On Linear Field Size Access-Optimal MDS Convertible Codes

Myna Vajha

Indian Institute of Technology, Hyderabad

GMU Talk, June 30

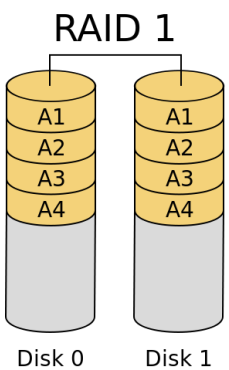
Outline

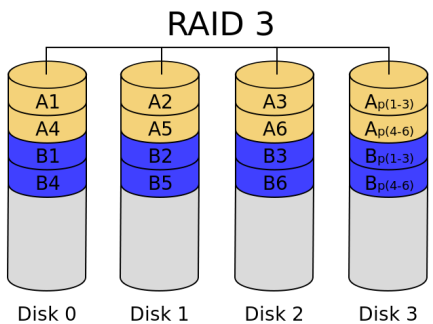
- Erasure Codes and Distributed Storage.
  - MDS codes
- Code conversion problem
  - MDS convertible codes
  - Access and Bandwidth Cost
  - Access Optimal Constructions
  - Bandwidth Optimal Constructions
- Node repair problem (Overview)

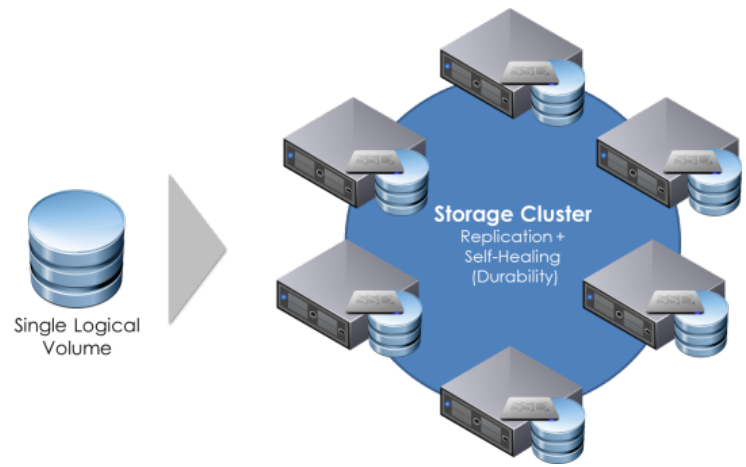
How to save yourself from Data Loss ?

1. Redundant Array of independent Disks (RAID)
   - RAID 1: uses replication
   - RAID 3: adds a parity

2. Distributed Storage (Replication or Erasure Codes)







2-Replication

$$A_{p(1-3)} = A_1 \oplus A_2 \oplus A_3$$

DSS

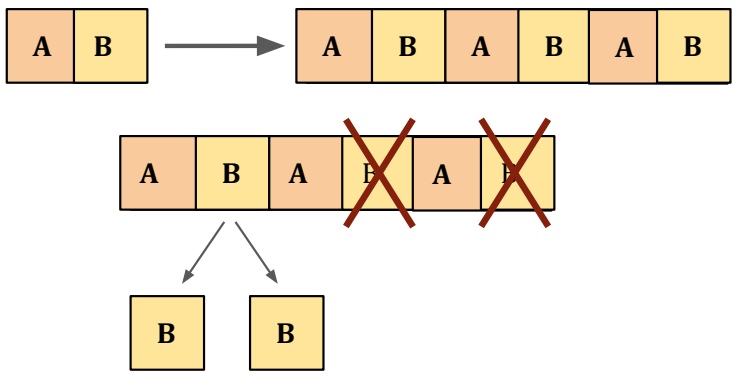
Image Souces: Wikipedia, vifx.co.nz, techmissile.com

Erasure Code

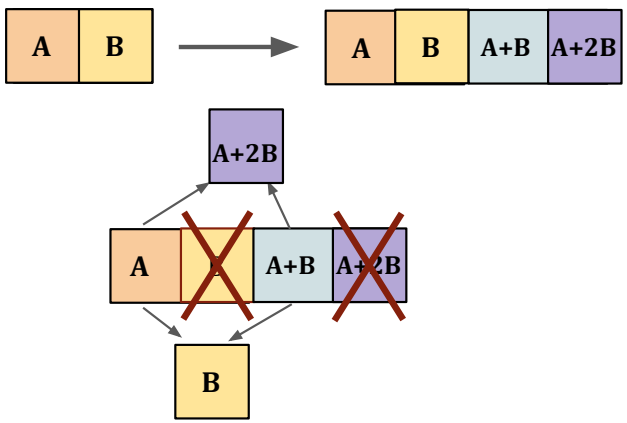
- Used to recover from data loss.
- In a (n, k) erasure code, k units of data are encoded to get n units of data.
  - Storage Overhead of such a code $\nu = \frac{n}{k} = 1 + \frac{r}{k}$
  - erasure tolerance (number of erasures (any pattern) that can be corrected)
- A Maximum Distance Separable (MDS) Code provides reliability against erasure of any $r = n - k$ units.
  - $(6, 4)$ MDS code below has storage overhead 1.5 and it can recover from 2 erasures.

Replication Code vs Erasure Code

- Google's GFS uses 3-replication code.
- Facebook's Hadoop uses [14, 10] Reed Solomon(RS) Code.



(6,2) 3-Replication



[4,2] Reed Solomon

Code	Storage O/h	Bandwidth [1]	Reliability
[6,2] 3-rep	3x	0.5	2
[4,2] RS	2x	1.0	2

[1] As a fraction of k units

MDS codes: Properties

- Let $G = [I \ P]$ be $(k \times n)$ generator matrix of $[n, k]$ code i.e.,

$$[p_1 \ p_2 \ \cdots \ p_{n-k}] = [m_1 \ \cdots \ m_k] \underbrace{\begin{bmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,n-k} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,n-k} \\ \vdots & \vdots & \ddots & \vdots \\ p_{k,1} & p_{k,2} & \cdots & p_{k,n-k} \end{bmatrix}}_{P}$$

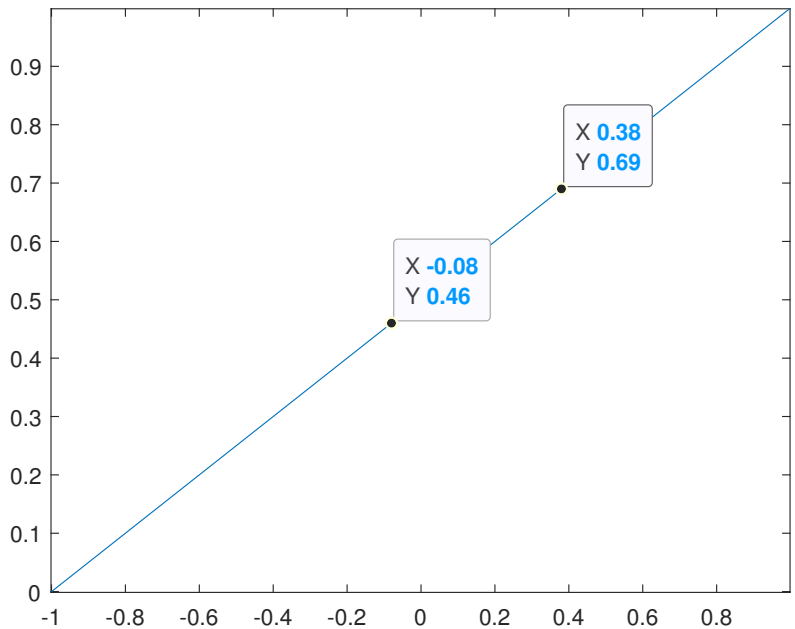
- Say there are $(n-k)$ erasures out of which m of them are message erasures given by $i_1, \cdots, i_m$ and available parities given by $j_1, \cdots, j_m$.

$$[p_{j_1} \ p_{j_2} \ \cdots \ p_{j_m}] = [m_{i_1} \ \cdots \ m_{i_m}] \underbrace{\begin{bmatrix} p_{i_1,j_1} & p_{i_1,j_2} & \cdots & p_{i_1,j_m} \\ p_{i_2,j_1} & p_{i_2,j_2} & \cdots & p_{i_2,j_m} \\ \vdots & \vdots & \ddots & \vdots \\ p_{i_m,j_1} & p_{i_m,j_2} & \cdots & p_{i_m,j_m} \end{bmatrix}}_{\text{sub-matrix of } P}$$

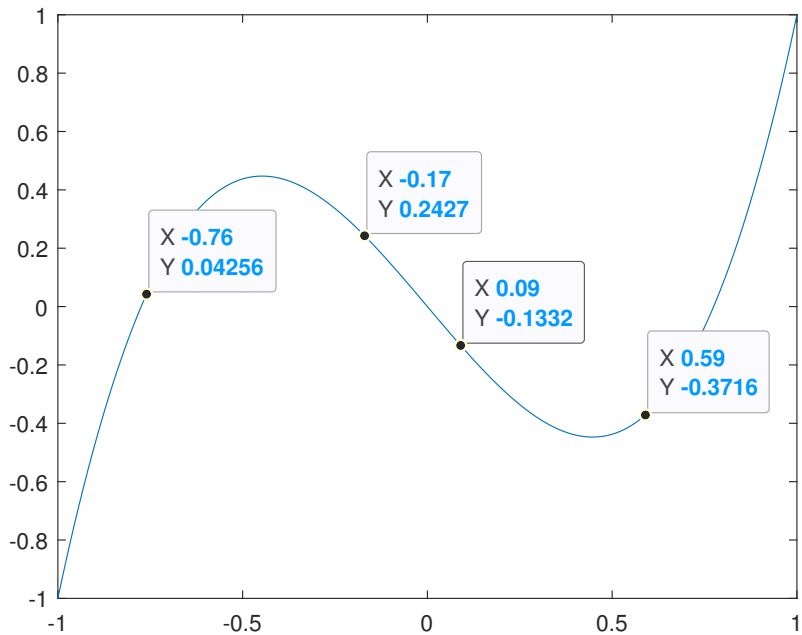
  - Code is MDS iff any $m \times m$ sub-matrix of P is invertible for all $m \leq \min(k, n-k)$

MDS Code: Reed Solomon Construction

- $[n, k]$ RS code.
  - A message of k symbols can be thought of as coefficients of a polynomial with degree $< k$.
  - Evaluate the polynomial at n distinct points. (message symbols need not be present directly in these n evaluations)



$k = 2$, the polynomial is a line. Two evaluations enough to recover a line



$k = 4$, the polynomial is of degree 3. Four evaluations enough to recover it

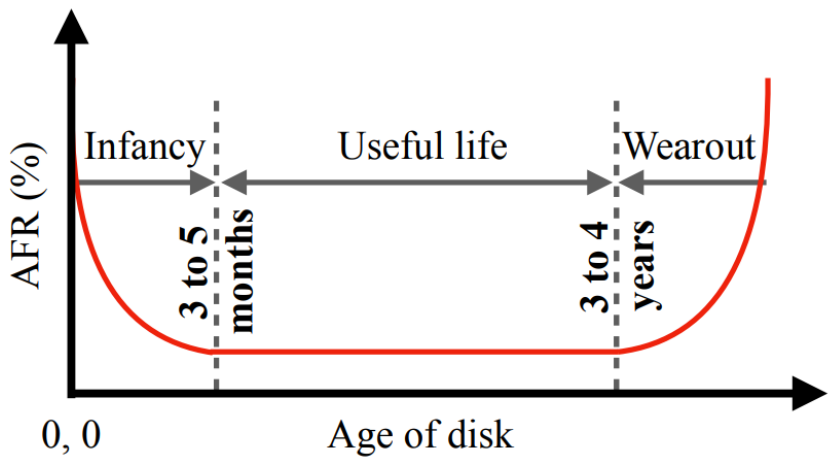
- If you have one evaluation of a line, then are infinitely many options for the lines that pass through the point.
- Need k evaluations to recover any polynomial of degree $< k$.

The Code Conversion Problem

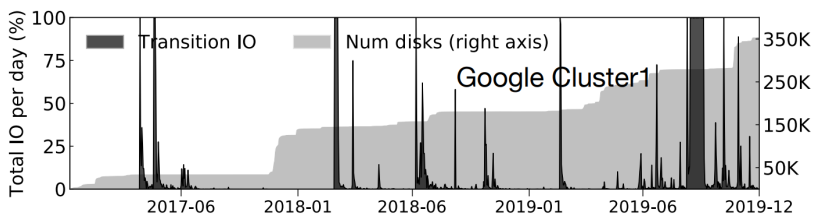
Joint work with

- Nikhil Krishnan (Assistant Professor, IIT Palakkad)
- Vinayak Ramkumar (Postdoc, TU Munich)
- Xiangliang Kong, (Postdoc, Tel Aviv University)

The Code Conversion Problem: Motivation



- As the disks age, it makes sense to increase/decrease overhead of the code to use the resources efficiently
- Typically conversions that do re-encoding incur high IO cost



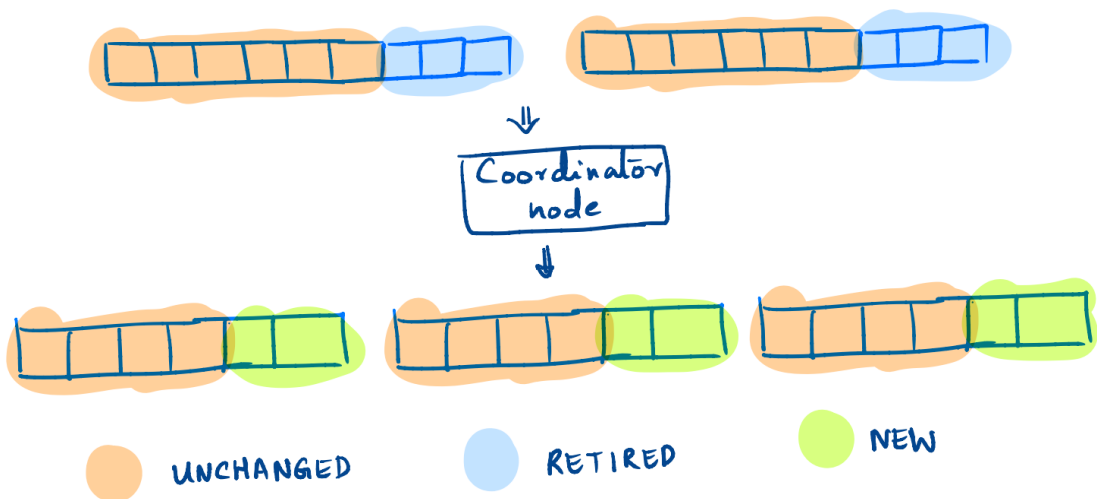
  - Weeks of 100 % cluster IO bandwidth spent on conversion

- Kadekodi et. al. "Cluster storage systems gotta have HeART: improving storage efficiency by exploiting disk-reliability heterogeneity". USENIX FAST, 2019.
- Kadekodi et. al, " PACEMAKER: Avoiding HeART attacks in storage clusters with disk-adaptive redundancy", USENIX OSDI 2020

'

Convertible Codes: Framework

- Convertible codes a framework introduced by Maturana and Rashmi [TIT 2022] to study the code conversion.
- Goal: To be able to effectively change from any $[n^I = k^I + r^I, k^I]$ initial code to a $[n^F = k^F + r^F, k^F]$ final code.
  - Multiple initial codewords converted to multiple final codewords



$$k^I = 6, k^F = 4, M = 12, \lambda^I = 2, \lambda^F = 3, r^I = 3, r^F = 2$$

  - For $M = \text{lcm}(k^I, k^F)$, $\lambda^I = \frac{M}{k^I}$ initial codewords get converted to $\lambda^F = \frac{M}{k^F}$ final codewords. Number of message symbols across initial and final codewords is M.

MDS Convertible Codes: Access cost

- MDS convertible codes are convertible codes where the initial and final codes are Maximum Distance Separable (MDS) codes.
- Access cost: number of symbols read from initial codewords to construct the final codewords plus the number of symbols written.
- We will focus on merge regime where $k^F = \lambda k^I$.
  - default approach: download λk^I symbols to construct the parity symbols of the final code.
  - Tight lower bound on access cost (Maturana and Rashmi, TIT 22)

$$\text{access cost} \geq \begin{cases} \lambda r^F + r^F & r^F \leq \min(r^I, k^I) \\ \lambda k^I + r^F & \text{otherwise} \end{cases}$$

  - Assume $r^F \leq \min(r^I, k^I)$ the non-trivial case
- Split regime when $k^I = \lambda^F k^F$

MDS Convertible Codes: An Example

- Let $G^I = [I_{k^I} \ P^I], G^F = [I_{k^F} \ P^F]$ be generator matrices of initial and final codes respectively. P^I, P^F are of sizes $k^I \times r^I$ and $\lambda k^I \times r^F$ respectively.
- Example: $k^I = 3, \lambda = 2, k^F = 6$ and $r^I = r^F = 2$

$$P^I = \begin{bmatrix} 1 & 1 \\ \theta_1 & \theta_2 \\ \theta_1^2 & \theta_2^2 \end{bmatrix}, P^F = \begin{bmatrix} 1 & 1 \\ \theta_1 & \theta_2 \\ \theta_1^2 & \theta_2^2 \\ \theta_1^3 & \theta_2^3 \\ \theta_1^4 & \theta_2^4 \\ \theta_1^5 & \theta_2^5 \end{bmatrix}$$

  - Let p_1^1, p_2^1 and p_1^2, p_2^2 be the parities from two initial codewords.

$$[p_1^j \ p_2^j] = [m_1^j \ m_2^j \ m_3^j] P_I \implies p_i^j = m_1^j + m_2^j \theta_i + m_3^j \theta_i^2$$

$$\left[p_1^F \ p_2^F\right] = [m_1^1 \ m_2^1 \ m_3^1 \ m_1^2 \ m_2^2 \ m_3^2] P_F$$

$$p_1^F = p_1^1 + \theta_1^3 p_1^2$$
$$p_2^F = p_2^1 + \theta_2^3 p_2^2$$

  - Can do conversion in this case by accessing $\lambda r^F = 4 < \lambda k^I = 6$ symbols

Access Optimality from Block-Reconstructable Property

$$P^F = \begin{bmatrix} P^{F,1} \\ P^{F,2} \\ \vdots \\ P_{F,\lambda} \end{bmatrix} \text{ where } P^{F,\ell} \text{ is } k^I \times r^F \text{ matrix.}$$

- P^F is said to be r^F-block reconstructable from P^I if for each $\ell \in [\lambda]$ there exists r^F columns of P^I that span the columns of $P^{F,\ell}$.
- MDS convertible code is access-optimal if P^F is r^F-block reconstructable from P^I
  - r^F final parities can be constructed by accessing exactly λr^I parity symbols.

Per-Symbol Access Optimality

- Recovery of each final parity symbol uses exactly λ initial parities with each parity belonging to an initial codeword.
- P^F is said to be parallel-block reconstructable from P^I if for each $\ell \in [\lambda]$ there exist r^F columns of P^I that are exactly equal to or scaling of columns seen in $P^{F,\ell}$.

parallel-block reconstructable $\rightarrow$ block-reconstructable

- Helps non-central conversion setting where the new node downloads the required data to reconstruct the corresponding final parity.

Earlier Approaches

1. Vandermonde construction

$$P^I = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \theta_1 & \theta_2 & \cdots & \theta_{r^I} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_1^{k^I-1} & \theta_2^{k^I-1} & \cdots & \theta_{r^I}^{k^I-1} \end{bmatrix}, P^F = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \theta_1 & \theta_2 & \cdots & \theta_{r^I} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_1^{\lambda k^I-1} & \theta_2^{\lambda k^I-1} & \cdots & \theta_{r^I}^{\lambda k^I-1} \end{bmatrix}$$

   - Need q to be large ($O(2^{(n^F)^3})$) to guarantee that P^F and P^I are super-regular

2. Hankel matrix based convertible codes
   - Super-regular property guaranteed for sub-matrices of Hankel matrices.
   - Constructions limited to parameters $r^F \leq r^I - \lambda + 1$ with linear field size for fixed number of parities

1	3	4	3	10	10	5	9	6	5	10
3	4	3	10	10	5	9	6	5	10	
4	3	10	10	5	9	6	5	10		
3	10	10	5	9	6	5	10			
10	10	5	9	6	5	10				
10	5	9	6	5	10					
5	9	6	5	10						
9	6	5	10							
6	5	10								
5	10									
10										

$k^I = 5$, $\lambda = 2$, $r^F = 2$, $r^I = 4$
Hankel array of size $n^F - 1 = 11$

Maturana and Rashmi, "Convertible Codes: New Class of Codes for Efficient Conversion of Coded Data in Distributed Storage", ITCS 2020

Earlier Approaches: Polynomial Construction

$$H_X = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ x_1 & x_2 & \cdots & x_m \\ x_1^2 & x_2^2 & \cdots & x_m^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{r^l-1} & x_2^{r^l-1} & \cdots & x_m^{r^l-1} \end{bmatrix}, \text{ where } X = \{x_1, \cdots, x_m\}$$

- Initial code: An (n^l, k^l) MDS code defined by parity check matrix:

$$H^l = [H_{X_1} \ H_Y] \implies -H_{X_1}\underline{m}^j = H_Y \underline{p}^j$$

- Final code: An (n^l, k^l) MDS code defined by parity check matrix:

$$H^F = [H_{X_1} \ H_{X_2} \ \cdots \ H_{X_\lambda} \ H_Y] \implies -\sum_{j=1}^{\lambda} H_{X_j}\underline{m}^j = H_Y \underline{p}^F$$

- Suppose $H_{X_j} = D_j H_{X_1}$ where D_j is $r^l \times r^l$ diagonal matrix

$$\underline{p}^F = -H_Y^{-1}\sum_{j=1}^{\lambda} H_{X_j}\underline{m}^j = -H_Y^{-1}\sum_{j=1}^{\lambda} D_j H_{X_1}\underline{m}^j = -H_Y^{-1}\sum_{j=1}^{\lambda} D_j \underline{p}^j$$

$\implies$ access-optimal MDS code. Pick $X_j = \gamma^j X_1$ such that X_i's, Y are disjoint.
Can add a column to H_Y such that $q \geq k^F + r^l - 1$ is sufficient

---

- Xiangliang Kong, "Locally Repairable Convertible Codes With Optimal Access Costs", IEEE Trans. in Info. Theory 2024.
- Krishnan et. al. "On Low Field Size MDS Convertible Codes", IEEE ISIT 2025

Our Approach

- P^I and P^F are Cauchy matrices.
- Cauchy matrix $C(X, Y)$ where $X = \{x_1, \cdots, x_k\}$, $Y = \{y_1, \cdots, y_r\}$:

$$C(X,Y) = \begin{bmatrix} \frac{1}{x_1 - y_1} & \frac{1}{x_1 - y_2} & \cdots & \frac{1}{x_1 - y_r} \\ \frac{1}{x_2 - y_1} & \frac{1}{x_2 - y_2} & \cdots & \frac{1}{x_2 - y_r} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{x_k - y_1} & \frac{1}{x_k - y_2} & \cdots & \frac{1}{x_k - y_r} \end{bmatrix}$$

- Cauchy matrices are super-regular.
- Goal: Design X, Y and X_1 of cardinalities k^F, r^I, k^I such that:

$$P^F = C(X, Y) \text{ is parallel-block reconstructable from } P^I = C(X_1, Y)$$

- Enough to look at $r^F = r^I = r$.
- For $r^F \leq r^I$, we do not convert $(r^I - r^F)$ nodes

Sub-Group Based Construction: Using Multiplicative Sub-Group of $\mathbb{F}_q^*$

- Let $\mathbb{F}_q$ be such that $r \mid (q-1)$ and $\nu = \frac{q-1}{r}$, $k' \leq \nu - 1$
- Let α be the primitive element of F_q and $\gamma = \alpha^\nu$.

$$Y = \{\gamma, \gamma^2, \cdots, \gamma^r = 1\}, \quad X_1 = \{\alpha, \alpha^2, \cdots, \alpha^{k'}\}, \quad X_\ell = \gamma^{\ell-1} X_1, X = \cup_{\ell=1}^{\lambda} X_\ell$$

- $X_1, \cdots, X_\lambda, Y$ are disjoint if $\lambda \leq r$
- $P^F = C(X, Y)$ is parallel-block reconstructable from $P' = C(X_1, Y)$

$$\begin{aligned} P^{F,\ell}(i,j) &= \frac{1}{\gamma^{\ell-1}\alpha^i - \gamma^j} = \frac{\gamma^{-(\ell-1)}}{\alpha^i - \gamma^{j-\ell+1}} \\ &= \gamma^{-(\ell-1)} P'(i,j') \end{aligned}$$

where $j' \in [r]$ such that $\gamma^{j'} = \gamma^{j-\ell+1}$.

  - j-th column of $P^{F,\ell}$ is scaling of j'-th column of P'

Sub-Group Based Construction: Using Multiplicative Sub-Group of $\mathbb{F}_q^*$

- Modification 1: $\lambda \leq (r-1)$
  - Let $\mathbb{F}_q$ be such that $(r-1) \mid (q-1)$ and $\nu = \frac{q-1}{r-1}$, $k^l \leq \nu - 1$, $\gamma = \alpha^\nu$.

$$Y = \{0, \gamma, \cdots, \gamma^{r-1} = 1\}$$

  - $P^F = C(X, Y)$ is parallel-block reconstructable from $P^l = C(X_1, Y)$
    - ★ $P^{F,\ell}(i,1) = \frac{1}{\gamma^{\ell-1}\alpha^i} = \gamma^{-(\ell-1)} P^l(i,1)$
- Modification 2: $\lambda \leq (r-2)$
  - Append an all-one column to Cauchy matrix is still super-regular matrix (Roth and G. Seroussi, TIT 85)
  - $\mathbb{F}_q$ be such that $(r-2) \mid (q-1)$ and $\nu = \frac{q-1}{r-2}$, $k^l \leq \nu - 1$, $\gamma = \alpha^\nu$.

$$Y = \{0, \gamma, \cdots, \gamma^{r-2} = 1\},$$

  - $P^F = \begin{bmatrix} 1 & C(X, Y) \end{bmatrix}$ is parallel-block reconstructable from $P^l = \begin{bmatrix} 1 & C(X_1, Y) \end{bmatrix}$

Sub-Group Based Construction: Using Multiplicative Sub-Group of $\mathbb{F}_q^*$

- $r^I = r^F = 4, k^I = 5, \lambda = 2 \implies k^F = 10, n^F = 14$
- Let $q = 13$ (meets the MDS conjecture $q = n^F - 1$)

$$Y = \{2^6 = 11, 2^{12} = 1, 0\}, X_1 = \{2, 2^2, 2^3, 2^4, 2^5\},$$
$$X_2 = \{2^6, 2^7, 2^8, 2^9, 2^{10}, 2^{11}\}$$

- If $\lambda = r - 2$ and $(r-2) \mid (q-1)$ for $q = (k^I + 1)(r-2) + 1 = n^F - 1$, then MDS conjecture is met.

$$P^I = \begin{bmatrix} 1 & 9 & 7 & 1 \\ 9 & 8 & 10 & 1 \\ 2 & 3 & 5 & 1 \\ 7 & 10 & 9 & 1 \\ 8 & 2 & 11 & 1 \\ 4 & 12 & 6 & 1 \\ 5 & 4 & 3 & 1 \\ 10 & 11 & 8 & 1 \\ 3 & 6 & 4 & 1 \\ 11 & 5 & 2 & 1 \end{bmatrix}$$

Sub-Group Based Construction: Additive Sub-Group of $\mathbb{F}_q$

- Let $q = p^m$ then $\mathbb{F}_q$ is isomorphic to $\{f(x) \in \mathbb{F}_p[x] \mid \deg(f(x)) < m\}$
- $r = p^u$ i.e., $r \mid q$ and $\nu = \frac{q}{r} = p^{m-u}$

$$\begin{aligned} Y &= \{f(x) \in \mathbb{F}_p[x] \mid \deg(f(x)) < u\} = \{y_1(x), \cdots, y_r(x)\} \\ X_1 &= \{f_1(x), \cdots, f_k(x)\} \\ &\subseteq \{x^u f(x) \mid f(x) \in \mathbb{F}_p[x], \deg(f(x)) < m-u-1\} \\ X_\ell &= y_\ell(x) + X_1 \end{aligned}$$

- $P^F = C(X, Y)$ is parallel-block reconstructable from $P^I = C(X_1, Y)$

$$\begin{aligned} P^{F,\ell}(i,j) &= \frac{1}{y_\ell(x) + f_i(x) - y_j(x)} = \frac{1}{f_i(x) - y_{j'}(x)} \\ &= P^I(i,j') \end{aligned}$$

  where $j' \in [r]$ such that $y_{j'}(x) = y_j(x) - y_\ell(x)$.
  - j-th column of $P^{F,\ell}$ is same as j'-th column of P^I
- Can similarly modify to add all one column.

Access Cost: Split regime

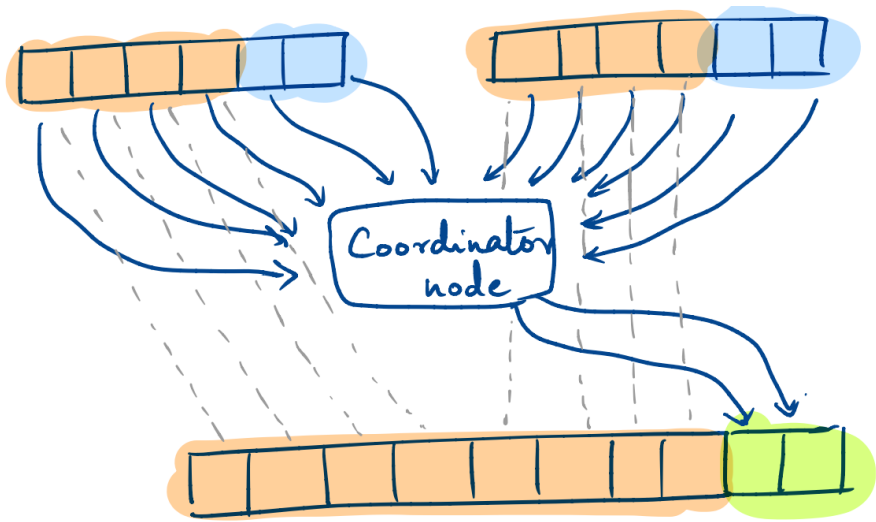
$$\text{access cost} \geq \begin{cases} (\lambda-1)k^F + \min(r^F, k^F) + r^F & r^I \geq r^F \\ \lambda k^F + r^F & \text{otherwise} \end{cases}$$

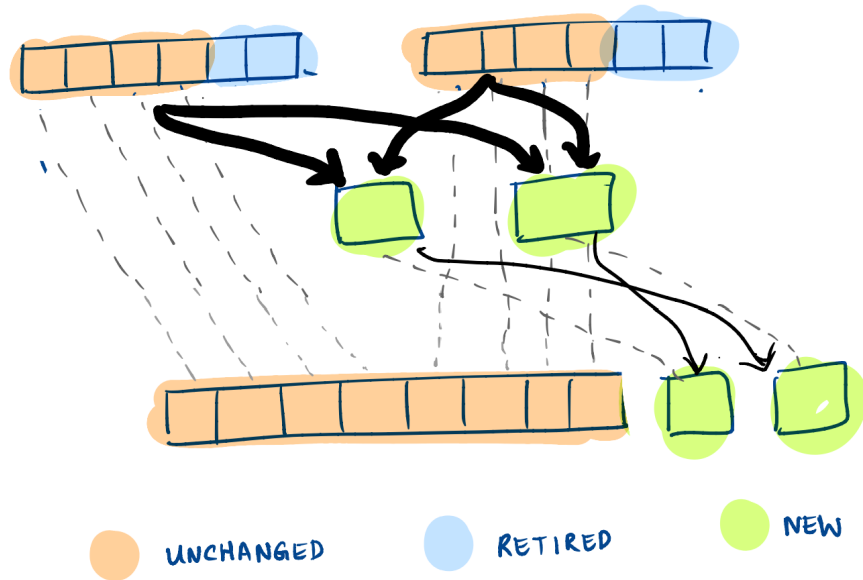
- $(n^I = k^I + r^F, k^I = \lambda k^F, n^F = k^F + r^F, k^F)$
- Consider any initial MDS code described by generator matrix $G^I = [I \ P^I]$
  - Let $G^F = [I \ P^F]$ be generator matrix of final code where $P^F = P^I(1 : k^F, 1 : r^F)$
  - Let $m^1, \cdots, m^\lambda$ be message symbols of the λ final codewords
  - Download $m^2, \cdots, m^\lambda$ and compute $m^2 p^F, \cdots, m^\lambda p^F$
  - Download the r^F initial parities given by:

$$[m_1 \ \cdots \ m_\lambda]P^I = m_1 P^F + \underbrace{[m_2 \ \cdots \ m_\lambda]}_{known} P^I(k^F + 1 : k^I, 1 : r^F)$$

recover $m_1 P^F$ from above.

Central vs Cooperative Conversion





$$\gamma_R \geq \begin{cases} \lambda\alpha(r^I + k^I(1 - \frac{r^I}{r^F})) & r^I \leq r^F \leq k^I \\ \lambda\alpha r^F & r^F \leq r^I \\ \lambda\alpha k^I & \text{otherwise} \end{cases}$$

- Write cost is $r^F\alpha$ in the central scheme
- Can be made to be much smaller in the cooperative scheme.

Maturana and Rashmi: "Bandwidth Cost of Code Conversions in Distributed Storage: Fundamental Limit and Optimal Constructions", Trans. in Info. Theory 2023

Bandwidth Cost of Code Conversion: Merge Regime

- α is the number of symbols stored at each node.
  - Example: $k^I = 4, r^I = 1, r^F = 3$. Two initial codewords shown below. Each code symbol is a vector with $\alpha = r^F = 3$ symbols in F_q. (Maturana and Rashmi, TIT 2023)

$m^1_{0,0}$	$m^1_{0,1}$	$m^1_{0,2}$
$m^1_{1,0}$	$m^1_{1,1}$	$m^1_{1,2}$
$m^1_{2,0}$	$m^1_{2,1}$	$m^1_{2,2}$
$m^1_{3,0}$	$m^1_{3,1}$	$m^1_{3,2}$
$m^1_0 p^{(1,0)}$	$m^1_1 p^{(1,0)} + m^1_0 p^{(1,1)}$	$m^1_2 p^{(1,0)} + m^1_0 p^{(1,2)}$

,

$m^2_{0,0}$	$m^2_{0,1}$	$m^2_{0,2}$
$m^2_{1,0}$	$m^2_{1,1}$	$m^2_{1,2}$
$m^2_{2,0}$	$m^2_{2,1}$	$m^2_{2,2}$
$m^2_{3,0}$	$m^2_{3,1}$	$m^2_{3,2}$
$m^2_0 p^{(1,0)}$	$m^2_1 p^{(1,0)} + m^2_0 p^{(1,1)}$	$m^2_2 p^{(1,0)} + m^2_0 p^{(1,2)}$

$m^1_{0,0}$	$m^1_{0,1}$	$m^1_{0,2}$
$m^1_{1,0}$	$m^1_{1,1}$	$m^1_{1,2}$
$m^1_{2,0}$	$m^1_{2,1}$	$m^1_{2,2}$
$m^1_{3,0}$	$m^1_{3,1}$	$m^1_{3,2}$
$m^2_{0,0}$	$m^2_{0,1}$	$m^2_{0,2}$
$m^2_{1,0}$	$m^2_{1,1}$	$m^2_{1,2}$
$m^2_{2,0}$	$m^2_{2,1}$	$m^2_{2,2}$
$m^2_{3,0}$	$m^2_{3,1}$	$m^2_{3,2}$
$m^1_0 p^{(1,0)} + m^2_0 p^{(2,0)}$	$m^1_1 p^{(1,0)} + m^2_1 p^{(2,0)}$	$m^1_2 p^{(1,0)} + m^2_2 p^{(2,0)}$
$m^1_0 p^{(1,1)} + m^2_0 p^{(2,1)}$	$m^1_1 p^{(1,1)} + m^2_1 p^{(2,1)}$	$m^1_2 p^{(1,1)} + m^2_2 p^{(2,1)}$
$m^1_0 p^{(1,2)} + m^2_0 p^{(2,2)}$	$m^1_1 p^{(1,2)} + m^2_1 p^{(2,2)}$	$m^1_2 p^{(1,2)} + m^2_2 p^{(2,2)}$

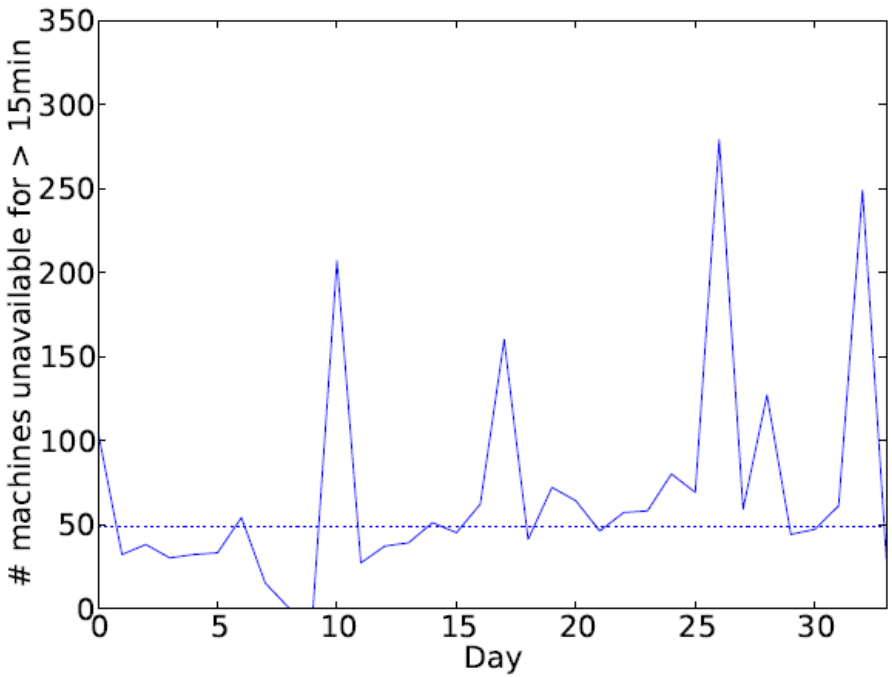
Maturana and Rashmi: "Bandwidth Cost of Code Conversions in Distributed Storage: Fundamental Limit and Optimal Constructions", Trans. in Info. Theory 2023

Conclusions and Open Questions

- Per-symbol access-optimal constructions for $\lambda \leq r$ with low field size.
  - Can this parameter restriction be removed without increasing the field size ?
- What are the fundamental bandwidth limits for code conversion in split regime ?
- What are the fundamental bandwidth limits of co-operative conversion ?
  - We have some preliminary schemes for the cooperative case that have smaller bandwidth requirements compared to central conversion bandwidth

The Node Repair Problem

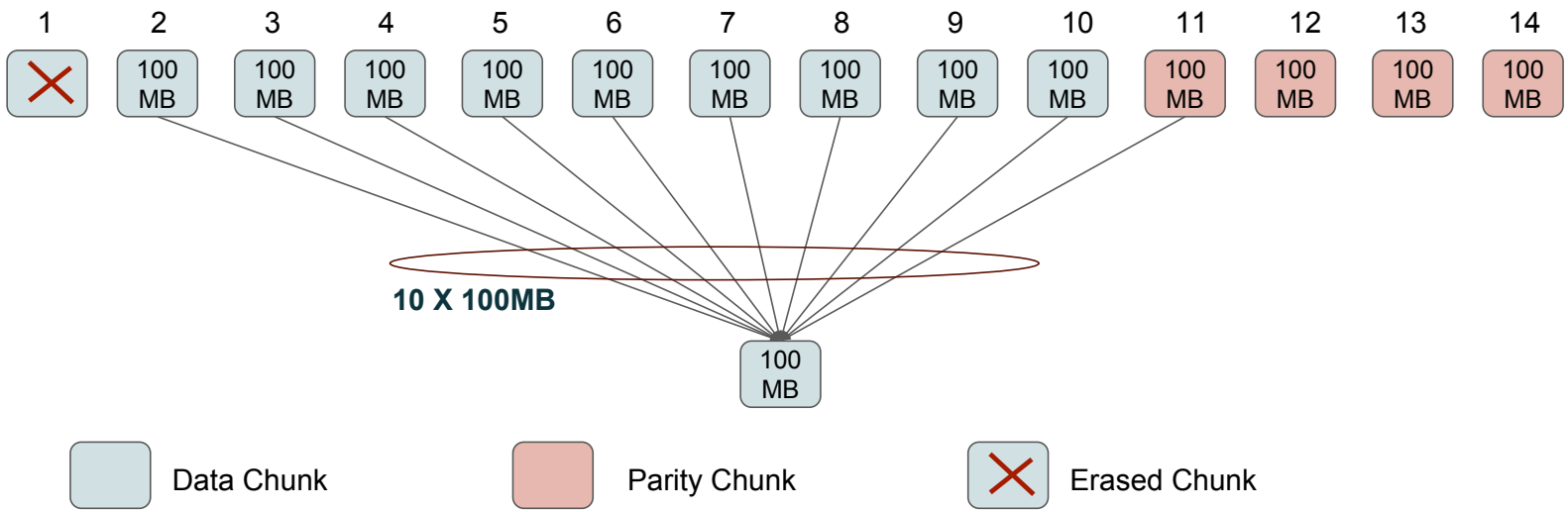
Erasure Codes and Node Failures



- A median of 50 nodes are unavailable per day.
- 98% of the failures are single node failures.
- A median of 180TB of network traffic per day is generated in order to reconstruct the RS coded data corresponding to unavailable machines.
- **Thus there is a strong need for erasure codes that can efficiently recover from single-node failures.**

Image courtesy: Rashmi et al.: "A Solution to the Network Challenges of Data Recovery in Erasure-coded Distributed Storage Systems: A Study on the Facebook Warehouse Cluster," USENIX Hotstorage, 2013.

Conventional Node Repair of an RS Code



In the example $(14, 10)$ RS code,

1. the amount of data downloaded to repair 100MB of data equals 1GB.

clearly, there is room for improvement...

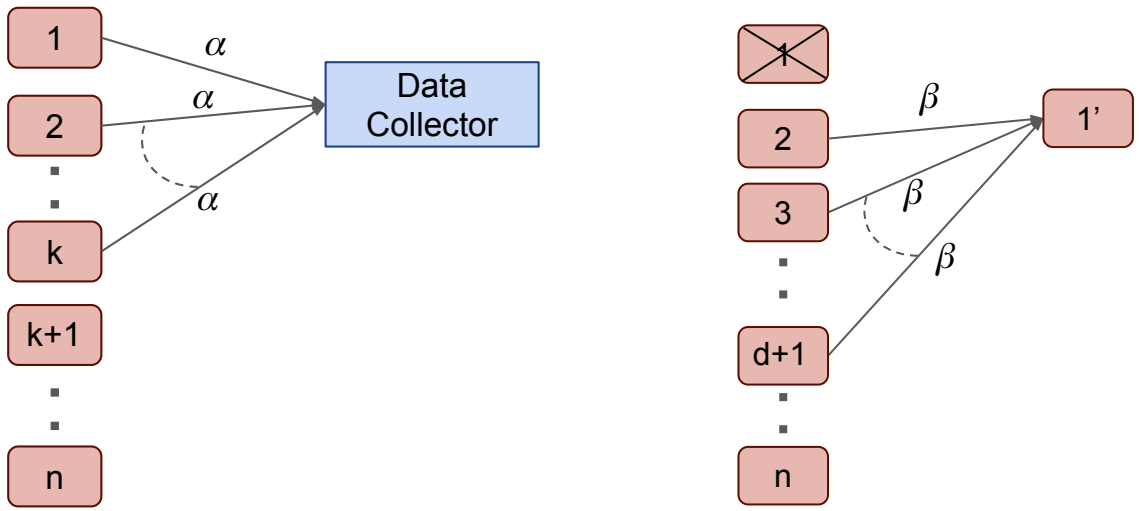
Things that matter.

- Low Storage Overhead. ✓
- High Tolerance for erasures. ✓
- Fast Repair (Single Node Failure). ??
  - Computation efficiency.
  - Speed in procuring repair data.
    - ★ low repair traffic.
    - ★ smaller disk read latencies.

Image source: sine.co, cartoonstock.com

Regenerating Codes

Parameters: ((n, k, d), (α, β), B, $\mathbb{F}_q$)



α capacity nodes

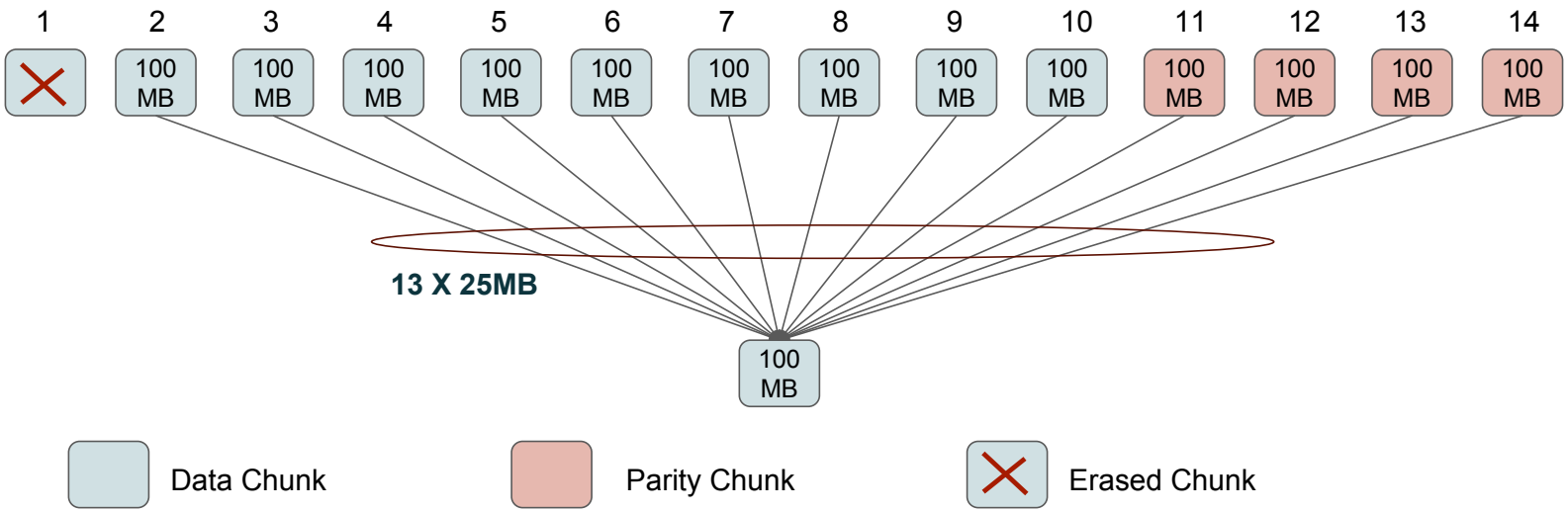
α capacity nodes

- Data (of size B) can be recovered by connecting to any k of n nodes
- A failed node can be repaired by connecting to any d nodes, downloading β symbols from each node; ($d\beta << $ file size B)

A. Dimakis, P. Godfrey, Y. Wu, M. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems", IEEE Transactions on Information Theory, 2010

Minimum Storage Regenerating Code

- MSR codes are MDS that have least possible repair bandwidth



1. Size of failed node's contents: 100MB
2. RS repair BW: 1 GB
3. MSR Repair BW: 325 MB

4-way Optimality of Clay code

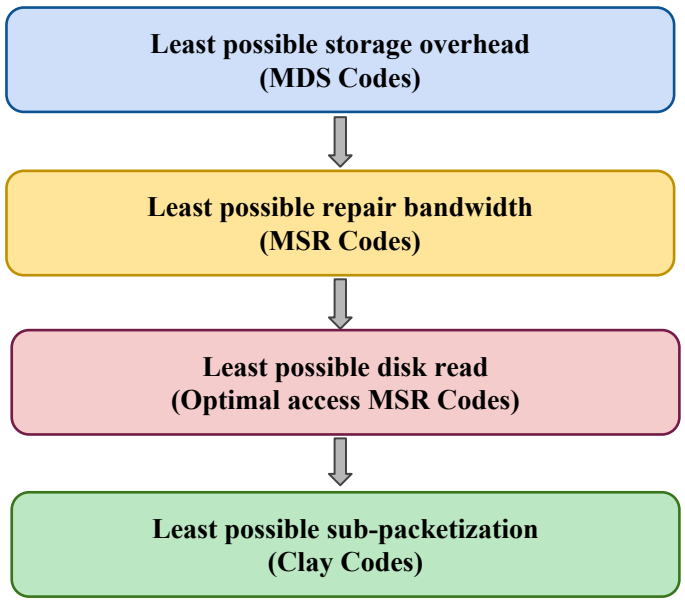


Image courtesy: denverpost.com

Vajha et. al., Clay Codes: Moulding MDS Codes to Yield an MSR Code, USENIX FAST 2018.

Ceph: Contributions

A popular opensource distributed storage system used by CERN, Flipkart, Cisco etc

"Us (+Vinayak) pitching Clay codes to Ceph in April 2017"

We have introduced Clay code as erasure code plugin. It is part of Ceph's Nautilus release (March 2019). As part of this we also introduced support for vector codes in Ceph.

Vajha et. al., Clay Codes: Moulding MDS Codes to Yield an MSR Code, USENIX FAST 2018.

Clay Code Summary

- The open-source implementation of Clay code in Ceph is for any (n, k, d) parameters.

- In comparison to $(20, 16)$ RS code, for Workloads with large sized objects (64MB), the Clay code $(20, 16, 19)$:
  - resulted in repair time reduction by $3X$.
  - Improved degraded read and write performance by 27.17% and 106.68% respectively.

Vajha et. al., Clay Codes: Moulding MDS Codes to Yield an MSR Code, USENIX FAST 2018.

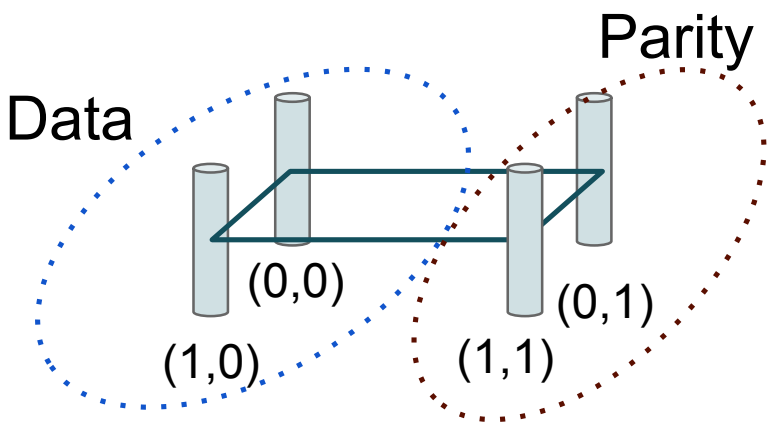
Open Questions

- Sub-packetization level prohibitive in realizing the benefits of MSR codes in practice.
- Given a fixed sub-packetization level, what is the minimal repair bandwidth that is required ?

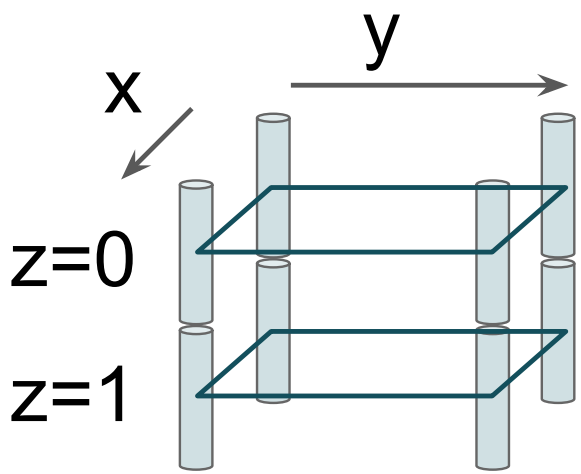
Appendix

Moulding an MDS Code to Yield the Clay Code

$(n = 4, k = 2)$ MDS code with optimal repair of systematic nodes, $\alpha = 2$



Code symbols of $[4, 2]$ MDS code.

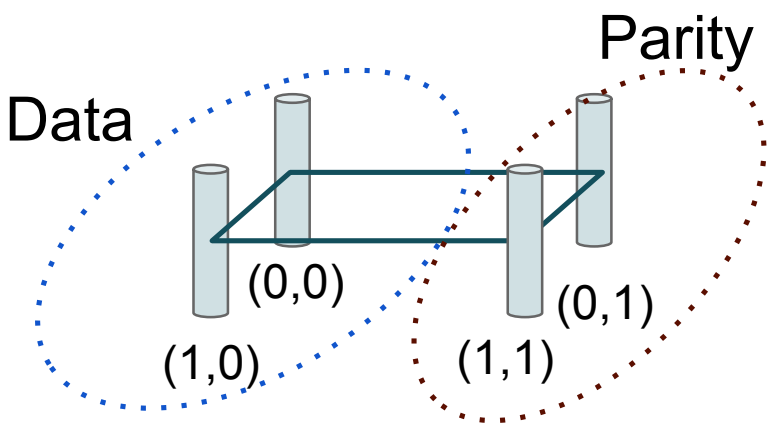


Layer two such units

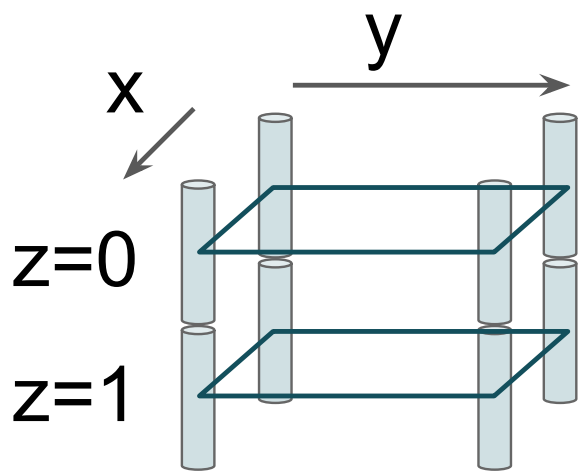
Uncoupled code still needs 4 symbols during recovery of single node (containing 2 symbols).

Moulding an MDS Code to Yield the Clay Code

$(n = 4, k = 2)$ MDS code with optimal repair of systematic nodes, $\alpha = 2$



Code symbols of $[4, 2]$ MDS code.

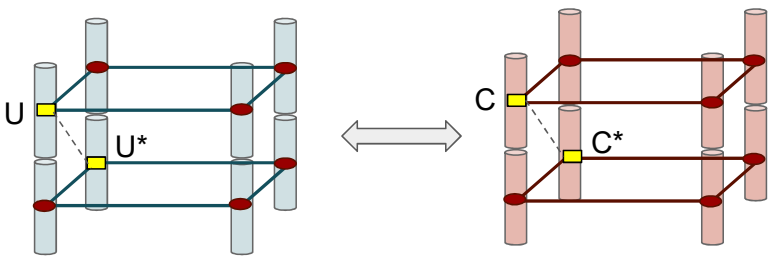


Layer two such units

Uncoupled code still needs 4 symbols during recovery of single node (containing 2 symbols).

Moulding an MDS Code to Yield the Clay Code

$(n = 4, k = 2)$ MDS code with optimal repair of systematic nodes, $\alpha = 2$

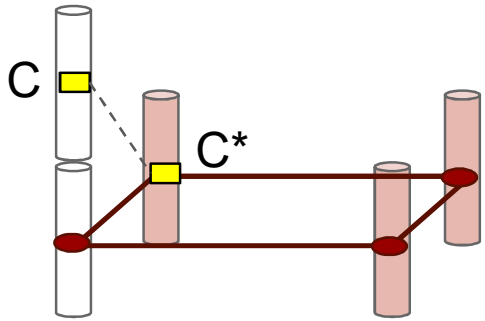


UnCoupled Code Coupled Code

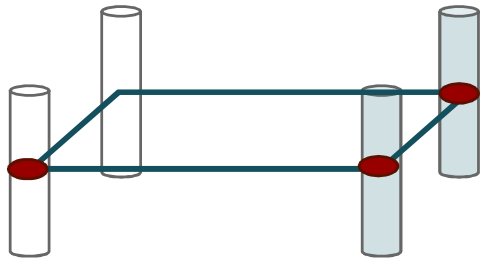
$$\begin{bmatrix} C \\ C^* \end{bmatrix} = \begin{bmatrix} 1 & \gamma \\ \gamma & 1 \end{bmatrix} \begin{bmatrix} U \\ U^* \end{bmatrix}$$

- Uncoupled code has 2 planes, where each plane corresponds to an $[4, 2]$ MDS code
- Coupled code symbols are obtained by:
  - Copying symbols with red dots
  - Pair of yellow symbols $\{C, C^*\}$ are obtained by transformation
- Note that recovery of any failed node in Uncoupled code requires 4 symbols

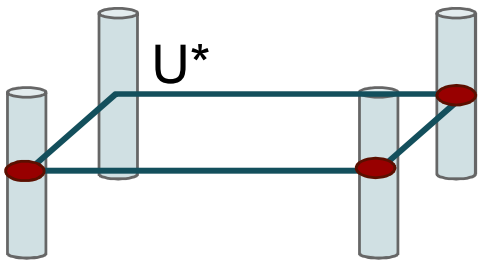
Repair from single node loss



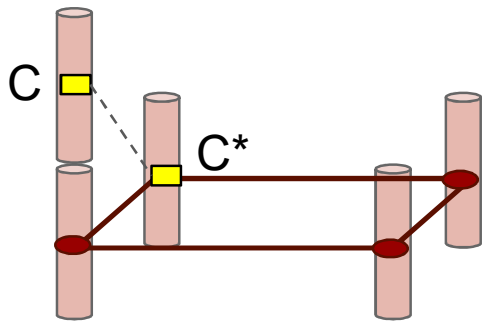
symbols that are available as part of helper information



symbols that are computable in uncoupled cube



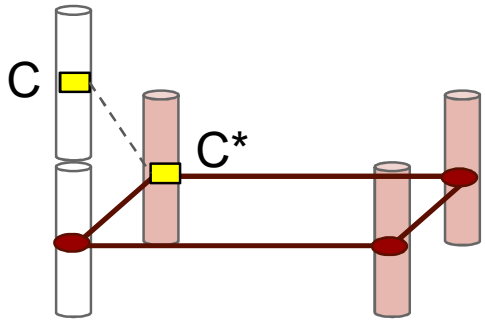
symbols recovered after using the $[4, 2]$ MDS code



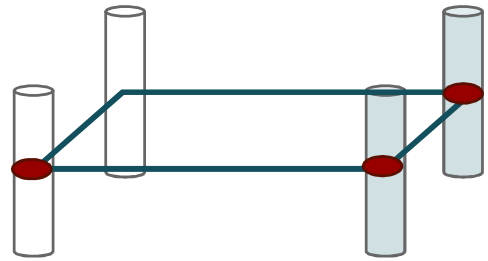
C recovered from C^*, U^*

- Used 3 symbols to recover the 2 symbols. For a file of size 1GB distributed across 2 500MB nodes. Need 750MB to recover the 500MB node instead of the whole 1GB.

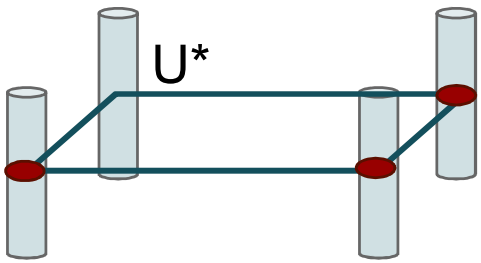
Repair from single node loss



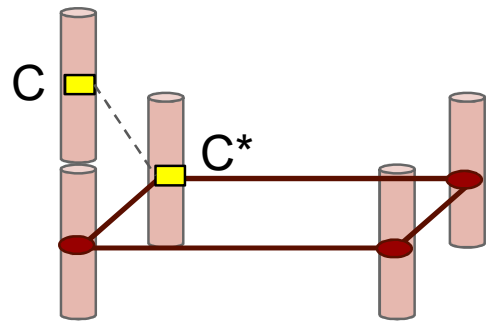
symbols that are available as part of helper information



symbols that are computable in uncoupled cube



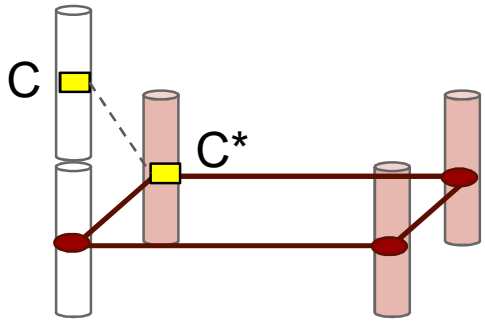
symbols recovered after using the $[4, 2]$ MDS code



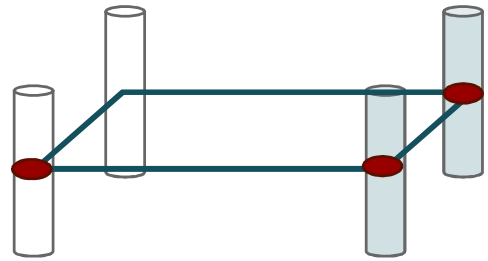
C recovered from C^*, U^*

- Used 3 symbols to recover the 2 symbols. For a file of size 1GB distributed across 2 500MB nodes. Need 750MB to recover the 500MB node instead of the whole 1GB.

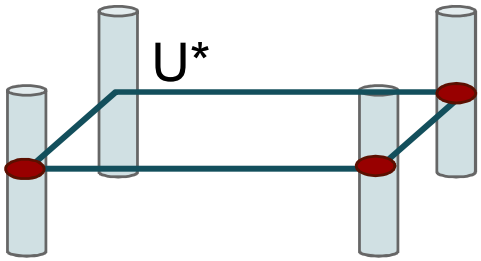
Repair from single node loss



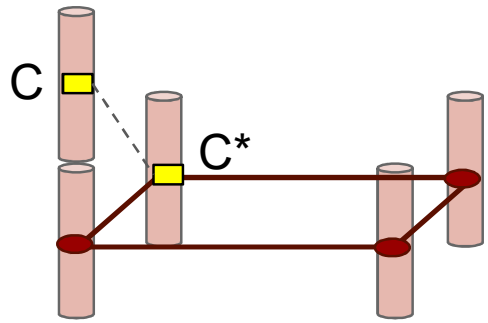
symbols that are available as part of helper information



symbols that are computable in uncoupled cube



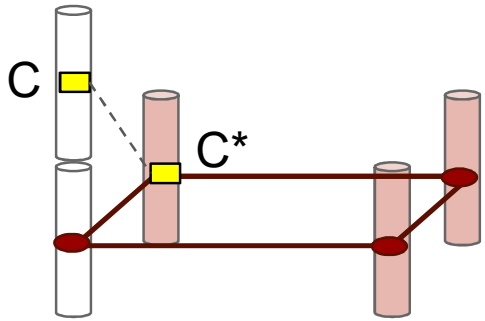
symbols recovered after using the $[4, 2]$ MDS code



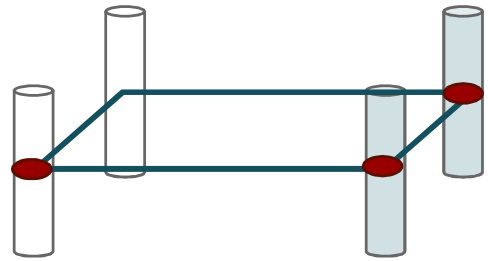
C recovered from C^*, U^*

- Used 3 symbols to recover the 2 symbols. For a file of size 1GB distributed across 2 500MB nodes. Need 750MB to recover the 500MB node instead of the whole 1GB.

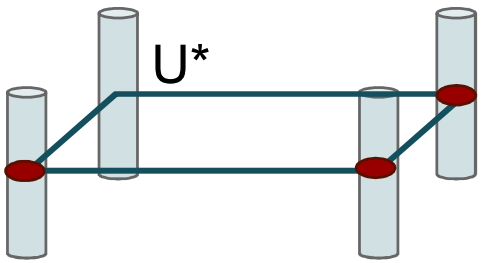
Repair from single node loss



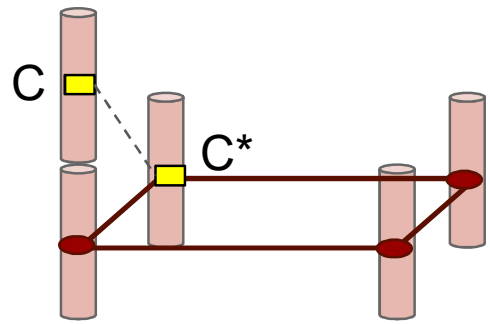
symbols that are available as part of helper information



symbols that are computable in uncoupled cube



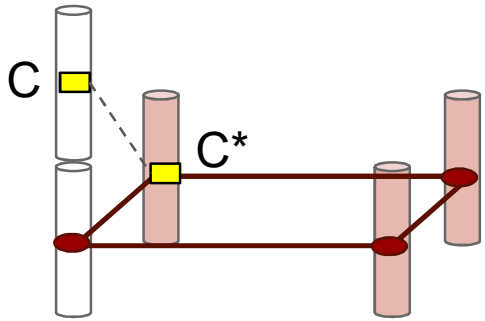
symbols recovered after using the $[4, 2]$ MDS code



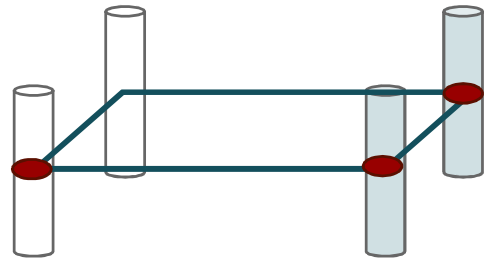
C recovered from C^*, U^*

- Used 3 symbols to recover the 2 symbols. For a file of size 1GB distributed across 2 500MB nodes. Need 750MB to recover the 500MB node instead of the whole 1GB.

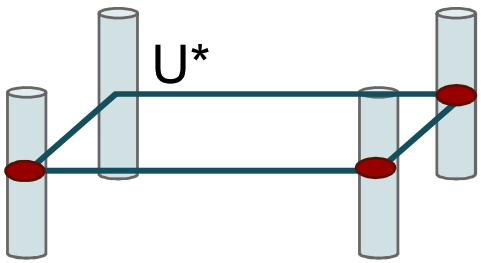
Repair from single node loss



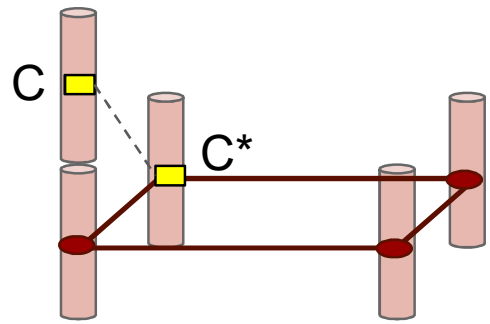
symbols that are available as part of helper information



symbols that are computable in uncoupled cube



symbols recovered after using the $[4, 2]$ MDS code

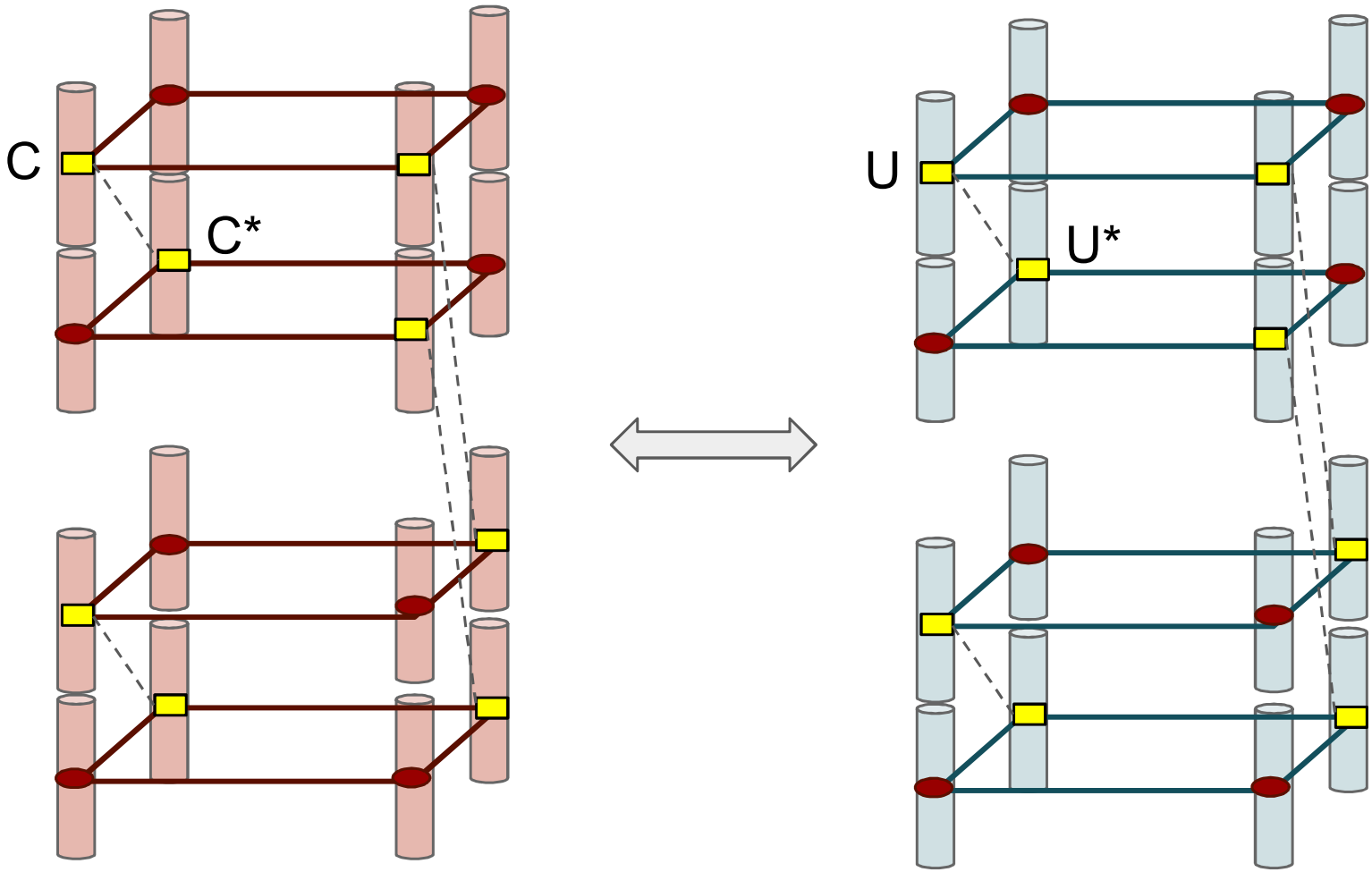


C recovered from C^*, U^*

- Used 3 symbols to recover the 2 symbols. For a file of size 1GB distributed across 2 500MB nodes. Need 750MB to recover the 500MB node instead of the whole 1GB.

Clay Code

$(n = 4, k = 2, d = 3)$ MSR code with all node optimal repair



Coupled Code

Uncoupled Code

- The same construction extends to any (n, k, d)