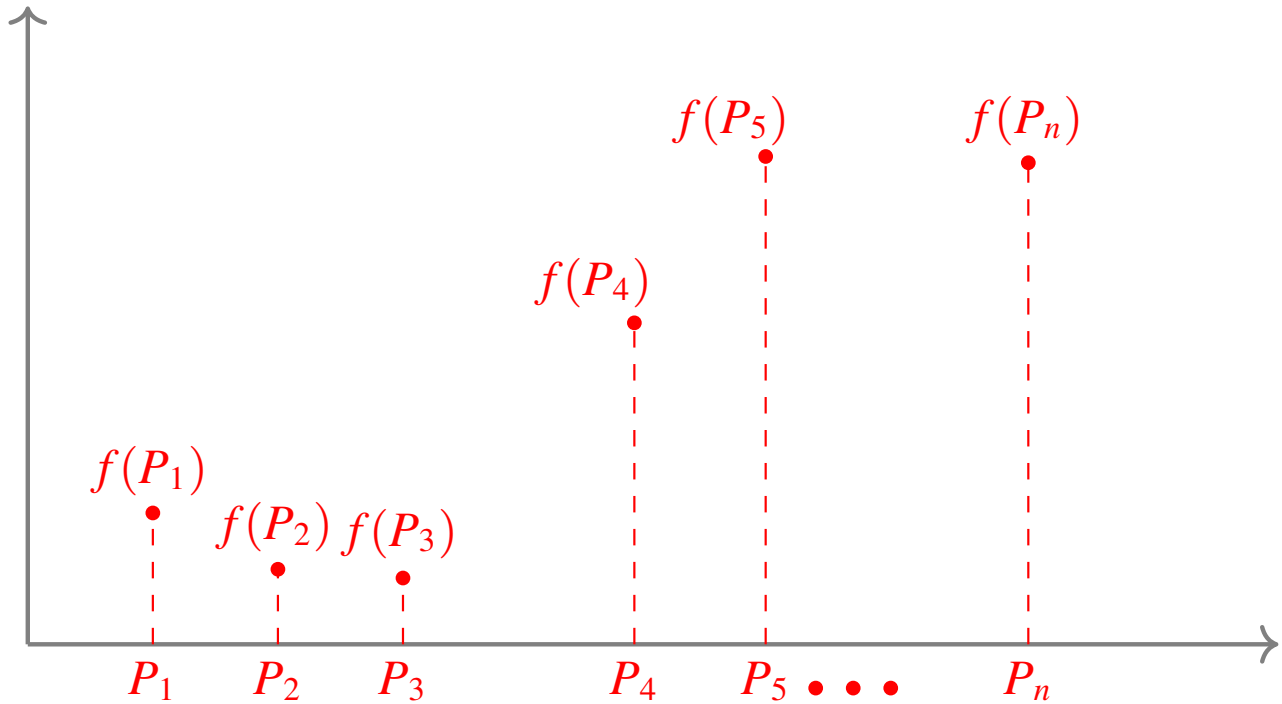
Use of RS codes for node repair

$$(a_0, a_1, a_2, a_3) \rightarrow f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \rightarrow f(x) = (f(P_1), f(P_2), \ldots, f(P_n))$$



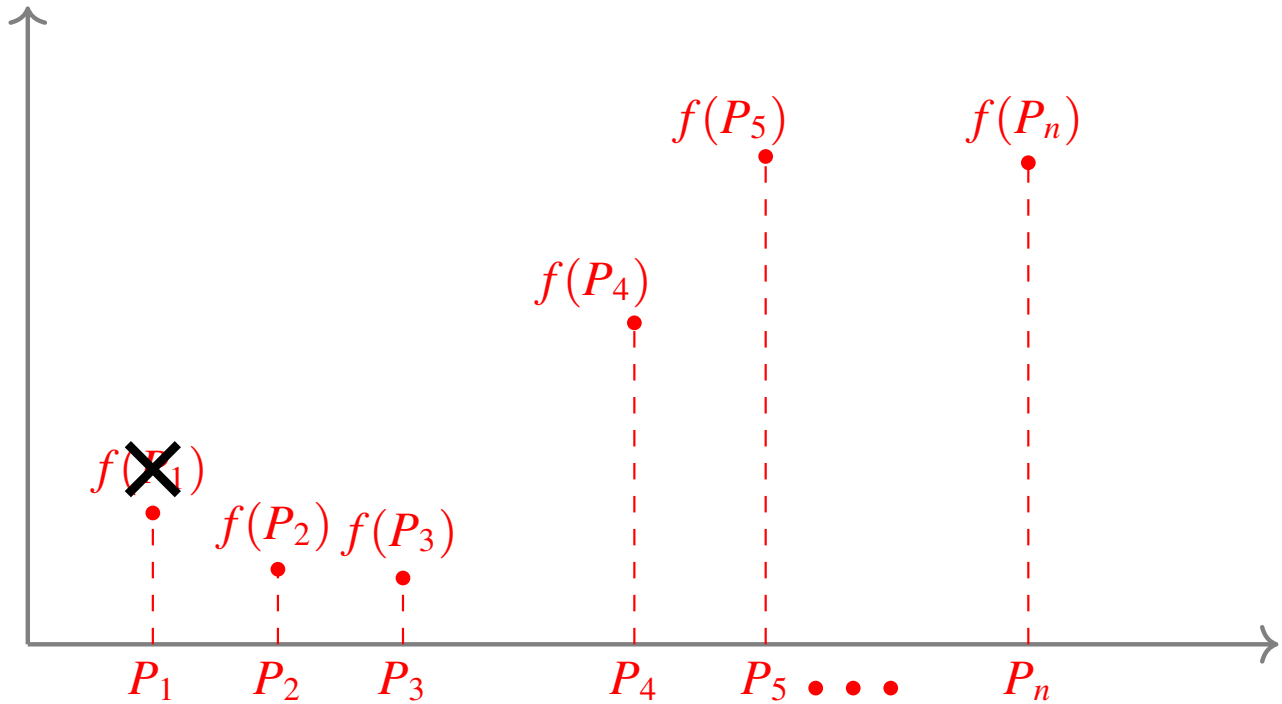
Use of RS codes for node repair

$$(a_0, a_1, a_2, a_3) \rightarrow f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \rightarrow f(x) = (f(P_1), f(P_2), \ldots, f(P_n))$$



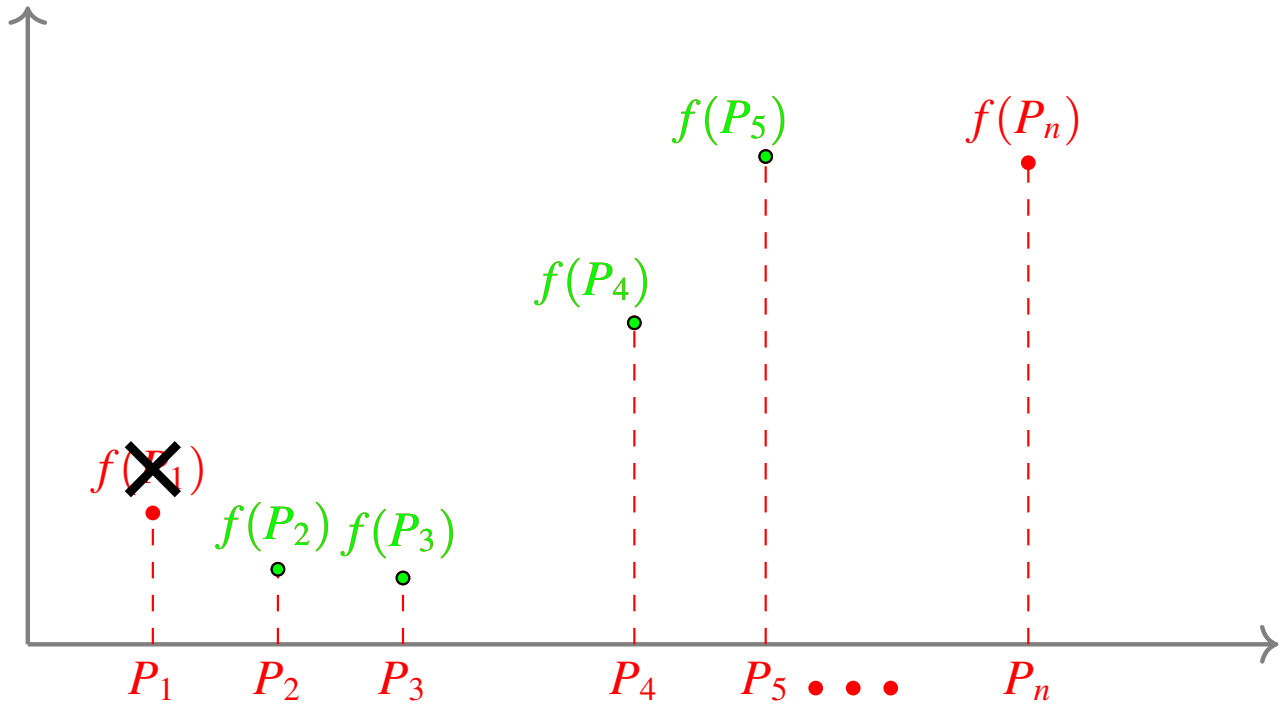
Use of RS codes for node repair

$$(a_0, a_1, a_2, a_3) \rightarrow f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \rightarrow f(x) = (f(P_1), f(P_2), \ldots, f(P_n))$$



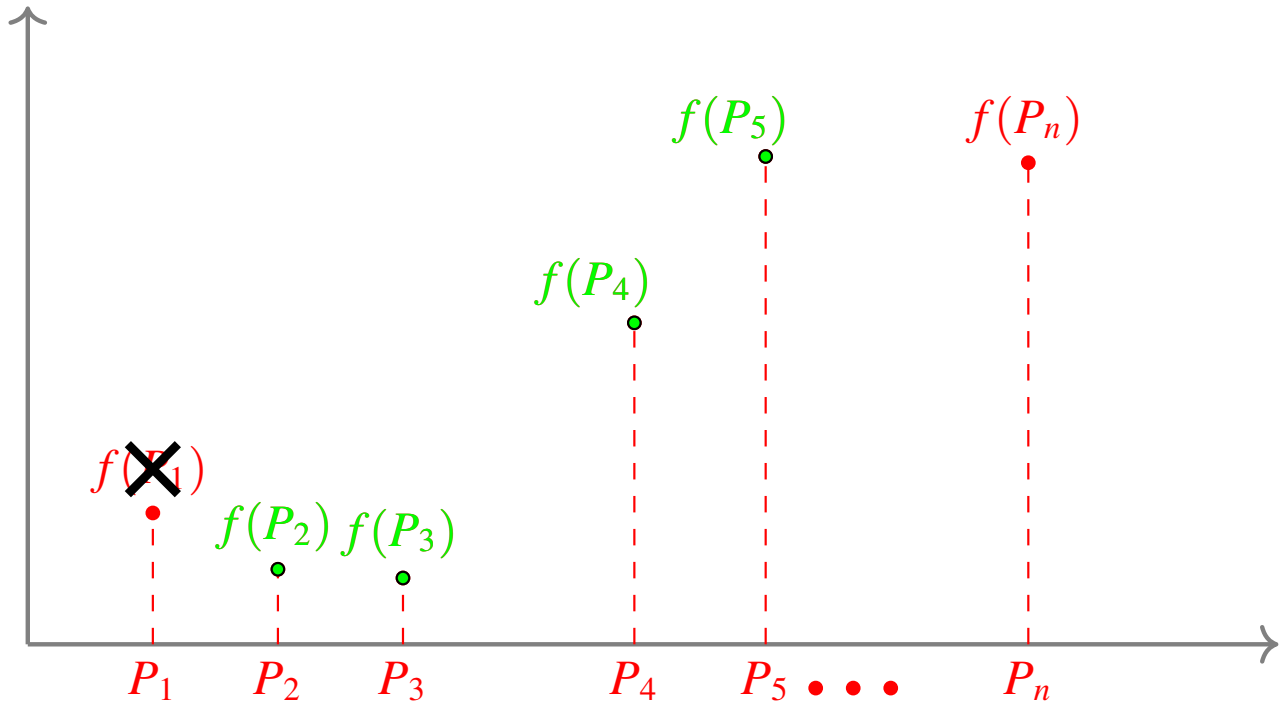
Use of RS codes for node repair

$$(a_0, a_1, a_2, a_3) \rightarrow f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \rightarrow f(x) = (f(P_1), f(P_2), \ldots, f(P_n))$$



Use of RS codes for node repair

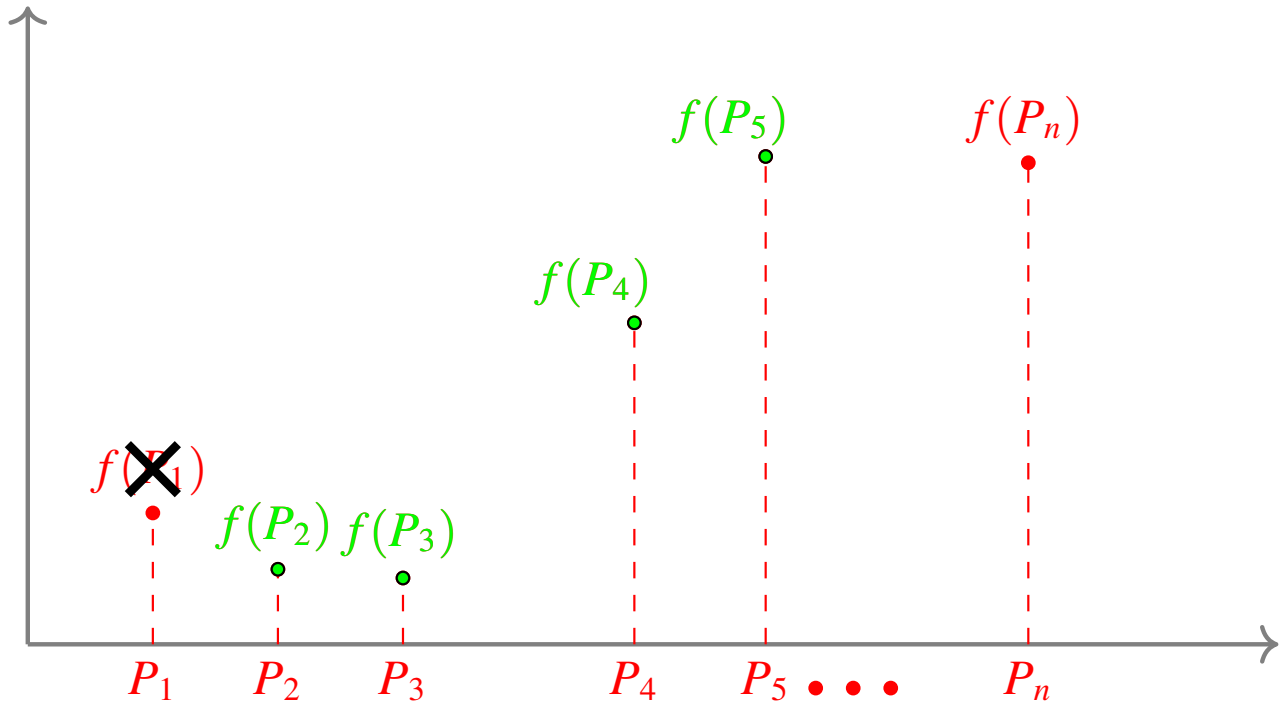
$$(a_0, a_1, a_2, a_3) \rightarrow f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \rightarrow f(x) = (f(P_1), f(P_2), \ldots, f(P_n))$$



Use of RS codes for node repair

$$(a_0, a_1, a_2, a_3) \rightarrow f(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 \rightarrow f(x) = (f(P_1), f(P_2), \ldots, f(P_n))$$

We need to download the contents of $k = 4$ nodes to repair a failed node



MDS array codes for storage

Main idea for improvement: Sub-packetize the encoding (consider vector codes over a finite field F)

[1] We use "coordinates" and "nodes" interchangeably

MDS array codes for storage

Main idea for improvement: Sub-packetize the encoding (consider vector codes over a finite field F)

Each coordinate[1] of the codeword $(C_1, C_2, \ldots, C_n) \in F^n$ is an l-dimensional vector over F, so the codeword can be viewed as an $l \times n$ array over F

- (n, k, l) MDS array code:

[1] We use "coordinates" and "nodes" interchangeably

MDS array codes for storage

Main idea for improvement: Sub-packetize the encoding (consider vector codes over a finite field F)

Each coordinate[1] of the codeword $(C_1, C_2, \ldots, C_n) \in F^n$ is an l-dimensional vector over F, so the codeword can be viewed as an $l \times n$ array over F

- (n, k, l) MDS array code:
  - code length n

[1] We use "coordinates" and "nodes" interchangeably

MDS array codes for storage

Main idea for improvement: Sub-packetize the encoding (consider vector codes over a finite field F)

Each coordinate[1] of the codeword $(C_1, C_2, \ldots, C_n) \in F^n$ is an l-dimensional vector over F, so the codeword can be viewed as an $l \times n$ array over F

- (n, k, l) MDS array code:
  - code length n
  - k data nodes

[1] We use "coordinates" and "nodes" interchangeably

MDS array codes for storage

Main idea for improvement: Sub-packetize the encoding (consider vector codes over a finite field F)

Each coordinate[1] of the codeword $(C_1, C_2, \ldots, C_n) \in F^n$ is an l-dimensional vector over F, so the codeword can be viewed as an $l \times n$ array over F

- (n, k, l) MDS array code:
  - code length n
  - k data nodes
  - $r = n - k$ parity nodes

[1] We use "coordinates" and "nodes" interchangeably

MDS array codes for storage

Main idea for improvement: Sub-packetize the encoding (consider vector codes over a finite field F)

Each coordinate[1] of the codeword $(C_1, C_2, \ldots, C_n) \in F^n$ is an l-dimensional vector over F, so the codeword can be viewed as an $l \times n$ array over F

- (n, k, l) MDS array code:
  - code length n
  - k data nodes
  - $r = n - k$ parity nodes
  - each codeword coordinate is a vector of dimension l over $\mathbb{F}$

[1] We use "coordinates" and "nodes" interchangeably

MDS array codes for storage

Main idea for improvement: Sub-packetize the encoding (consider vector codes over a finite field F)

Each coordinate[1] of the codeword $(C_1, C_2, \ldots, C_n) \in F^n$ is an l-dimensional vector over F, so the codeword can be viewed as an $l \times n$ array over F

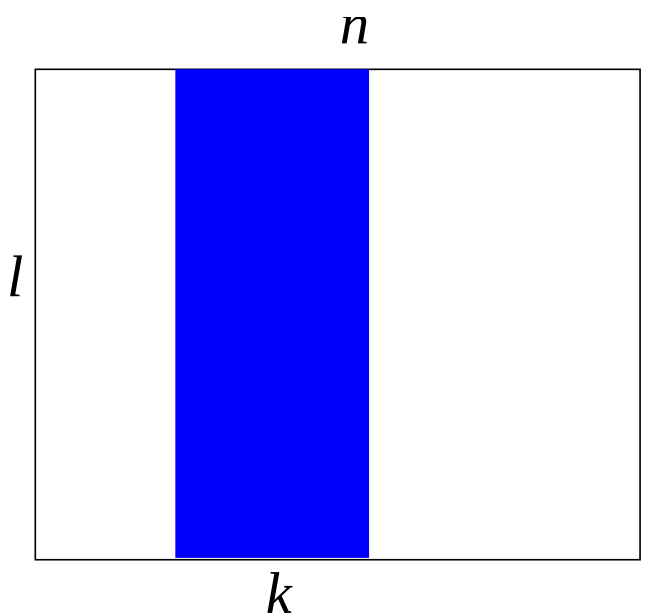
- (n, k, l) MDS array code:
  - code length n
  - k data nodes
  - $r = n - k$ parity nodes
  - each codeword coordinate is a vector of dimension l over $\mathbb{F}$

  - MDS property: Contents of any r nodes can be determined by the other k nodes.

[1] We use "coordinates" and "nodes" interchangeably

MDS array codes

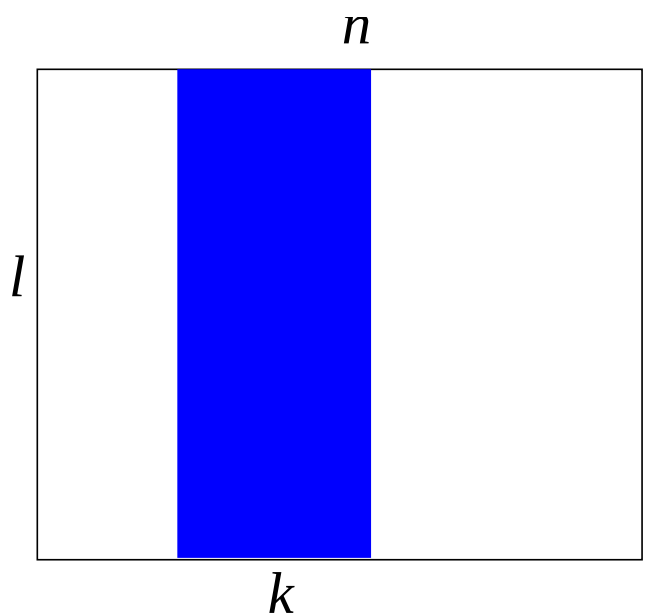
Consider an $\ell \times n$ array code $\mathcal{C}$ ('vector code') over the field F



n - code length
k - code dimension, i.e., $|\mathcal{C}| = q^{k\ell}$
ℓ - length of column, or 'sub-packetization'

MDS array codes

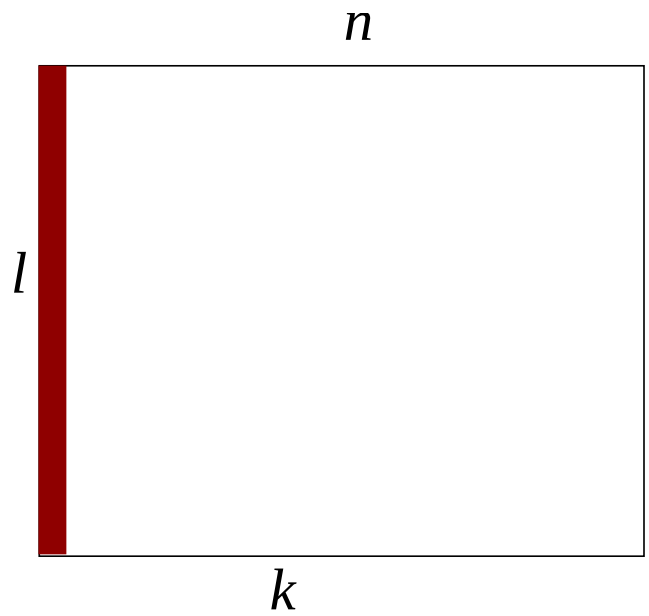
Consider an $\ell \times n$ array code $\mathcal{C}$ ('vector code') over the field F



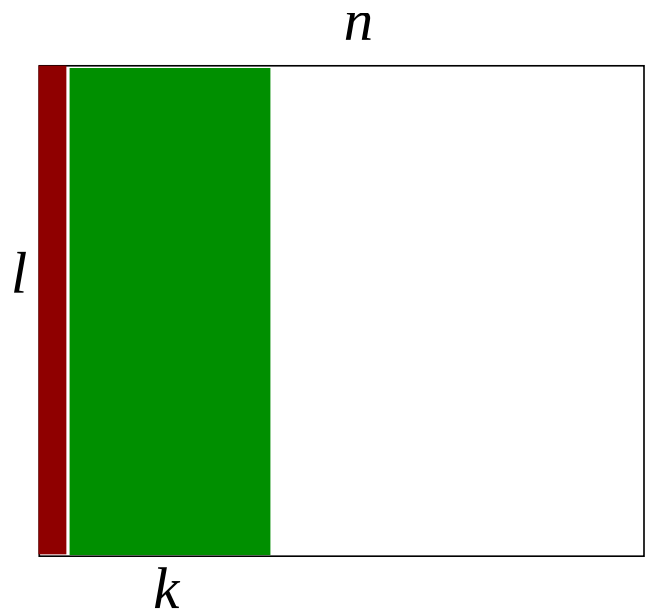
n - code length
k - code dimension, i.e., $|\mathcal{C}| = q^{k\ell}$
ℓ - length of column, or 'sub-packetization'

The code $\mathcal{C}$ is called MDS if any k columns suffice to recover the entire codeword
A column in this matrix is also called a node.

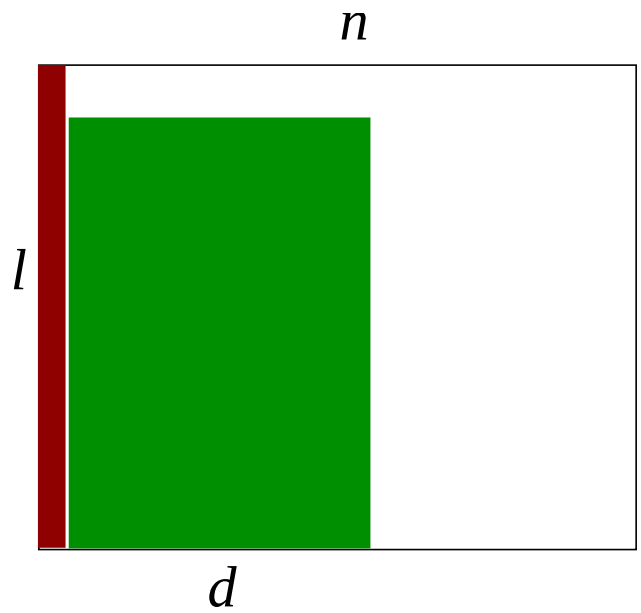
MDS array codes and repair bandwidth



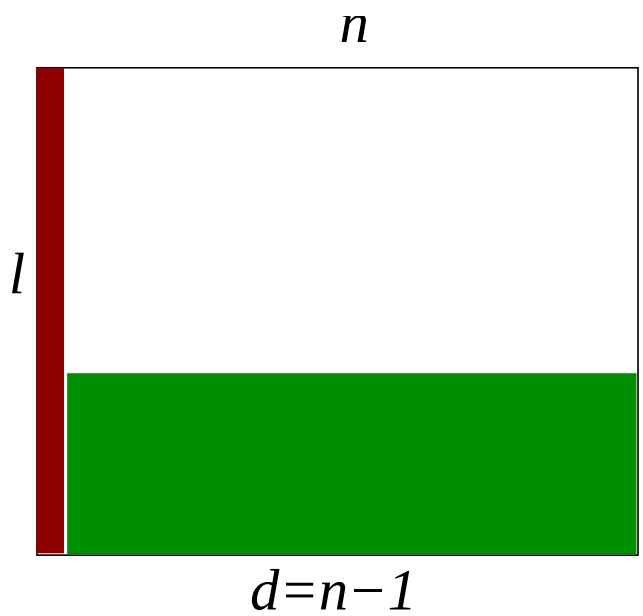
MDS array codes and repair bandwidth



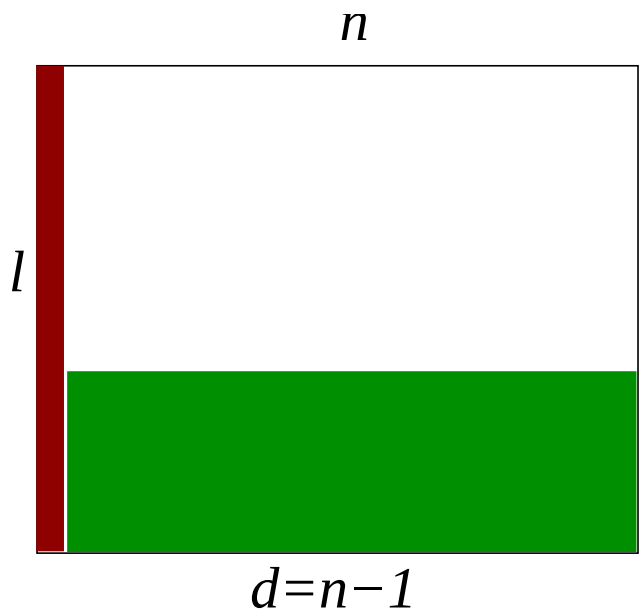
MDS array codes and repair bandwidth



MDS array codes and repair bandwidth

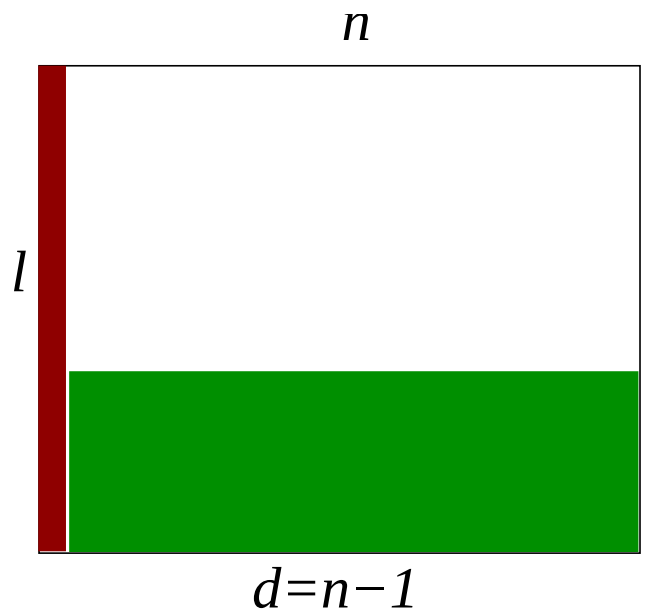


MDS array codes and repair bandwidth



The number of 'downloaded' symbols is called the repair bandwidth β

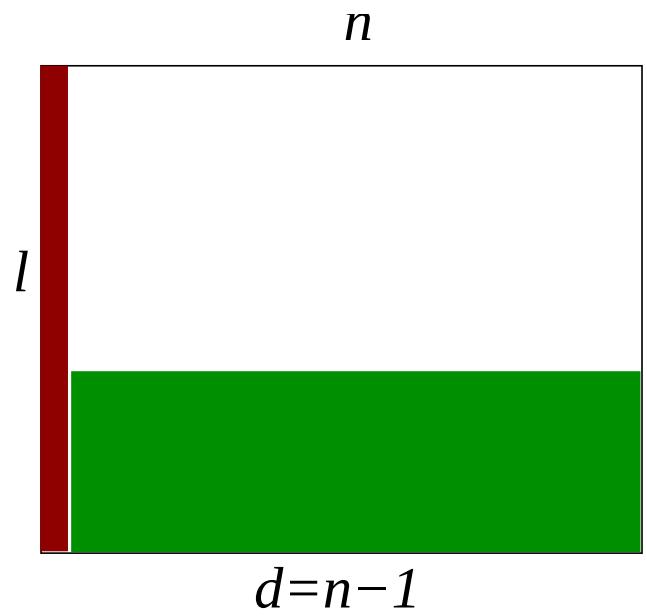
MDS array codes and repair bandwidth



The number of 'downloaded' symbols is called the repair bandwidth β

$$\beta \geq \frac{d\ell}{d+1-k} \qquad \text{(Dimakis et al., 2010)}$$

MDS array codes and repair bandwidth

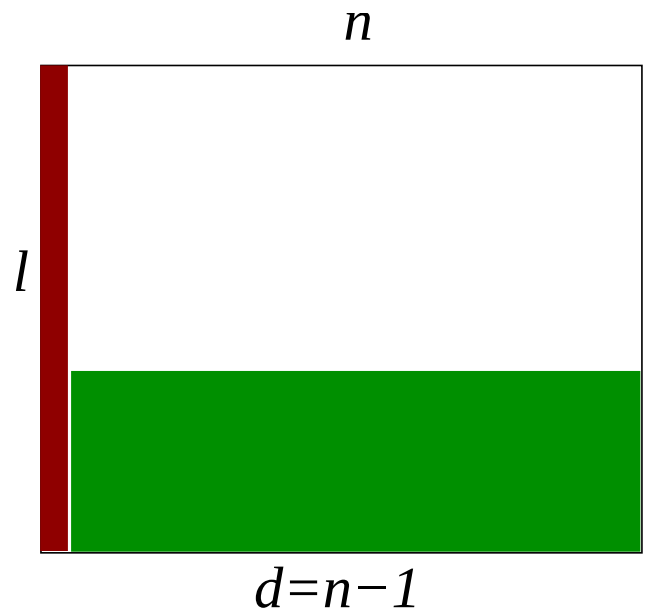


The number of 'downloaded' symbols is called the repair bandwidth β

$$\beta \geq \frac{d\ell}{d+1-k} \qquad \text{(Dimakis et al., 2010)}$$

The right-hand side is a decreasing function of d, thus $\beta \to \min$ if $d = n-1$

MDS array codes and repair bandwidth

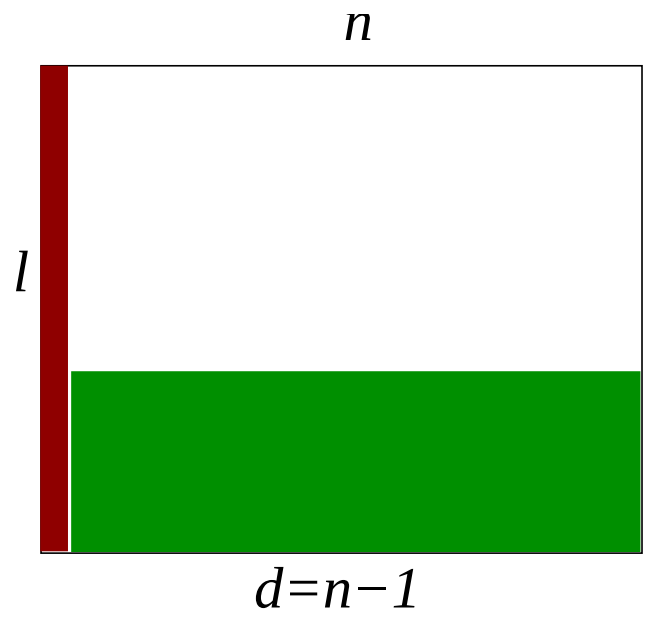


The number of 'downloaded' symbols is called the repair bandwidth β

$$\beta \geq \frac{d\ell}{d+1-k} \qquad \text{(Dimakis et al., 2010)}$$

The right-hand side is a decreasing function of d, thus $\beta \to \min$ if $d = n-1$
The code meeting this bound with equality is said to have optimal repair bandwidth

MDS array codes and repair bandwidth



The number of 'downloaded' symbols is called the repair bandwidth β

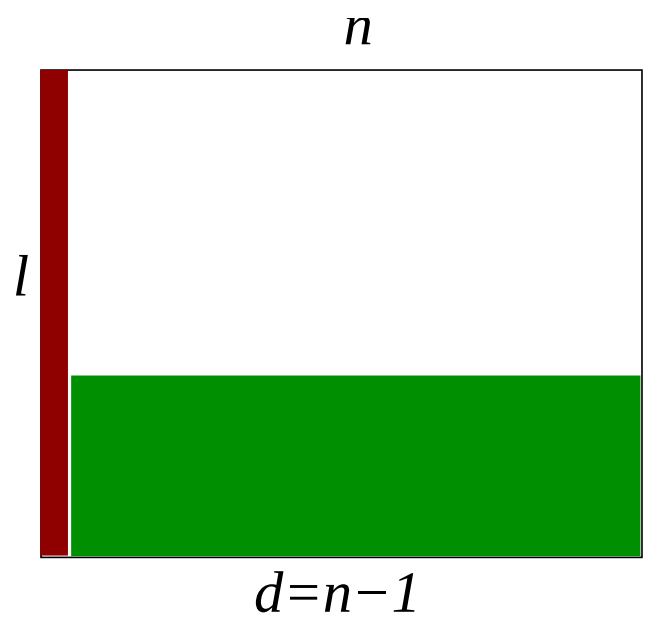
$$\beta \geq \frac{d\ell}{d+1-k} \qquad \text{(Dimakis et al., 2010)}$$

The right-hand side is a decreasing function of d, thus $\beta \to \min$ if $d = n-1$

For $d = n-1$, $r = n-k$

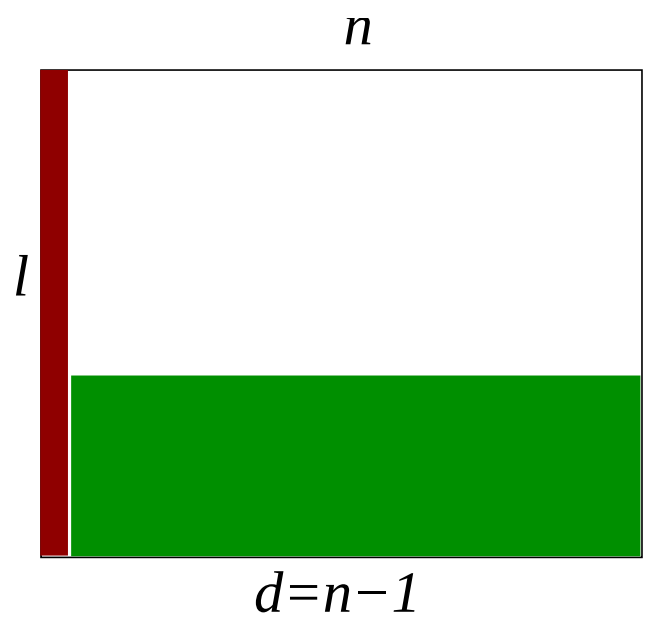
$$\beta \geq \frac{\ell}{r}(n-1)$$

The repair problem



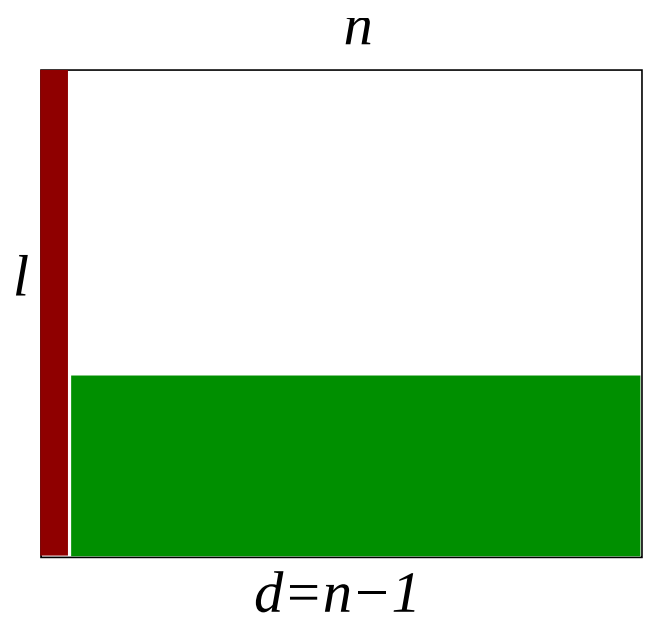
- The contents of the erased column is a function of the contents of the helper nodes.

The repair problem



- The contents of the erased column is a function of the contents of the helper nodes.
- If the downloaded information is the same as the information accessed at the helper nodes, the code is said to support repair by transfer

The repair problem



- The contents of the erased column is a function of the contents of the helper nodes.
- If the downloaded information is the same as the information accessed at the helper nodes, the code is said to support repair by transfer
- If in addition the code has optimal repair bandwidth, it is called the optimal-access code

Repair bandwidth of MDS codes

It is possible to generalize the bound on the minimum bandwidth to the case of $h \geq 1$ failed nodes

Repair bandwidth of MDS codes

It is possible to generalize the bound on the minimum bandwidth to the case of $h \geq 1$ failed nodes

(Cut-set bound, DIMAKIS ET AL., 2010): Suppose that h nodes of the codeword in $\mathcal{C}(n,k,l)$ are unavailable and we use h helper nodes to perform the repair. The repair bandwidth is at least

$$\beta(\mathcal{C},d,h) \geq \frac{hdl}{d-k+1}.$$

Repair bandwidth of MDS codes

It is possible to generalize the bound on the minimum bandwidth to the case of $h \geq 1$ failed nodes

(Cut-set bound, DIMAKIS ET AL., 2010): Suppose that h nodes of the codeword in $\mathcal{C}(n,k,l)$ are unavailable and we use h helper nodes to perform the repair. The repair bandwidth is at least

$$\beta(\mathcal{C},d,h) \geq \frac{hdl}{d-k+1}.$$

The codes that meet this bound are said to have (h,d)-*optimal repair property*

This talk

Part I: Explicit constructions of MDS array codes with optimal repair

Part II: Asymptotically optimal family of Reed-Solomon codes

Part I: Optimal-repair MDS array codes

Previous results (a sampling):

- Low rate $k/n \leq 1/2$: Constructions of codes with $(1, n-1)$ optimal repair (RASHMI ET AL., '11, SUH-RAMCHANDRAN, '11)
- High rate: Constructions for $r = 2, 3$ (TAMO-WANG-BRUCK '13; PAPAILIOPOULOS ET AL. '13; WANG-TAMO-BRUCK '16; RAVIV-SILBERSTEIN-ETZION '17)
- High rate, any r: A number of existence results over large alphabets (CADAMBE ET AL., '12; WANG ET AL., '16; GOPARAJU-FAZELI-VARDY, '16)
- M. YE AND A.B., '16
- SASIDHARAN-VAJHA-KUMAR, '16-'17

Part I: Optimal-repair MDS array codes

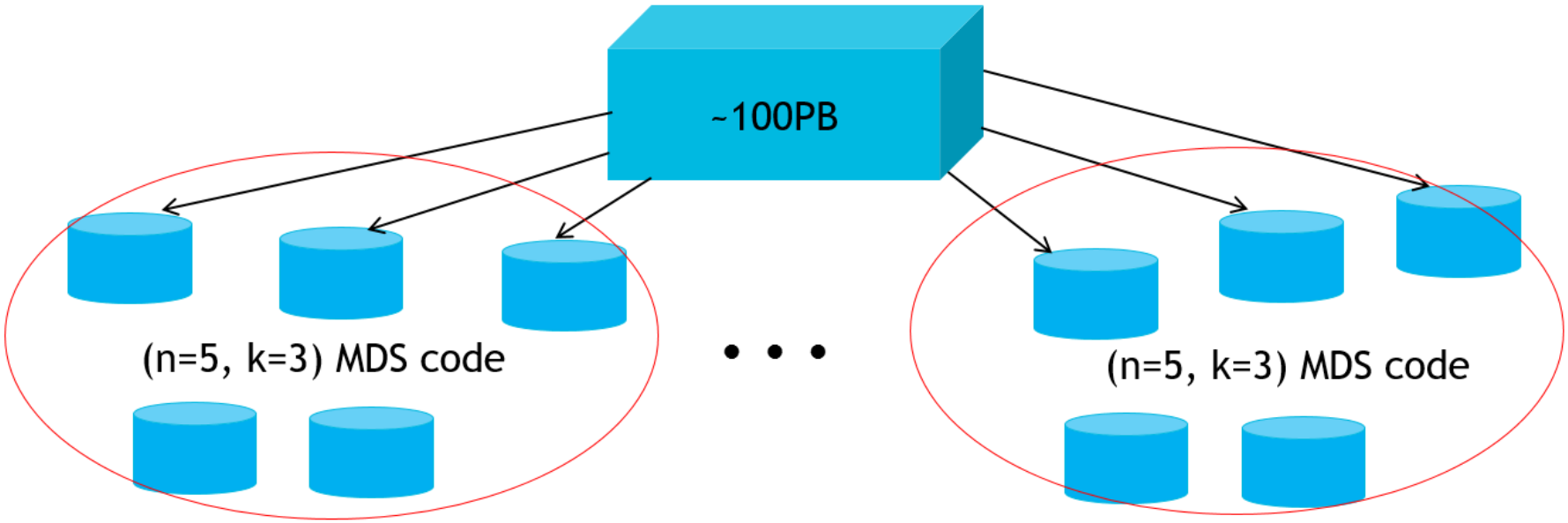
Previous results (a sampling):

- Low rate $k/n \leq 1/2$: Constructions of codes with $(1, n-1)$ optimal repair (RASHMI ET AL., '11, SUH-RAMCHANDRAN, '11)
- High rate: Constructions for $r = 2, 3$ (TAMO-WANG-BRUCK '13; PAPAILIOPOULOS ET AL. '13; WANG-TAMO-BRUCK '16; RAVIV-SILBERSTEIN-ETZION '17)
- High rate, any r: A number of existence results over large alphabets (CADAMBE ET AL., '12; WANG ET AL., '16; GOPARAJU-FAZELI-VARDY, '16)
- M. YE AND A.B., '16
- SASIDHARAN-VAJHA-KUMAR, '16-'17

In this talk (MIN YE and A.B., '16-'17):

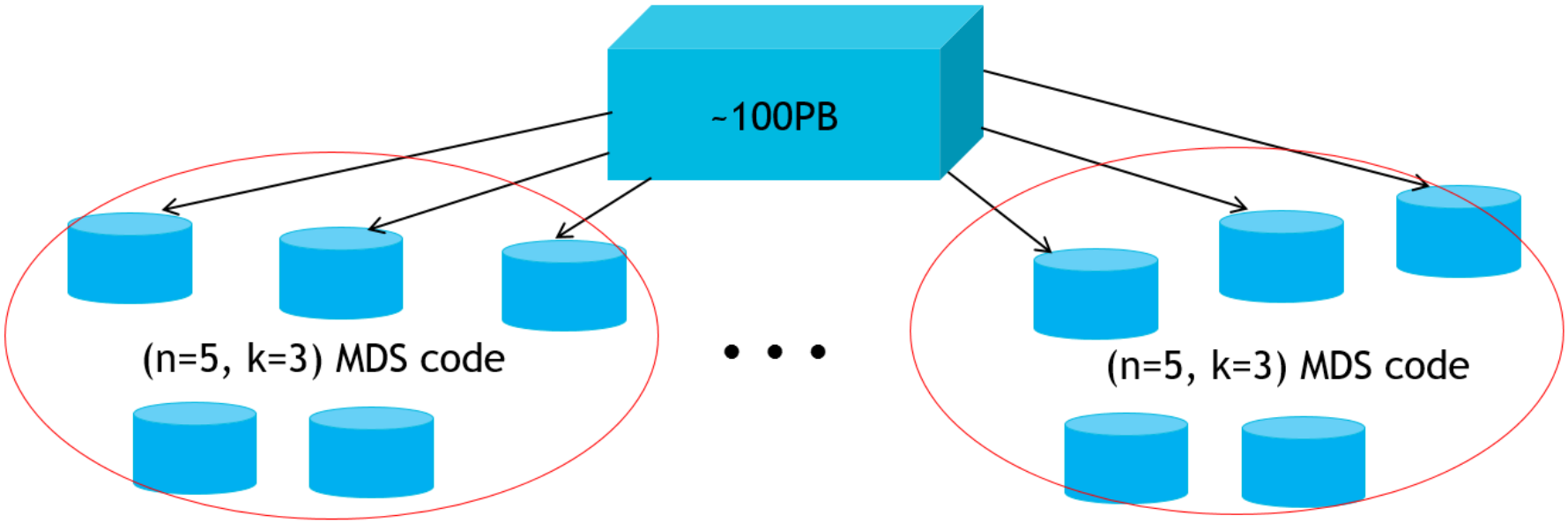
- A construction of (h, d)-optimal repair codes for any r, any $h \leq r$, and $d, k \leq d \leq n - h$
- Code families that work for all h and all d simultaneously
- Error-resilient optimal-repair codes
- Optimal-access codes with the above properties

Placement of MDS codewords in storage nodes



Any k nodes suffice to recover the data

Placement of MDS codewords in storage nodes



Any k nodes suffice to recover the data

Each node stores l symbols of the block

General encoding method

The code is formed of $l \times n$ matrices over F, each encoding kl data symbols.

General encoding method

The code is formed of $l \times n$ matrices over F, each encoding kl data symbols.

- Parity-check equations:

$$\mathcal{C} = \{(C_1, C_2, \ldots, C_n) : \sum_{i=1}^{n} A_{t,i} C_i = 0, t = 1, \ldots, r\}$$

General encoding method

The code is formed of $l \times n$ matrices over F, each encoding kl data symbols.

- Parity-check equations:

$$\mathcal{C} = \{(C_1, C_2, \ldots, C_n) : \sum_{i=1}^{n} A_{t,i} C_i = 0, t = 1, \ldots, r\}$$

- $r \times n$ parity check matrix

$$\begin{bmatrix} A_{1,1} & A_{1,2} & A_{1,3} & \ldots & A_{1,n} \\ A_{2,1} & A_{2,2} & A_{2,3} & \ldots & A_{2,n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ A_{r,1} & A_{r,2} & A_{r,3} & \ldots & A_{r,n} \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ \vdots \\ C_n \end{bmatrix} = 0$$

where each A_{ij} is an $l \times l$ matrix and C_i is an l-vector over F

General encoding method

The code is formed of $l \times n$ matrices over F, each encoding kl data symbols.

- Parity-check equations:

$$\mathcal{C} = \{(C_1, C_2, \ldots, C_n) : \sum_{i=1}^{n} A_{t,i} C_i = 0, t = 1, \ldots, r\}$$

- Choose $(A_{t,i})$ above to follow block Vandermonde structure

$$\begin{bmatrix} I & I & I & \ldots & I \\ A_1 & A_2 & A_3 & \ldots & A_n \\ A_1^2 & A_2^2 & A_3^2 & \ldots & A_n^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ A_1^{r-1} & A_2^{r-1} & A_3^{r-1} & \ldots & A_n^{r-1} \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ \vdots \\ C_n \end{bmatrix} = 0$$

General encoding method

The code is formed of $l \times n$ matrices over F, each encoding kl data symbols.

- Parity-check equations:

$$\mathcal{C} = \{(C_1, C_2, \ldots, C_n) : \sum_{i=1}^{n} A_{t,i} C_i = 0, t = 1, \ldots, r\}$$

- Choose $(A_{t,i})$ above to follow block Vandermonde structure

$$\begin{bmatrix} I & I & I & \ldots & I \\ A_1 & A_2 & A_3 & \ldots & A_n \\ A_1^2 & A_2^2 & A_3^2 & \ldots & A_n^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ A_1^{r-1} & A_2^{r-1} & A_3^{r-1} & \ldots & A_n^{r-1} \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ \vdots \\ C_n \end{bmatrix} = 0$$

- Commuting: $A_i A_j = A_j A_i$

General encoding method

The code is formed of $l \times n$ matrices over F, each encoding kl data symbols.

- Parity-check equations:

$$\mathcal{C} = \{(C_1, C_2, \ldots, C_n) : \sum_{i=1}^{n} A_{t,i} C_i = 0, t = 1, \ldots, r\}$$

- Choose $(A_{t,i})$ above to follow block Vandermonde structure

$$\begin{bmatrix} I & I & I & \ldots & I \\ A_1 & A_2 & A_3 & \ldots & A_n \\ A_1^2 & A_2^2 & A_3^2 & \ldots & A_n^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ A_1^{r-1} & A_2^{r-1} & A_3^{r-1} & \ldots & A_n^{r-1} \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ \vdots \\ C_n \end{bmatrix} = 0$$

- Commuting: $A_i A_j = A_j A_i$
- $A_i - A_j$ invertible

General encoding method

The code is formed of $l \times n$ matrices over F, each encoding kl data symbols.

- Parity-check equations:

$$\mathcal{C} = \{(C_1, C_2, \ldots, C_n) : \sum_{i=1}^{n} A_{t,i} C_i = 0, t = 1, \ldots, r\}$$

- Choose $(A_{t,i})$ above to follow block Vandermonde structure

$$\begin{bmatrix} I & I & I & \ldots & I \\ A_1 & A_2 & A_3 & \ldots & A_n \\ A_1^2 & A_2^2 & A_3^2 & \ldots & A_n^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ A_1^{r-1} & A_2^{r-1} & A_3^{r-1} & \ldots & A_n^{r-1} \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \\ C_3 \\ \vdots \\ C_n \end{bmatrix} = 0$$

- Commuting: $A_i A_j = A_j A_i$
- $A_i - A_j$ invertible
- Natural choice: Diagonal matrices

Construction I: Optimal $(1, n-1)$ repair MDS codes

- Take $l = r^n$; take the $l \times l$ matrix $A_i, i = 1, \ldots, n$ in the form

$$A_i = \sum_{a=0}^{l-1} \lambda_{i,a_i} e_a e_a^T$$

where λ_{ij} are distinct elements of F and $(a_n, a_{n-1}, \ldots, a_1)$ is r-ary expansion of a

[1] Note that we need $r \cdot n$ distinct elements of the finite field F, i.e., $|F| \geq rn$.

[2] By GOPARAJU-TAMO-CALDERBANK '14, $l \geq \exp(\sqrt{k/(2r-1)})$

Construction I: Optimal $(1, n-1)$ repair MDS codes

- Take $l = r^n$; take the $l \times l$ matrix $A_i, i = 1, \ldots, n$ in the form

$$A_i = \sum_{a=0}^{l-1} \lambda_{i,a_i} e_a e_a^T$$

where λ_{ij} are distinct elements of F and $(a_n, a_{n-1}, \ldots, a_1)$ is r-ary expansion of a

[1] Note that we need $r \cdot n$ distinct elements of the finite field F, i.e., $|F| \geq rn$.

[2] By GOPARAJU-TAMO-CALDERBANK '14, $l \geq \exp(\sqrt{k/(2r-1)})$

- The codeword has the form $C = (C_1, \ldots, C_n)$, where $C_i = (c_{i,0}, c_{i,1}, \ldots, c_{i,l-1})^T$

$$\begin{matrix} c_{1,0} & c_{2,0} & \ldots & c_{n,0} \\ c_{1,1} & c_{2,1} & \ldots & c_{n,1} \\ \vdots & \vdots & \vdots & \vdots \\ c_{1,l-1} & c_{2,l-1} & \ldots & c_{n,l-1} \end{matrix}$$

- Idea: Every row forms an RS code with different evaluation points $\{P_{i,j}\}$
  For $a = 0, 1, \ldots, l-1$, write r-ary expansion $a = (a_1, a_2, \ldots, a_n)$
  Evaluation points for a-th row: $(\lambda_{1,a_1}, \lambda_{2,a_2}, \ldots, \lambda_{n,a_n})$

$(1, n-1)$-optimal repair property

Idea (cont'd): Let $C_i = (c_{i,a}, a = 0, 1, \ldots, l-1)$ be the missing node.
Repair the contents by ***groups of size*** r that differ only in position i of the expansion

$(1, n-1)$-optimal repair property

Idea (cont'd): Let $C_i = (c_{i,a}, a = 0, 1, \ldots, l-1)$ be the missing node.
Repair the contents by ***groups of size*** r that differ only in position i of the expansion

- $a(i,u) = (a_1, a_2, \ldots, a_{i-1}, u, a_{i+1}, a_{i+2}, \ldots, a_n), \quad 0 \leq u \leq r-1$

$(1, n-1)$-optimal repair property

Idea (cont'd): Let $C_i = (c_{i,a}, a = 0, 1, \ldots, l-1)$ be the missing node.
Repair the contents by ***groups of size*** r that differ only in position i of the expansion

- $a(i,u) = (a_1, a_2, \ldots, a_{i-1}, u, a_{i+1}, a_{i+2}, \ldots, a_n), \quad 0 \leq u \leq r-1$
- $$\lambda^t_{1,a_1} c_{1,a} + \lambda^t_{2,a_2} c_{2,a} + \cdots + \lambda^t_{n,a_n} c_{n,a} = 0, \qquad t = 0, 1, \ldots, r-1$$

$(1, n-1)$-optimal repair property

Idea (cont'd): Let $C_i = (c_{i,a}, a = 0, 1, \ldots, l-1)$ be the missing node.
Repair the contents by ***groups of size*** r that differ only in position i of the expansion

- $a(i,u) = (a_1, a_2, \ldots, a_{i-1}, u, a_{i+1}, a_{i+2}, \ldots, a_n), \quad 0 \leq u \leq r-1$
- $$\lambda^t_{1,a_1} c_{1,a} + \lambda^t_{2,a_2} c_{2,a} + \cdots + \lambda^t_{n,a_n} c_{n,a} = 0, \qquad t = 0, 1, \ldots, r-1$$
- $$\lambda^t_{i,a_i} c_{i,a} + \sum_{j \neq i} \lambda^t_{j,a_j} c_{j,a} = 0$$

$(1, n-1)$-optimal repair property

Idea (cont'd): Let $C_i = (c_{i,a}, a = 0, 1, \ldots, l-1)$ be the missing node.
Repair the contents by ***groups of size*** r that differ only in position i of the expansion

- $a(i,u) = (a_1, a_2, \ldots, a_{i-1}, u, a_{i+1}, a_{i+2}, \ldots, a_n), \quad 0 \le u \le r-1$
- $$\lambda^t_{1,a_1} c_{1,a} + \lambda^t_{2,a_2} c_{2,a} + \cdots + \lambda^t_{n,a_n} c_{n,a} = 0, \qquad t = 0, 1, \ldots, r-1$$
- $$\lambda^t_{i,a_i} c_{i,a} + \sum_{j \neq i} \lambda^t_{j,a_j} c_{j,a} = 0$$
- $$\lambda^t_{i,u} c_{i,a(i,u)} + \sum_{j \neq i} \lambda^t_{j,a_j} c_{j,a(i,u)} = 0, \qquad u = 0, 1, \ldots, r-1$$

$(1, n-1)$-optimal repair property

Idea (cont'd): Let $C_i = (c_{i,a}, a = 0, 1, \ldots, l-1)$ be the missing node.
Repair the contents by ***groups of size*** r that differ only in position i of the expansion

- $a(i,u) = (a_1, a_2, \ldots, a_{i-1}, u, a_{i+1}, a_{i+2}, \ldots, a_n), \quad 0 \leq u \leq r-1$
- $$\lambda^t_{1,a_1} c_{1,a} + \lambda^t_{2,a_2} c_{2,a} + \cdots + \lambda^t_{n,a_n} c_{n,a} = 0, \qquad t = 0, 1, \ldots, r-1$$
- $$\lambda^t_{i,a_i} c_{i,a} + \sum_{j \neq i} \lambda^t_{j,a_j} c_{j,a} = 0$$
- $$\lambda^t_{i,u} c_{i,a(i,u)} + \sum_{j \neq i} \lambda^t_{j,a_j} c_{j,a(i,u)} = 0, \qquad u = 0, 1, \ldots, r-1$$
- $$\sum_{u=0}^{r-1} \lambda^t_{i,u} c_{i,a(i,u)} + \sum_{u=0}^{r-1} \sum_{j \neq i} \lambda^t_{j,a_j} c_{j,a(i,u)} = 0$$

$(1, n-1)$-optimal repair property

$$\sum_{u=0}^{r-1} \lambda_{i,u}^{t} c_{i,a(i,u)} + \sum_{j \neq i} \Big(\sum_{u=0}^{r-1} \lambda_{j,a_j}^{t} c_{j,a(i,u)} \Big) = 0, \qquad t = 0, 1, \ldots, r-1$$

$(1, n-1)$-optimal repair property

$$\sum_{u=0}^{r-1} \lambda_{i,u}^{t} c_{i,a(i,u)} + \sum_{j \neq i} \Big(\sum_{u=0}^{r-1} \lambda_{j,a_j}^{t} c_{j,a(i,u)} \Big) = 0, \qquad t = 0, 1, \ldots, r-1$$

$$\underbrace{\begin{bmatrix} 1 & 1 & \cdots & 1 \\ \lambda_{i,0} & \lambda_{i,1} & \cdots & \lambda_{i,r-1} \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_{i,0}^{r-1} & \lambda_{i,1}^{r-1} & \cdots & \lambda_{i,r-1}^{r-1} \end{bmatrix}}_{\text{Vandermonde, rank } = r} \underbrace{\begin{bmatrix} c_{i,a(i,0)} \\ c_{i,a(i,1)} \\ \vdots \\ c_{i,a(i,r-1)} \end{bmatrix}}_{\text{missing information}}$$

$$\sum_{u=0}^{r-1} c_{j,a(i,u)}$$

$$+ \sum_{j \neq i} \underbrace{\begin{bmatrix} 1 & 1 & \cdots & 1 \\ \lambda_{j,a_j} & \lambda_{j,a_j} & \cdots & \lambda_{j,a_j} \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_{j,a_j}^{r-1} & \lambda_{j,a_j}^{r-1} & \cdots & \lambda_{j,a_j}^{r-1} \end{bmatrix}}_{\text{aligned, rank } = 1} \begin{bmatrix} c_{j,a(i,0)} \\ c_{j,a(i,1)} \\ \vdots \\ c_{j,a(i,r-1)} \end{bmatrix} = 0$$

$(1, n-1)$-optimal repair property

- For $u = 0, 1, \ldots, r-1$ let $a(i,u) := (a_n, \ldots, a_{i+1}, u, a_{i-1}, \ldots, a_1)$.
- Partition the symbols on the failed node i into r^{l-1} groups of size r each:

$$\{c_{i,a(i,0)}, c_{i,a(i,1)}, \ldots, c_{i,a(i,r-1)}\}$$

  for some $a \in \{0, 1, \ldots, l-1\}$

- Say C_i is unavailable. The elements in each group can be found by acquiring ***one element*** $\sum_{u=0}^{r-1} c_{ja(i,u)}$ from each of the $n-1$ remaining nodes.

- Total repair bandwidth $= (n-1) \times 1 \times r^{l-1} = (n-1)(l/r)$, matching the lower bound

Construction II: Codes with $(1, d)$-optimal repair and error correction, $k < d < n - 1$

The bound on the repair bandwidth can be extended to error correction:

$$\beta(d, t) \geq \frac{(d + 2t)l}{d + 1 - k}$$

(PAWAR ET AL., '11; RASHMI ET AL., '12)

Construction II: Codes with $(1, d)$-optimal repair and error correction, $k < d < n - 1$

The bound on the repair bandwidth can be extended to error correction:

$$\beta(d, t) \geq \frac{(d + 2t)l}{d + 1 - k}$$

(PAWAR ET AL., '11; RASHMI ET AL., '12)

- Construct (n, k) MDS code $\mathcal{C}$ over $(F)^l$, $l = s^n$, $s = d + 1 - k$
- Take $|F| \geq sn$, $\{\lambda_{ij}, i = 1, \ldots, n; j = 0, \ldots, s - 1\} \subset F$,
- Parity-check matrix H is a block Vandermonde matrix with

$$A_i = \sum_{a=0}^{l-1} \lambda_{i,a_i} e_a e_a^T$$

  where $a = (a_n, a_{n-1}, \ldots, a_1)$ is the s-ary expansion of $a = 0, 1, \ldots, l - 1$.
- We use the fact that the symbols transmitted from the helper nodes form a (generalized) RS code

Construction III: Codes with (h, d)-optimal repair for all h, d simultaneously

We take $s = \text{lcm}(1, 2, \ldots, r)$

Sequential repair:

Let $C_1, \ldots, C_h$ be the failed nodes, $\mathcal{R}, |\mathcal{R}| = d + 2t$ be the helper set.

Construction III: Codes with (h, d)-optimal repair for all h, d simultaneously

We take $s = \text{lcm}(1, 2, \ldots, r)$

Sequential repair:

Let $C_1, \ldots, C_h$ be the failed nodes, $\mathcal{R}, |\mathcal{R}| = d + 2t$ be the helper set.

We use

- $C_{\mathcal{R}}$ to repair C_1

Construction III: Codes with (h, d)-optimal repair for all h, d simultaneously

We take $s = \text{lcm}(1, 2, \ldots, r)$

Sequential repair:

Let $C_1, \ldots, C_h$ be the failed nodes, $\mathcal{R}, |\mathcal{R}| = d + 2t$ be the helper set.

We use

- $C_{\mathcal{R}}$ to repair C_1
- $C_{\mathcal{R}} \cup C_1$ to repair C_2

Construction III: Codes with (h, d)-optimal repair for all h, d simultaneously

We take $s = \text{lcm}(1, 2, \ldots, r)$

Sequential repair:

Let $C_1, \ldots, C_h$ be the failed nodes, $\mathcal{R}, |\mathcal{R}| = d + 2t$ be the helper set.

We use

- $C_{\mathcal{R}}$ to repair C_1
- $C_{\mathcal{R}} \cup C_1$ to repair C_2
- $C_{\mathcal{R}} \cup C_1 \cup C_2$ to repair C_3

Construction III: Codes with (h, d)-optimal repair for all h, d simultaneously

We take $s = \text{lcm}(1, 2, \ldots, r)$

Sequential repair:

Let $C_1, \ldots, C_h$ be the failed nodes, $\mathcal{R}, |\mathcal{R}| = d + 2t$ be the helper set.

We use

- $C_{\mathcal{R}}$ to repair C_1
- $C_{\mathcal{R}} \cup C_1$ to repair C_2
- $C_{\mathcal{R}} \cup C_1 \cup C_2$ to repair C_3

  $\ldots$

Construction III: Codes with (h, d)-optimal repair for all h, d simultaneously

We take $s = \text{lcm}(1, 2, \ldots, r)$

Sequential repair:

Let $C_1, \ldots, C_h$ be the failed nodes, $\mathcal{R}, |\mathcal{R}| = d + 2t$ be the helper set.

We use

- $C_{\mathcal{R}}$ to repair C_1
- $C_{\mathcal{R}} \cup C_1$ to repair C_2
- $C_{\mathcal{R}} \cup C_1 \cup C_2$ to repair C_3

  $\ldots$

- $C_{\mathcal{R}} \cup C_1 \cup C_2 \cup \cdots \cup C_{h-1}$ to repair C_h

Further extensions

1. d-optimal repair $(n, k = n - r, l = r^{n-1})$ MDS codes with optimal access property and error correction, $|F| \geq n + 1$
   (Idea: Use permutation matrices instead of diagonal matrices)

2. Codes for optimal repair from $d_1, \ldots, d_m$ helper nodes simultaneously, $k \leq d_1, \ldots, d_m < n$

Further extensions

1. d-optimal repair $(n, k = n - r, l = r^{n-1})$ MDS codes with optimal access property and error correction, $|F| \geq n + 1$
   (Idea: Use permutation matrices instead of diagonal matrices)

2. Codes for optimal repair from $d_1, \ldots, d_m$ helper nodes simultaneously, $k \leq d_1, \ldots, d_m < n$

The codes have small encoding, repair, and update complexity.

They also afford efficient *systematic representation*

Systematic encoding; Updates

For $a = 0, 1, \ldots, l-1$

$$\begin{bmatrix} 1 & 1 & \cdots & 1 \\ \lambda_{k+1,a_{k+1}} & \lambda_{k+2,a_{k+2}} & \cdots & \lambda_{k+r,a_{k+r}} \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_{k+1,a_{k+1}}^{r-1} & \lambda_{k+2,a_{k+2}}^{r-1} & \cdots & \lambda_{k+r,a_{k+r}}^{r-1} \end{bmatrix} \begin{bmatrix} c_{k+1,a} \\ c_{k+2,a} \\ \vdots \\ c_{k+r,a} \end{bmatrix}$$

$$= - \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \lambda_{1,a_1} & \lambda_{2,a_2} & \cdots & \lambda_{k,a_k} \\ \vdots & \vdots & \vdots & \vdots \\ \lambda_{1,a_1}^{r-1} & \lambda_{2,a_2}^{r-1} & \cdots & \lambda_{k,a_k}^{r-1} \end{bmatrix} \begin{bmatrix} c_{1,a} \\ c_{2,a} \\ \vdots \\ c_{k,a} \end{bmatrix}$$

- To encode, invert l matrices of dimension $r \times r$
- To update $c_{i,a}$, re-compute $c_{k+1,a}, \ldots, c_{k+r,a}$. To update one element of an information node, it is necessary to update at least one element in every parity node (TAMO ET AL., '14). Our codes have optimal update complexity.

Part II: Repair bandwidth of Reed-Solomon codes

It is possible to repair RS codes better than discussed in the beginning of the talk.

Part II: Repair bandwidth of Reed-Solomon codes

It is possible to repair RS codes better than discussed in the beginning of the talk.

Idea: Given an RS code over E, "vectorize" it (consider as an array code over a subfield F of E with l symbols of F per node).

Part II: Repair bandwidth of Reed-Solomon codes

It is possible to repair RS codes better than discussed in the beginning of the talk.

Idea: Given an RS code over E, "vectorize" it (consider as an array code over a subfield F of E with l symbols of F per node).

Example:

Part II: Repair bandwidth of Reed-Solomon codes

It is possible to repair RS codes better than discussed in the beginning of the talk.

Idea: Given an RS code over E, "vectorize" it (consider as an array code over a subfield F of E with l symbols of F per node).

Example:

- Consider an RS code over $E = \mathbb{F}_{16}$ as an array code over $F = \mathbb{F}_2$, i.e., $l = 4$

Part II: Repair bandwidth of Reed-Solomon codes

It is possible to repair RS codes better than discussed in the beginning of the talk.

Idea: Given an RS code over E, "vectorize" it (consider as an array code over a subfield F of E with l symbols of F per node).

Example:

- Consider an RS code over $E = \mathbb{F}_{16}$ as an array code over $F = \mathbb{F}_2$, i.e., $l = 4$
- E can be represented as a 4-dimensional vector space over $F = \{0, 1\}$

Part II: Repair bandwidth of Reed-Solomon codes

It is possible to repair RS codes better than discussed in the beginning of the talk.

Idea: Given an RS code over E, "vectorize" it (consider as an array code over a subfield F of E with l symbols of F per node).

Example:

- Consider an RS code over $E = \mathbb{F}_{16}$ as an array code over $F = \mathbb{F}_2$, i.e., $l = 4$
- E can be represented as a 4-dimensional vector space over $F = \{0, 1\}$
- Let $\alpha \in E$ be such that $\alpha^4 = \alpha + 1$, then $(1, \alpha, \alpha^2, \alpha^3)$ form a basis of E over F

Part II: Repair bandwidth of Reed-Solomon codes

It is possible to repair RS codes better than discussed in the beginning of the talk.

Idea: Given an RS code over E, "vectorize" it (consider as an array code over a subfield F of E with l symbols of F per node).

Example:

- Consider an RS code over $E = \mathbb{F}_{16}$ as an array code over $F = \mathbb{F}_2$, i.e., $l = 4$
- E can be represented as a 4-dimensional vector space over $F = \{0, 1\}$
- Let $\alpha \in E$ be such that $\alpha^4 = \alpha + 1$, then $(1, \alpha, \alpha^2, \alpha^3)$ form a basis of E over F
- Trace $\mathrm{tr}(x) = x + x^2 + x^{2^2} + x^{2^3}$ is a map from E to F:

$$\mathrm{tr}(0) = 0, \mathrm{tr}(1) = 0, \mathrm{tr}(\alpha) = 1, \quad \text{etc.}$$

Part II: Repair bandwidth of Reed-Solomon codes

It is possible to repair RS codes better than discussed in the beginning of the talk.

Idea: Given an RS code over E, "vectorize" it (consider as an array code over a subfield F of E with l symbols of F per node).

Example:

- Consider an RS code over $E = \mathbb{F}_{16}$ as an array code over $F = \mathbb{F}_2$, i.e., $l = 4$
- E can be represented as a 4-dimensional vector space over $F = \{0, 1\}$
- Let $\alpha \in E$ be such that $\alpha^4 = \alpha + 1$, then $(1, \alpha, \alpha^2, \alpha^3)$ form a basis of E over F
- Trace $\mathrm{tr}(x) = x + x^2 + x^{2^2} + x^{2^3}$ is a map from E to F:

$$\mathrm{tr}(0) = 0, \mathrm{tr}(1) = 0, \mathrm{tr}(\alpha) = 1, \quad \text{etc.}$$

- For any $c \in E$ the values $\mathrm{tr}(c), \mathrm{tr}(\alpha c), \mathrm{tr}(\alpha^2 c), \mathrm{tr}(\alpha^3 c)$ suffice to recover c

General repair scheme

Let $F \subset E$ be finite fields, $[E : F] = l$; $\Lambda \subset E$; $|\Lambda| = \{P_1, \ldots, P_n\}$
Let $\mathcal{C} = RS(n, k, \Lambda)$ be the RS code; $r = n - k$

General repair scheme

Let $F \subset E$ be finite fields, $[E : F] = l$; $\Lambda \subset E$; $|\Lambda| = \{P_1, \ldots, P_n\}$
Let $\mathcal{C} = RS(n, k, \Lambda)$ be the RS code; $r = n - k$

The repair scheme of GURUSWAMI-WOOTTERS '16: For a given $i \in [n]$

General repair scheme

Let $F \subset E$ be finite fields, $[E : F] = l$; $\Lambda \subset E$; $|\Lambda| = \{P_1, \ldots, P_n\}$
Let $\mathcal{C} = RS(n, k, \Lambda)$ be the RS code; $r = n - k$

The repair scheme of GURUSWAMI-WOOTTERS '16: For a given $i \in [n]$

- Let $b_1, b_2, \ldots, b_l \in \mathcal{C}^{\perp}$ be such that $b_{1,i}, \ldots, b_{l,i}$ form a basis of E over F

General repair scheme

Let $F \subset E$ be finite fields, $[E : F] = l$; $\Lambda \subset E$; $|\Lambda| = \{P_1, \ldots, P_n\}$
Let $\mathcal{C} = RS(n, k, \Lambda)$ be the RS code; $r = n - k$

The repair scheme of GURUSWAMI-WOOTTERS '16: For a given $i \in [n]$

- Let $b_1, b_2, \ldots, b_l \in \mathcal{C}^\perp$ be such that $b_{1,i}, \ldots, b_{l,i}$ form a basis of E over F
- The values $\mathrm{tr}(b_{ji} c_i)$ suffice to recover c_i

General repair scheme

Let $F \subset E$ be finite fields, $[E : F] = l$; $\Lambda \subset E$; $|\Lambda| = \{P_1, \ldots, P_n\}$
Let $\mathcal{C} = RS(n, k, \Lambda)$ be the RS code; $r = n - k$

The repair scheme of GURUSWAMI-WOOTTERS '16: For a given $i \in [n]$

- Let $b_1, b_2, \ldots, b_l \in \mathcal{C}^\perp$ be such that $b_{1,i}, \ldots, b_{l,i}$ form a basis of E over F
- The values $\mathrm{tr}(b_{ji}c_i)$ suffice to recover c_i
- We have $\mathrm{tr}(b_{ji}c_i) = -\sum_{t \neq i} \mathrm{tr}(b_{ji}c_t), j = 1, \ldots, l$

General repair scheme

Let $F \subset E$ be finite fields, $[E : F] = l$; $\Lambda \subset E$; $|\Lambda| = \{P_1, \ldots, P_n\}$
Let $\mathcal{C} = RS(n, k, \Lambda)$ be the RS code; $r = n - k$

The repair scheme of GURUSWAMI-WOOTTERS '16: For a given $i \in [n]$

- Let $b_1, b_2, \ldots, b_l \in \mathcal{C}^\perp$ be such that $b_{1,i}, \ldots, b_{l,i}$ form a basis of E over F
- The values $\mathrm{tr}(b_{ji}c_i)$ suffice to recover c_i
- We have $\mathrm{tr}(b_{ji}c_i) = -\sum_{t \neq i} \mathrm{tr}(b_{ji}c_t), j = 1, \ldots, l$
- We need $\{\mathrm{tr}(b_{jt}c_t), j = 1, \ldots, l; t \neq i\}$

General repair scheme

Let $F \subset E$ be finite fields, $[E : F] = l$; $\Lambda \subset E$; $|\Lambda| = \{P_1, \ldots, P_n\}$
Let $\mathcal{C} = RS(n, k, \Lambda)$ be the RS code; $r = n - k$

The repair scheme of GURUSWAMI-WOOTTERS '16: For a given $i \in [n]$

- Let $b_1, b_2, \ldots, b_l \in \mathcal{C}^\perp$ be such that $b_{1,i}, \ldots, b_{l,i}$ form a basis of E over F
- The values $\mathrm{tr}(b_{ji}c_i)$ suffice to recover c_i
- We have $\mathrm{tr}(b_{ji}c_i) = -\sum_{t \neq i} \mathrm{tr}(b_{ji}c_t), j = 1, \ldots, l$
- We need $\{\mathrm{tr}(b_{jt}c_t), j = 1, \ldots, l; t \neq i\}$
- Let B_t be a maximum-size linearly independent subset of $\{b_{jt}, j = 1 \ldots l\}$
  We can find c_i from $\cup_{t \neq i}\{\mathrm{tr}(\beta c_t), \beta \in B_t\}$

General repair scheme

Let $F \subset E$ be finite fields, $[E : F] = l$; $\Lambda \subset E$; $|\Lambda| = \{P_1, \ldots, P_n\}$
Let $\mathcal{C} = RS(n, k, \Lambda)$ be the RS code; $r = n - k$

The repair scheme of GURUSWAMI-WOOTTERS '16: For a given $i \in [n]$

- Let $b_1, b_2, \ldots, b_l \in \mathcal{C}^{\perp}$ be such that $b_{1,i}, \ldots, b_{l,i}$ form a basis of E over F
- The values $\mathrm{tr}(b_{ji}c_i)$ suffice to recover c_i
- We have $\mathrm{tr}(b_{ji}c_i) = -\sum_{t \neq i} \mathrm{tr}(b_{ji}c_t), j = 1, \ldots, l$
- We need $\{\mathrm{tr}(b_{jt}c_t), j = 1, \ldots, l; t \neq i\}$
- Let B_t be a maximum-size linearly independent subset of $\{b_{jt}, j = 1 \ldots l\}$
  We can find c_i from $\cup_{t \neq i}\{\mathrm{tr}(\beta c_t), \beta \in B_t\}$
- Guruswami and Wootters attain a 1.5 times optimal bandwidth

Asymptotically optimal construction

- Let $E = F(\beta)$, where β is an element of degree $l = r^n$

Asymptotically optimal construction

- Let $E = F(\beta)$, where β is an element of degree $l = r^n$
- Choose $\Lambda = \{\beta, \beta^r, \ldots, \beta^{r^{n-1}}\}$

Asymptotically optimal construction

- Let $E = F(\beta)$, where β is an element of degree $l = r^n$
- Choose $\Lambda = \{\beta, \beta^r, \ldots, \beta^{r^{n-1}}\}$
- For $a = 0, 1, \ldots, l-1$ let $a = (a_n, a_{n-1}, \ldots, a_1)$ be its r-ary representation

Asymptotically optimal construction

- Let $E = F(\beta)$, where β is an element of degree $l = r^n$
- Choose $\Lambda = \{\beta, \beta^r, \ldots, \beta^{r^{n-1}}\}$
- For $a = 0, 1, \ldots, l-1$ let $a = (a_n, a_{n-1}, \ldots, a_1)$ be its r-ary representation
- Define

$$f_{ij}(x) = \beta^a x^s, \text{ where } a_i = 0, s = 0, 1, \ldots, r-1$$

(here $j = 1, \ldots, l$)

Asymptotically optimal construction

- Let $E = F(\beta)$, where β is an element of degree $l = r^n$
- Choose $\Lambda = \{\beta, \beta^r, \ldots, \beta^{r^{n-1}}\}$
- For $a = 0, 1, \ldots, l-1$ let $a = (a_n, a_{n-1}, \ldots, a_1)$ be its r-ary representation
- Define

$$f_{ij}(x) = \beta^a x^s, \text{ where } a_i = 0, s = 0, 1, \ldots, r-1$$

(here $j = 1, \ldots, l$)

- $\{f_{ij}(\beta^{r^{i-1}}), j = 1, \ldots, l\} = \{1, \beta, \ldots, \beta^{l-1}\}$

Asymptotically optimal construction

- Let $E = F(\beta)$, where β is an element of degree $l = r^n$
- Choose $\Lambda = \{\beta, \beta^r, \ldots, \beta^{r^{n-1}}\}$
- For $a = 0, 1, \ldots, l-1$ let $a = (a_n, a_{n-1}, \ldots, a_1)$ be its r-ary representation
- Define

$$f_{ij}(x) = \beta^a x^s, \text{ where } a_i = 0, s = 0, 1, \ldots, r-1$$

(here $j = 1, \ldots, l$)

- $\{f_{ij}(\beta^{r^{i-1}}), j = 1, \ldots, l\} = \{1, \beta, \ldots, \beta^{l-1}\}$
-

$$\sum_{0 \le t < n, t \ne i-1} \dim_F(\{f_{ij}(\beta^{r^t}), j = 1, \ldots, l\}) \le l\frac{n+1}{n-k}$$

Asymptotically optimal construction

- Let $E = F(\beta)$, where β is an element of degree $l = r^n$
- Choose $\Lambda = \{\beta, \beta^r, \ldots, \beta^{r^{n-1}}\}$
- For $a = 0, 1, \ldots, l-1$ let $a = (a_n, a_{n-1}, \ldots, a_1)$ be its r-ary representation
- Define

$$f_{ij}(x) = \beta^a x^s, \text{ where } a_i = 0, s = 0, 1, \ldots, r-1$$

(here $j = 1, \ldots, l$)

- $\{f_{ij}(\beta^{r^{i-1}}), j = 1, \ldots, l\} = \{1, \beta, \ldots, \beta^{l-1}\}$
-

$$\sum_{0 \le t < n, t \ne i-1} \dim_F(\{f_{ij}(\beta^{r^t}), j = 1, \ldots, l\}) \le l\frac{n+1}{n-k}$$

For large n the repair bandwidth approaches the cut-set bound

Improved construction (announcement)

It is possible to achieve the cut-set bound for RS code repair exactly:

Improved construction (announcement)

It is possible to achieve the cut-set bound for RS code repair exactly:

- We present an explicit construction of RS codes that use the minimum possible repair bandwidth.
- We also prove a lower bound on the sub-packetization value of optimally repaired linear MDS codes

Improved construction (announcement)

It is possible to achieve the cut-set bound for RS code repair exactly:

- We present an explicit construction of RS codes that use the minimum possible repair bandwidth.
- We also prove a lower bound on the sub-packetization value of optimally repaired linear MDS codes

Theorem (Tamo-Ye-B., arXiv1706:00112)

- *Let* $\mathcal{C}$ *be an* $(n, k = n - r)$ *scalar linear MDS code over the field* $E = \mathbb{F}_{q^l}$
- *Let* d *be an integer satisfying* $k + 1 \leq d \leq n - 1$
- *Suppose that for any single failed node of* $\mathcal{C}$ *and any* d *helper nodes there is a linear repair scheme over* $\mathbb{F}_q$ *that uses the bandwidth* $dl/(d + 1 - k)$ *symbols of* $\mathbb{F}_q$
- *For a fixed* $s = d + 1 - k$ *and* $n, k \to \infty$ *the following bounds hold true:*

$$e^{(1+o(1))k \log k} \leq l \leq e^{(1+o(1))n \log n}.$$

References

1. M. YE AND A.B., *Explicit constructions of high-rate MDS array codes with optimal repair bandwidth*, T-IT 63, 4, 2017

2. M. YE AND A.B., *Explicit constructions of MDS array codes and RS codes with optimal repair bandwidth*, Proc. ISIT 2016

3. M. YE AND A.B., *Explicit constructions of optimal-access MDS codes with nearly optimal sub-packetization*, T-IT 63, 2017

4. I. TAMO, M. YE, AND A.B., *Optimal repair of Reed-Solomon codes: Achieving the cut-set bound*, arXiv:1706.00112

References

1. M. YE AND A.B., *Explicit constructions of high-rate MDS array codes with optimal repair bandwidth*, T-IT 63, 4, 2017

2. M. YE AND A.B., *Explicit constructions of MDS array codes and RS codes with optimal repair bandwidth*, Proc. ISIT 2016

3. M. YE AND A.B., *Explicit constructions of optimal-access MDS codes with nearly optimal sub-packetization*, T-IT 63, 2017

4. I. TAMO, M. YE, AND A.B., *Optimal repair of Reed-Solomon codes: Achieving the cut-set bound*, arXiv:1706.00112

THANK YOU!