Fundamental Limits and Bounds for Distributed Data Storage Networks

Ali Tebbi[†] Terence Chan[†] Chi Wan Sung [‡]

[†]Institute for Telecommunications Research, University of South Australia

[‡]Department of Electronic Engineering, City University of Hong Kong

March 15, 2017

University of South Australia | Institute for Telecommunications Research

Outline

Introduction

- Erasure coding techniques are used recently in storage networks (HDFS, Windows Azure, ...).
- Coding techniques offer high reliability and low storage overhead than the conventional replication. [1]

The high network traffic and I/O overhead in repair process is a major performance bottleneck of storage networks.

Code locality is a metric to measure the number of nodes involved in the repairing process of a failed node. [2]

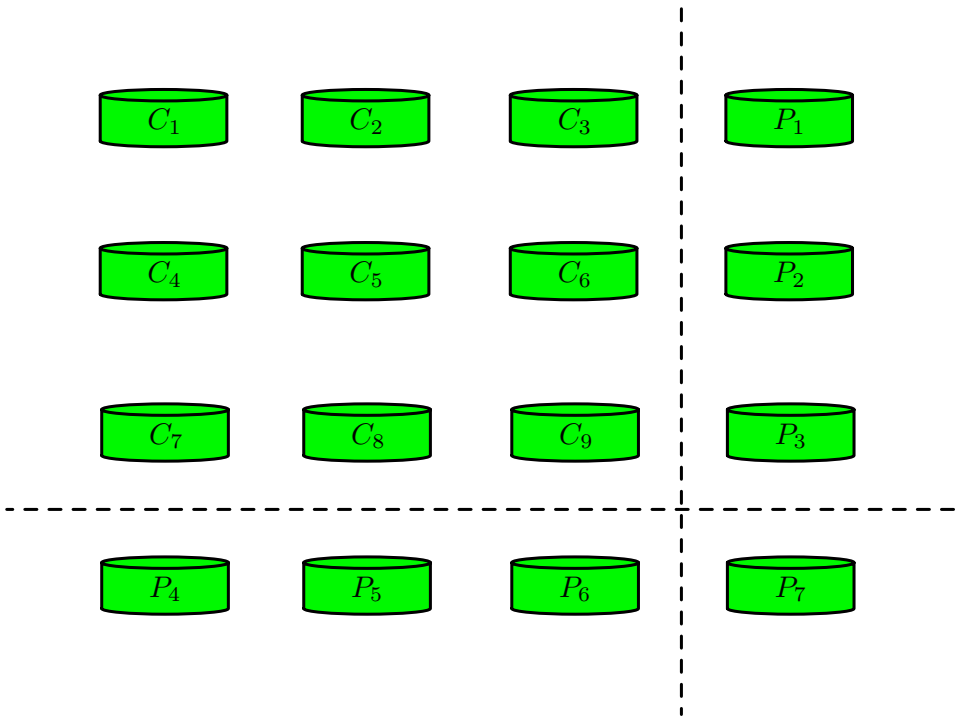
---

[1] H. Weatherspoon and J. D. Kubiatowicz, "Erasure coding vs. replication: A quantitative comparison," in *Proc. Int. Workshop Peer-to-Peer Syst.*, 2002.

[2] P. Gopalan, C. Huang, H. Simitci, and S. Yekhanin, "On the locality of codeword symbols," *IEEE Trans. Inf. Theory*, vol. 58, no. 11, pp. 6925-6934, Nov. 2012.

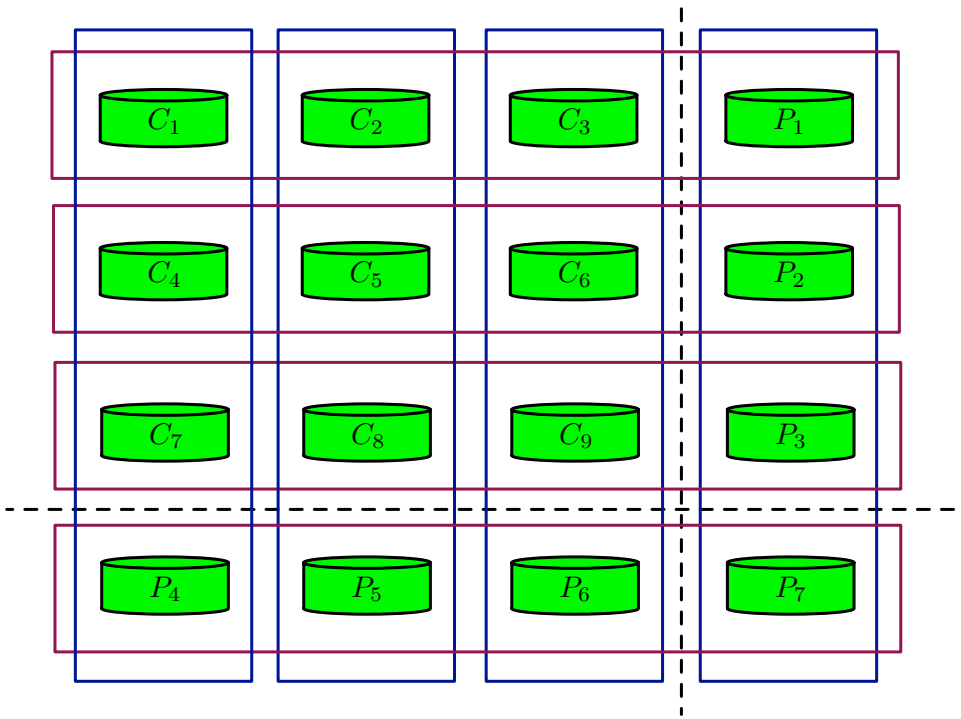
Robust Locally Repairable Codes

- A binary linear storage code of length $n = 16$ with $k = 9$.
- The minimum distance of the code is 4.
- Each node has two locally repair groups of size $r = 3$.



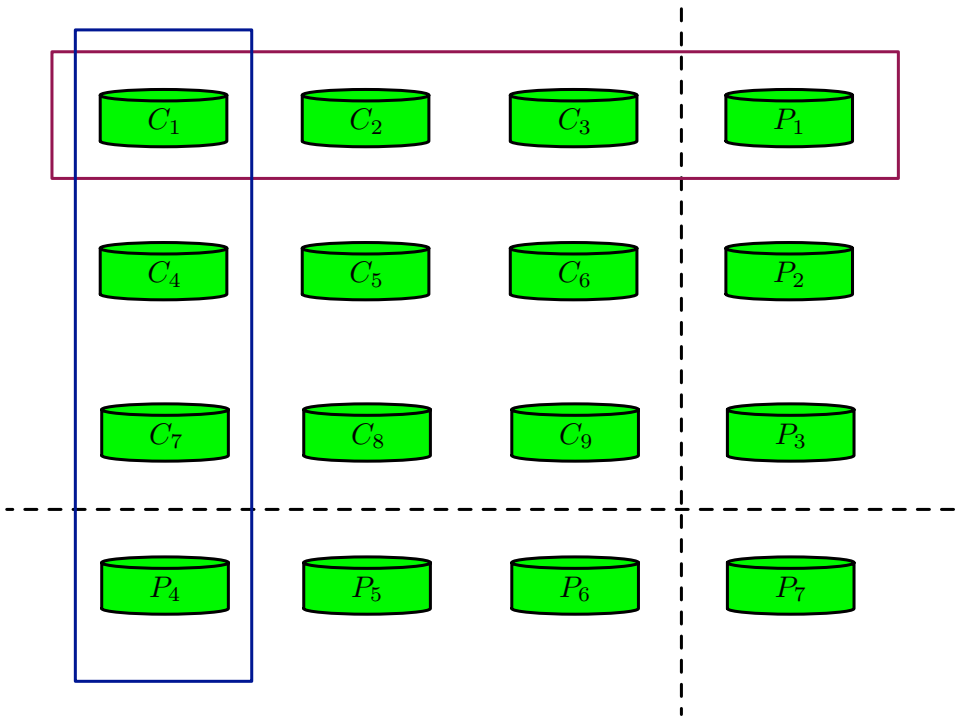
Robust Locally Repairable Codes

- A binary linear storage code of length $n = 16$ with $k = 9$.
- The minimum distance of the code is 4.
- Each node has two locally repair groups of size $r = 3$.



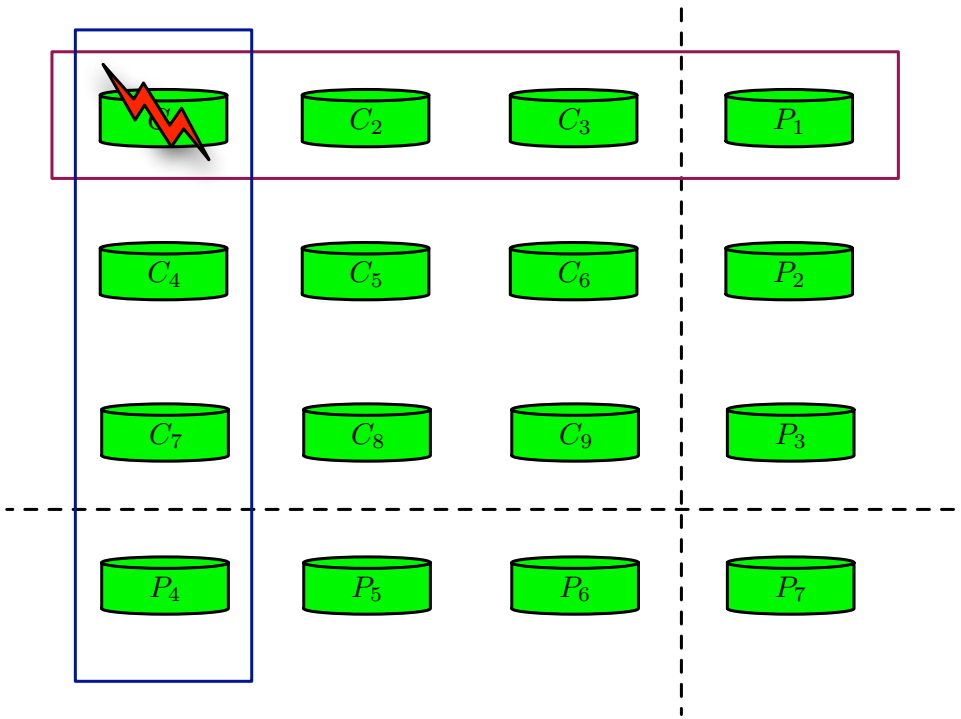
Robust Locally Repairable Codes

- A binary linear storage code of length $n = 16$ with $k = 9$.
- The minimum distance of the code is 4.
- Each node has two locally repair groups of size $r = 3$.



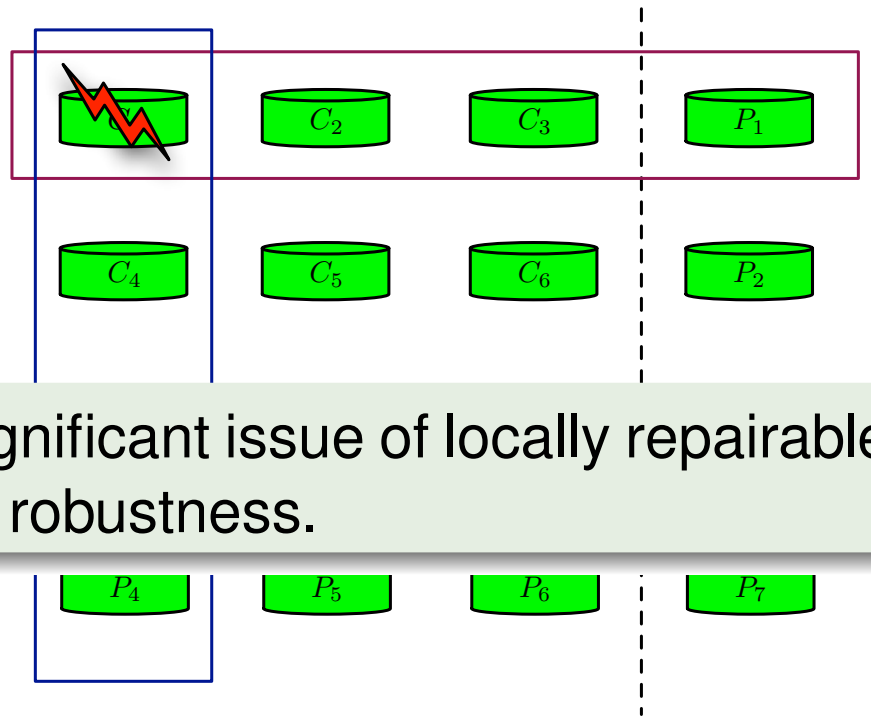
Robust Locally Repairable Codes

- A binary linear storage code of length $n = 16$ with $k = 9$.
- The minimum distance of the code is 4.
- Each node has two locally repair groups of size $r = 3$.



Robust Locally Repairable Codes

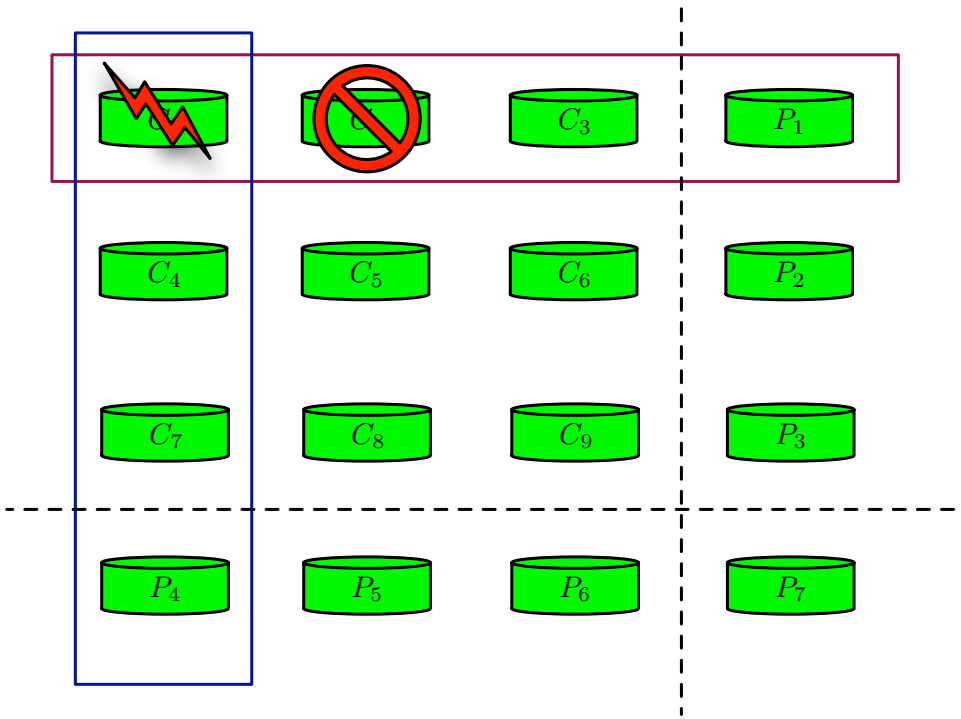
- A binary linear storage code of length $n = 16$ with $k = 9$.
- The minimum distance of the code is 4.
- Each node has two locally repair groups of size $r = 3$.



One significant issue of locally repairable codes is their robustness.

Robust Locally Repairable Codes

- A binary linear storage code of length $n = 16$ with $k = 9$.
- The minimum distance of the code is 4.
- Each node has two locally repair groups of size $r = 3$.



Robust Locally Repairable Codes

- An $(r, \beta, \Gamma, \zeta)$ robust locally repairable code is a linear code satisfying the following criteria:

Robust Locally Repairable Codes

1. **Robust Local Recovery (RLR).** For any failed node in the network in the presence of any extra Γ failures, there exist ζ different repair groups with size of at most r.

2. **Global Recovery (GR).** Any β simultaneous failure can be repaired by other survived nodes.

T. H. Chan, M. A. Tebbi, and C. W. Sung, "Linear programming bounds for storage codes," *in 9th International Conference on Information, Communication, and Signal Processing (ICICS 2013)*, Dec. 2013.

Robust Locally Repairable Codes

- For a linear code $\mathcal{C}$, the parity check matrix H is an $(n-k, n)$ matrix such that

$$GH^{\top} = \mathbf{0}$$

- The dual code $\mathcal{C}^{\perp}$ is generated by the all row spans of the parity matrix H such that

$$\mathcal{C}^{\perp} = \left\{ \mathbf{h} \in \mathbb{F}_q^n : \mathbf{h}\mathbf{c}^{\top} = 0 \text{ for all } \mathbf{c} \in \mathcal{C} \right\}$$

$$c_i = -h_i^{-1} \sum_{j \in \lambda(h) \setminus i} c_j h_j$$

L. Pamies-Juarez, H. D. Hollmann, and F. Oggier, "Locally repairable codes with multiple repair alternatives," in *proc. IEEE Int. Symp. Information Theory*, 2013, pp. 892-896.

Robust Locally Repairable Codes

- An $(r, \beta, \Gamma, \zeta)$ robust locally repairable code is a linear code satisfying the following criteria:

Robust Locally Repairable Codes

1. **Robust Local Recovery (RLR).** for any $i \in \mathcal{N}$ and $\gamma \subset \mathcal{N} \setminus i$ such that $|\gamma| = \Gamma$, there exists $\mathbf{h}_1, \ldots, \mathbf{h}_\zeta \in \mathcal{C}^\perp$ such that for all $j = 1, \ldots, \zeta$,
    1. $i \in \lambda(\mathbf{h}_j)$, $\gamma \cap \lambda(\mathbf{h}_j) = \emptyset$, and $|\lambda(\mathbf{h}_j)| - 1 \leq r$.
    2. $\lambda(\mathbf{h}_j) \neq \lambda(\mathbf{h}_k)$ for $k \neq j$.
2. **Global Recovery (GR).** $A_{\mathbf{w}} = 0$, for all $\mathbf{w} \subseteq \mathcal{N}$ such that $1 \leq |\mathbf{w}| \leq \beta$.

M. A. Tebbi, T. H. Chan, and C. Sung, "Linear programming bounds for robust locally repairable storage codes," *in proc. Information Theory Workshop (ITW), 2014 IEEE*, Nov. 2014, pp. 50-54.

Linear Programming Upper Bounds

- Consider any $(r, \beta, \Gamma, \zeta)$ robust locally repairable code $\mathcal{C}$. Then, $|\mathcal{C}|$ is upper bounded by the optimal value of the following optimisation problem.

$$
\begin{aligned}
&\textbf{maximize} && \textstyle\sum_{\mathbf{w} \subseteq \mathcal{N}} A_{\mathbf{w}} && \\
&\textbf{subject to} && && \\
&\quad A_{\mathbf{w}} \geq 0 && && \forall \mathbf{w} \subseteq \mathcal{N} \\
&\quad B_{\mathbf{w}} = \frac{\sum_{\mathbf{s} \subseteq \mathcal{N}} A_{\mathbf{s}} \prod_{j=1}^{n} \kappa(s_j, w_j)}{\sum_{\mathbf{s} \subseteq \mathcal{N}} A_{\mathbf{s}}} && && \forall \mathbf{w} \subseteq \mathcal{N} \\
&\quad B_{\mathbf{w}} \geq 0 && && \forall \mathbf{w} \subseteq \mathcal{N} \\
&\quad A_{\mathbf{w}} = 0 && && 1 \leq |\mathbf{w}| \leq \beta \\
&\quad A_{\varnothing} = 1 && && \\
&\quad \sum_{\mathbf{s} \in \Omega_i : \gamma \cap \mathbf{s} = \varnothing} B_{\mathbf{s}} \geq \zeta(q-1) && && \forall i \in \mathcal{N}, \gamma \in \Delta_i
\end{aligned}
$$

Linear Programming Upper Bounds

Robust locally repairing constraint

$$\sum_{\mathbf{s}\in\Omega_i:\gamma\cap\mathbf{s}=\emptyset} B_{\mathbf{s}} \geq \zeta(q-1) \qquad \forall i \in \mathcal{N}, \gamma \in \Delta_i$$

Ω_i is the collection of all subsets of $\mathcal{N}$ that contains i and of size at most $r+1$ and Δ_i is the collection of all subsets of $\mathcal{N} \setminus i$ of size at most Γ.

For a linear code $\mathcal{C}$ over $\mathbb{F}_q$, if $\mathbf{c} \in \mathcal{C}$, then $a\mathbf{c} \in \mathcal{C}$ for all $a \in \mathbb{F}_q$ and $a \neq 0$. Therefore, except for the zero codeword, there exists at least $q-1$ codewords which have the same support.

Linear Programming Upper Bounds

Linear optimisation problem

maximize $\sum_{\mathbf{w}\subseteq\mathcal{N}} A_{\mathbf{w}}$

subject to

$$A_{\mathbf{w}} \geq 0 \qquad \forall \mathbf{w} \subseteq \mathcal{N}$$

$$\sum_{\mathbf{s}\subseteq\mathcal{N}} A_{\mathbf{s}} \prod_{j=1}^{n} \kappa(s_j, w_j) \geq 0 \qquad \forall \mathbf{w} \subseteq \mathcal{N}$$

$$A_{\mathbf{w}} = 0 \qquad 1 \leq |\mathbf{w}| \leq \beta$$

$$A_{\varnothing} = 1$$

$$\sum_{\mathbf{w}\in\Omega_i:\gamma\cap\mathbf{w}=\emptyset} \left(\sum_{\mathbf{s}\subseteq\mathcal{N}} A_{\mathbf{s}} \prod_{j=1}^{n} \kappa(s_j, w_j) \right) \geq \zeta(q-1) \sum_{\mathbf{w}\subseteq\mathcal{N}} A_{\mathbf{w}} \qquad \forall i \in \mathcal{N}, \gamma \in \Delta_i.$$

The complexity of the linear programming problem will increase exponentially with the number of storage nodes n.

Linear Programming Upper Bounds

- The complexity of the linear programming problem can be reduced by exploiting the symmetries in the problem.

Proposition

Suppose $(a_{\mathbf{w}} : \mathbf{w} \subseteq \mathcal{N})$ satisfies the constraint in the optimisation problem. Then, $a^{\sigma}_{\mathbf{w}}$ for any $\sigma \in S_{\mathcal{N}}$ also satisfies the constraints.

Corollary

it is sufficient to consider only "symmetric" feasible solution. Then, we can impose additional constraint

$$A_{\mathbf{w}} = A_{\mathbf{s}}, \quad \forall |\mathbf{w}| = |\mathbf{s}|.$$

without affecting the bound.

Linear Programming Upper Bounds

- Consider a $(r, \beta, \Gamma, \zeta)$ robust locally repairable code $\mathcal{C}$. Then, $|\mathcal{C}|$ is upper bounded by the optimal value in the following maximisation problem

Theorem

$$\begin{aligned}
&\textbf{maximize} \quad \textstyle\sum_{t=0}^{n} \binom{n}{t} a_t \\
&\textbf{subject to} \\
&a_t \geq 0, \quad \forall t = 0, \ldots, n \\
&b_t = \textstyle\sum_{i=0}^{t} \sum_{j=0}^{n-t} \binom{t}{i} \binom{n-t}{j} a_{i+j} (-1)^i (q-1)^{t-i} \\
&\qquad\qquad \forall t = 0, \ldots, n \\
&b_t \geq 0, \quad \forall t = 0, \ldots, n \\
&\textstyle\sum_{t=1}^{\beta} a_t = 0 \\
&a_0 = 1 \\
&\displaystyle\sum_{t=1}^{r} \binom{n-1-\Gamma}{t} b_{t+1} \geq \zeta(q-1) \sum_{t=0}^{n} \binom{n}{t} a_t.
\end{aligned}$$

Examples

- The code achieves the bound at the point $(r = 3, k = 9)$.
- This code is an optimum $(3, 3, 1, 1)$, $(3, 3, 0, 2)$ robust locally repairable code.

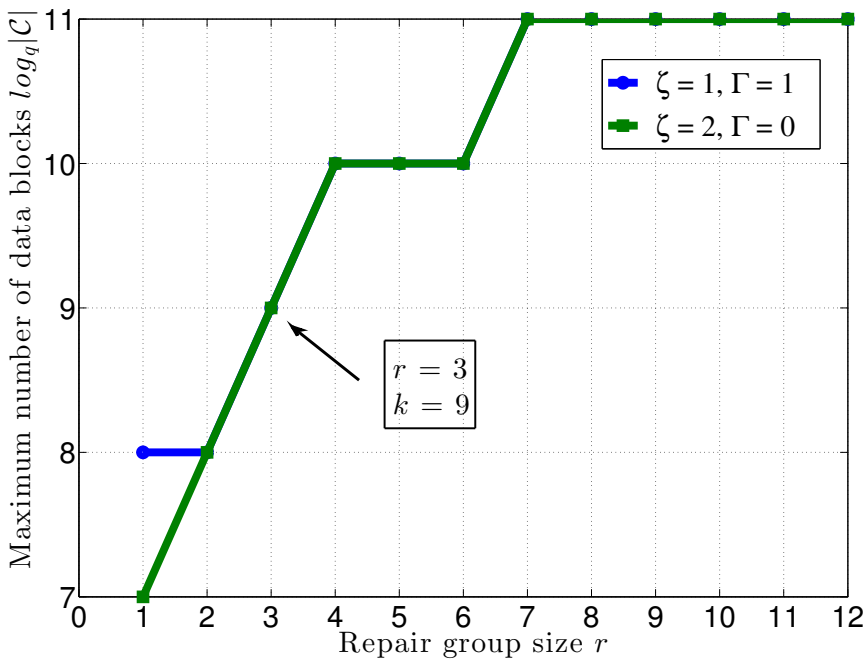
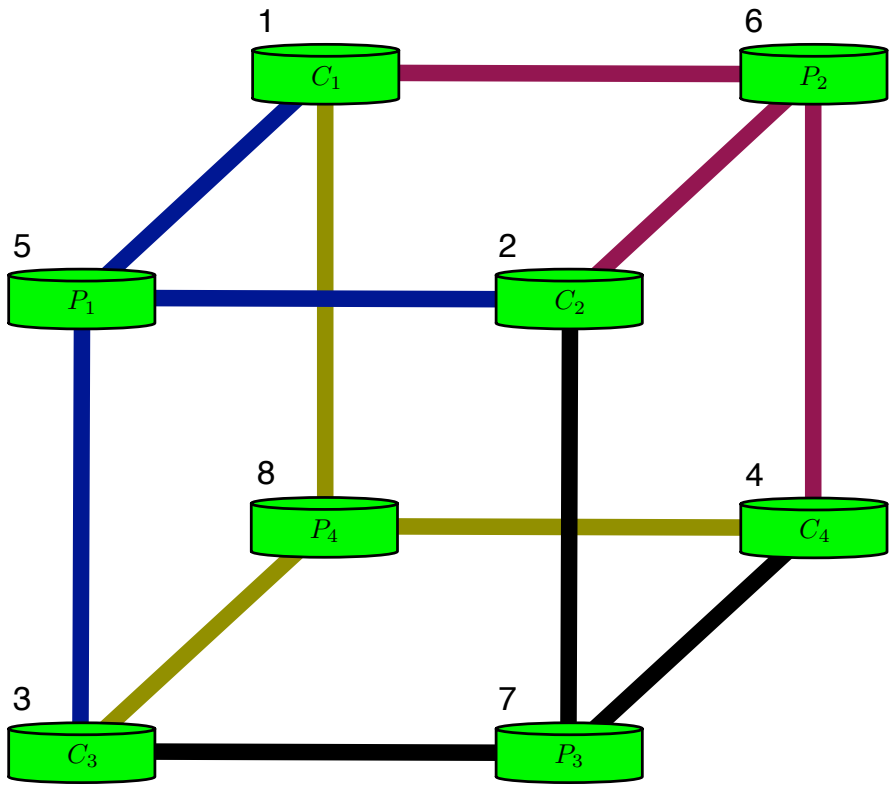


Figure: Upper bounds of $(r, 3, \Gamma, \zeta)$ linear storage code with $n = 16$.

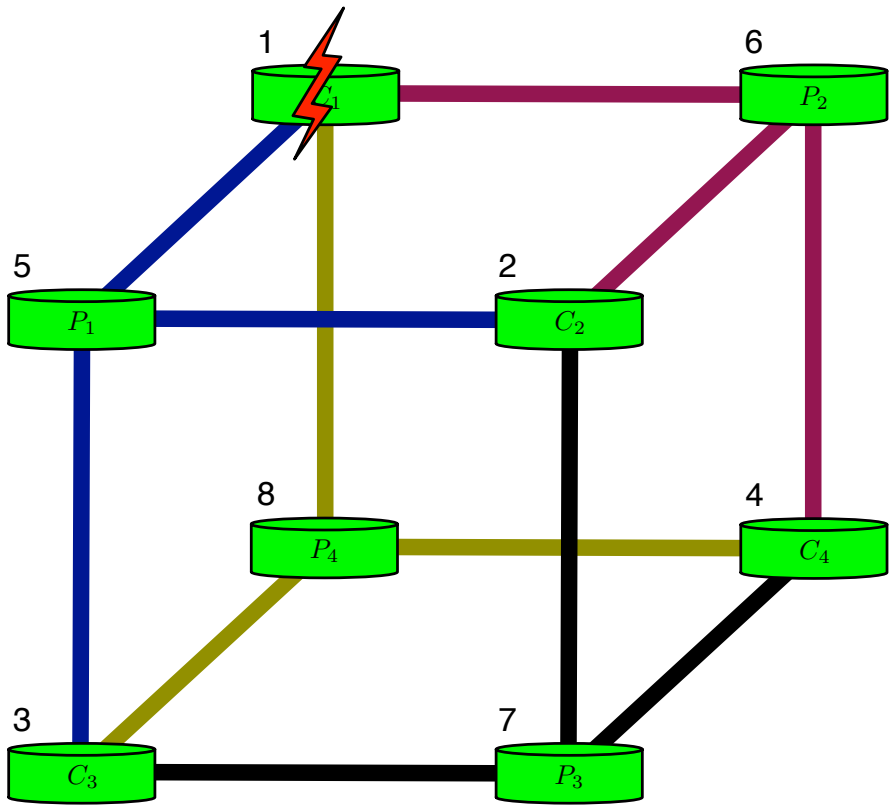
Examples

- A $(8, 4)$ binary linear storage code.
- Any failed node has $\zeta = 7$ different repair groups of size $r = 3$.



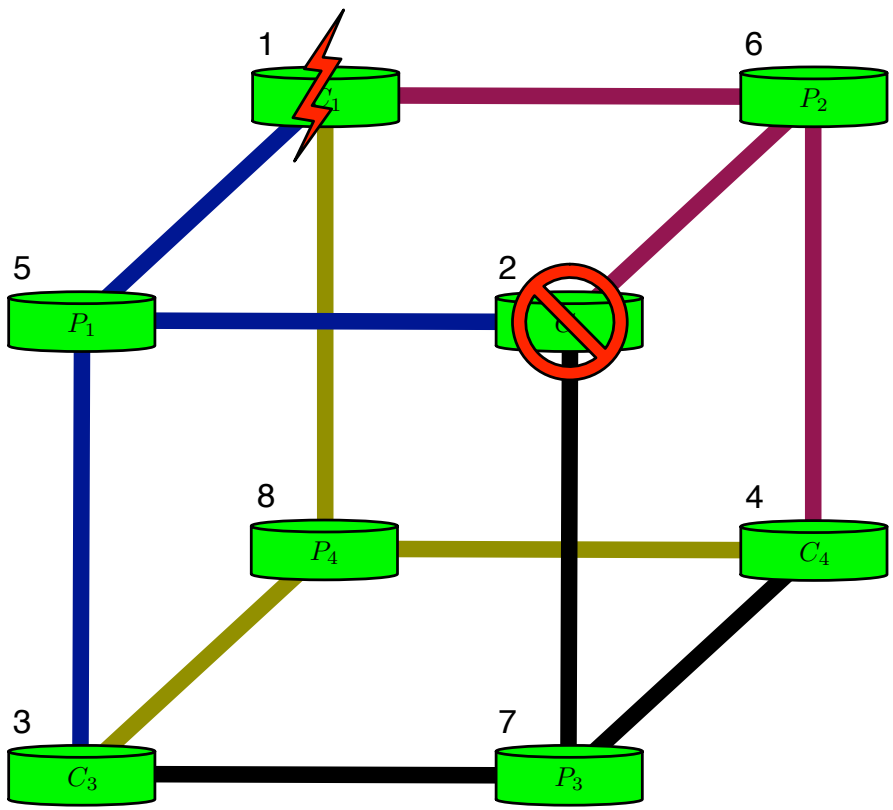
Examples

- repair groups of node 1:
  $\{3,4,8\}, \{2,7,8\}, \{2,4,6\}, \{3,6,7\}, \{2,3,5\}, \{4,5,7\}, \{5,6,8\}$



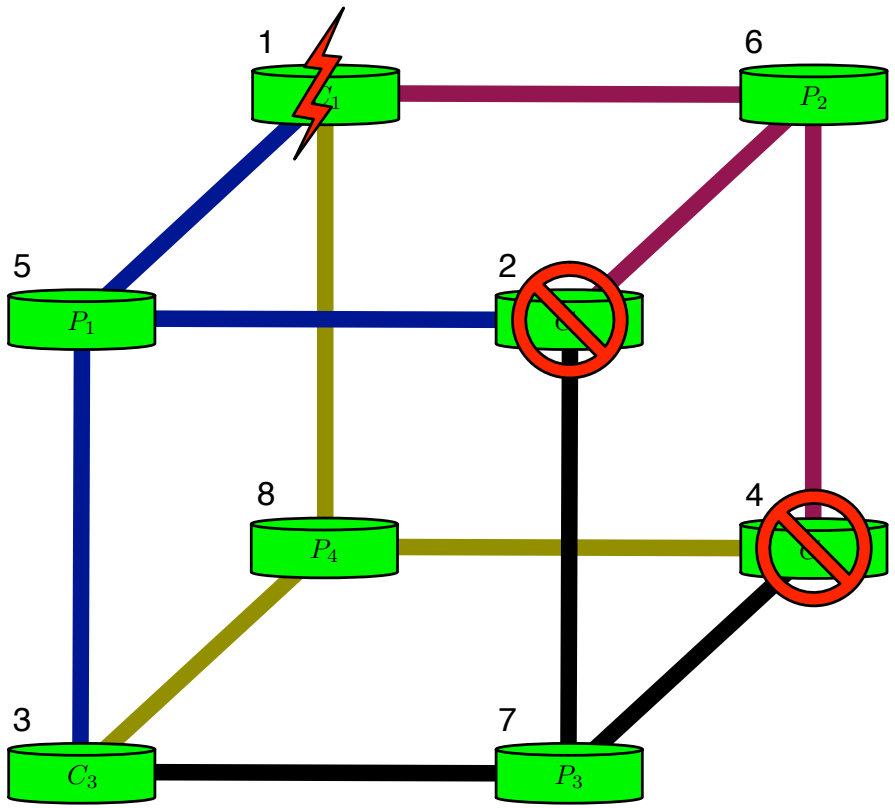
Examples

- repair groups of node 1 at the presence of one extra failure: $\{3, 4, 8\}, \{3, 6, 7\}, \{4, 5, 7\}, \{5, 6, 8\}$



Examples

- repair groups of node 1 at the presence of one extra failure: $\{3, 6, 7\}, \{5, 6, 8\}$



Examples

- The code achieves the bound at the point $(r = 3, k = 4)$.
- This code is an optimum $(3, 3, 0, 7)$, $(3, 3, 1, 4)$, and $(3, 3, 2, 2)$ robust locally repairable code.

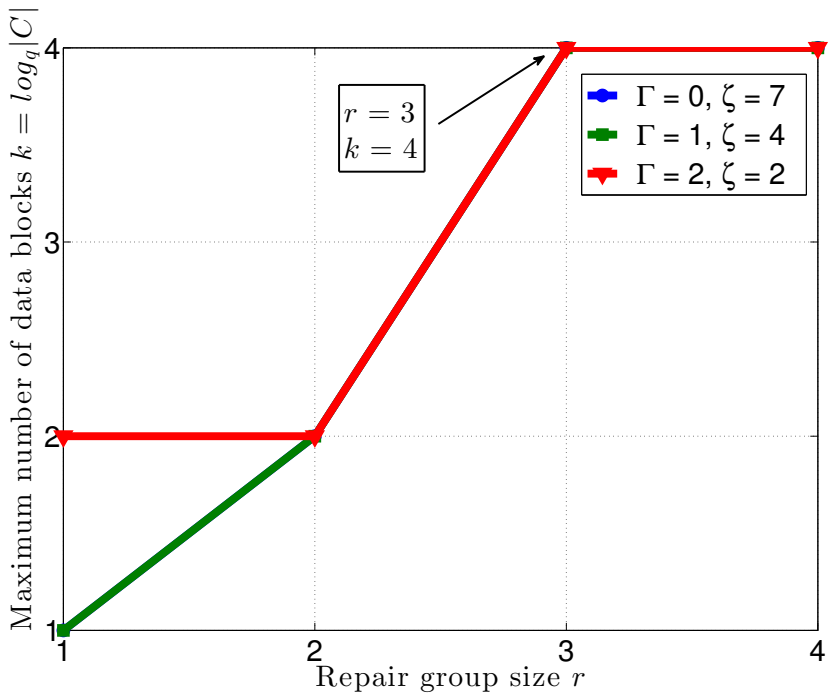
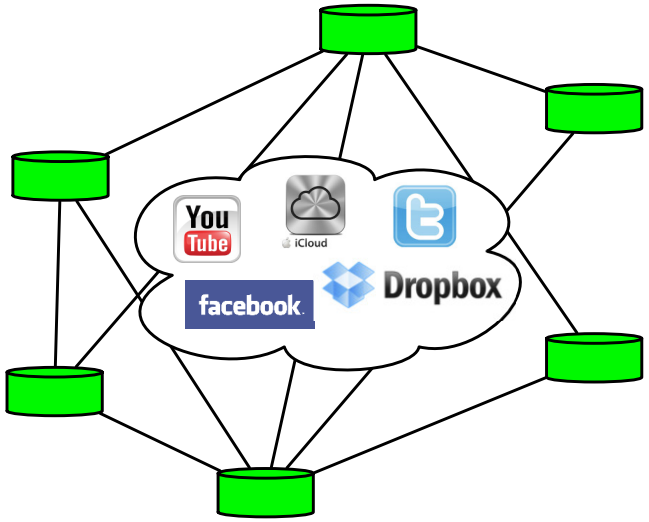


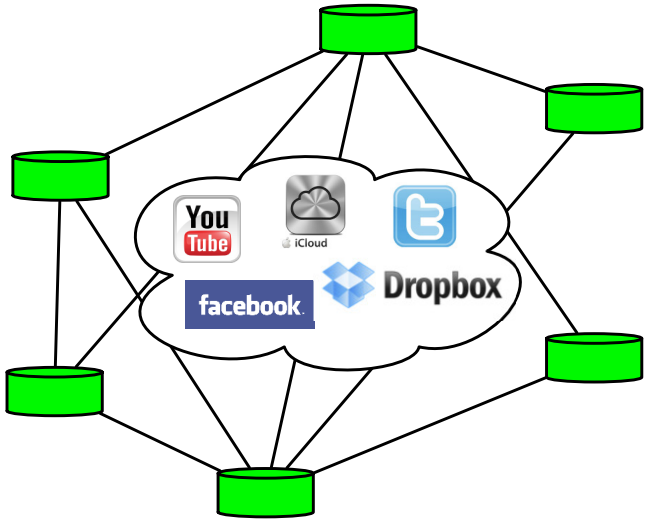
Figure: Upper bounds of $(r, 3, \Gamma, \zeta)$ linear storage code with $n = 8$.

Introduction



- In the majority of existing models the storage nodes and the transmission cost between nodes assumed to be ”identical”.

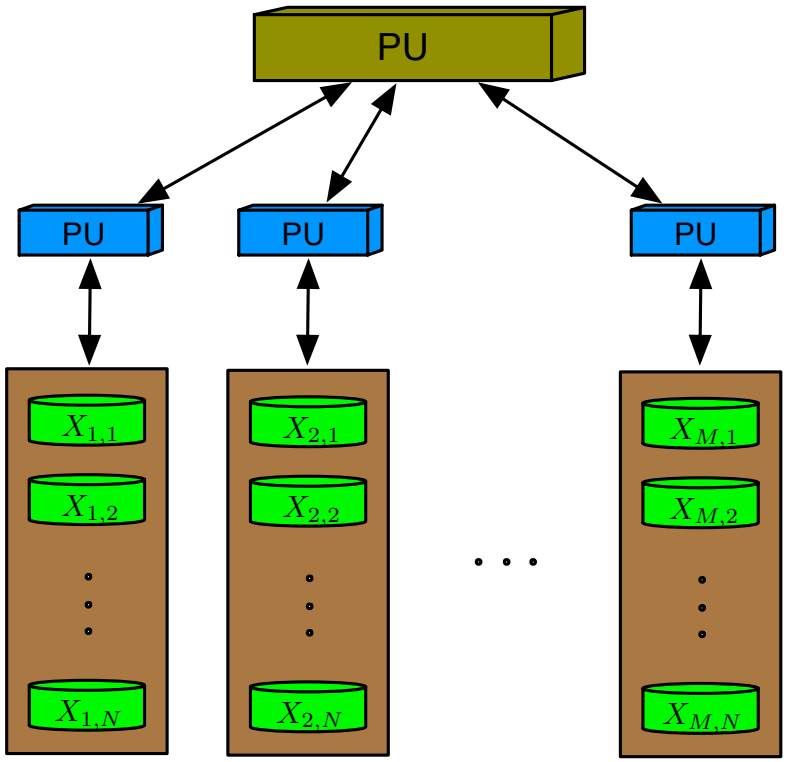
Introduction



- In the majority of existing models the storage nodes and the transmission cost between nodes assumed to be ”identical”.

In practice, this model is rarely close to the truth.

Rack model for storage networks



- A storage network with *M* racks each contains *N* storage nodes.
- The communication cost inside racks is much less than communication cost between racks.

Multi-rack storage codes

- Matrix $\mathbf{X}$ presents the stored data in the network.

$$\mathbf{X} = \begin{bmatrix} X_{1,1} & \cdots & X_{1,N} \\ \vdots & \ddots & \vdots \\ X_{M,1} & \cdots & X_{M,N} \end{bmatrix}$$

Definition

Consider three parity check matrices $\mathbf{H}$, $\mathbf{K}$ and $\mathbf{G}$ over $GF(q)$ of respectively sizes $S_1 \times N$, $S_2 \times N$ and $L \times M$. The three matrices induce a storage code such that $\mathbf{X}$ must satisfy the following parity check equations

$$\mathbf{H}\mathbf{X}^\top = \mathbf{0}$$
$$\mathbf{K}\mathbf{X}^\top\mathbf{G}^\top = \mathbf{0}$$

Multi-rack storage codes

- Node $X_{1,1}$ is failed:

$$\begin{bmatrix} 1 & 1 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} X_{1,1} \\ \vdots \\ X_{1,5} \end{bmatrix} = 0$$

$$X_{1,1} + X_{1,2} + X_{1,4} + X_{1,5} = 0$$

Multi-rack storage codes

- Node $X_{1,1}$ is failed:

$$\begin{bmatrix} 1 & 1 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} X_{1,1} \\ \vdots \\ X_{1,5} \end{bmatrix} = 0$$

$$X_{1,1} + X_{1,2} + X_{1,4} + X_{1,5} = 0$$

- Node $X_{1,2}$ is also failed or unavailable during repairing node $X_{1,1}$:

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} X_{1,1} & X_{2,1} & X_{3,1} \\ \vdots & \vdots & \vdots \\ X_{1,5} & X_{2,5} & X_{3,5} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = 0$$

$$(X_{1,2} + X_{1,3}) + (X_{3,2} + X_{3,3}) = 0$$

Multi-rack storage codes

- Node $X_{1,1}$ is failed:

$$\begin{bmatrix} 1 & 1 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} X_{1,1} \\ \vdots \\ X_{1,5} \end{bmatrix} = 0$$

$$X_{1,1} + X_{1,2} + X_{1,4} + X_{1,5} = 0$$

- Node $X_{1,2}$ is also failed or unavailable during repairing node $X_{1,1}$:

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} X_{1,1} & X_{2,1} & X_{3,1} \\ \vdots & \vdots & \vdots \\ X_{1,5} & X_{2,5} & X_{3,5} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = 0$$

$$(X_{1,2} + X_{1,3}) + (X_{3,2} + X_{3,3}) = 0 \quad \Rightarrow X_{1,2} = X_{1,3} + b$$

Multi-rack storage codes

- Node $X_{1,1}$ is failed:

$$\begin{bmatrix} 1 & 1 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} X_{1,1} \\ \vdots \\ X_{1,5} \end{bmatrix} = 0$$

$$X_{1,1} + X_{1,2} + X_{1,4} + X_{1,5} = 0$$

- Repairing $X_{1,1}$ by the survived nodes inside rack 1 and the nodes in other racks:

$$X_{1,1} = X_{1,3} + X_{1,4} + X_{1,5} + b$$

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \vdots & \vdots & \vdots \\ X_{1,5} & X_{2,5} & X_{3,5} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = 0$$

$$(X_{1,2} + X_{1,3}) + (X_{3,2} + X_{3,3}) = 0 \quad \Rightarrow X_{1,2} = X_{1,3} + b$$

Inside rack repair process

Let γ be the index set for all failed nodes in rack 1 and $j \in \gamma$. If $\beta_j \subseteq \{1, \ldots, N\}$ satisfies the following two criteria,

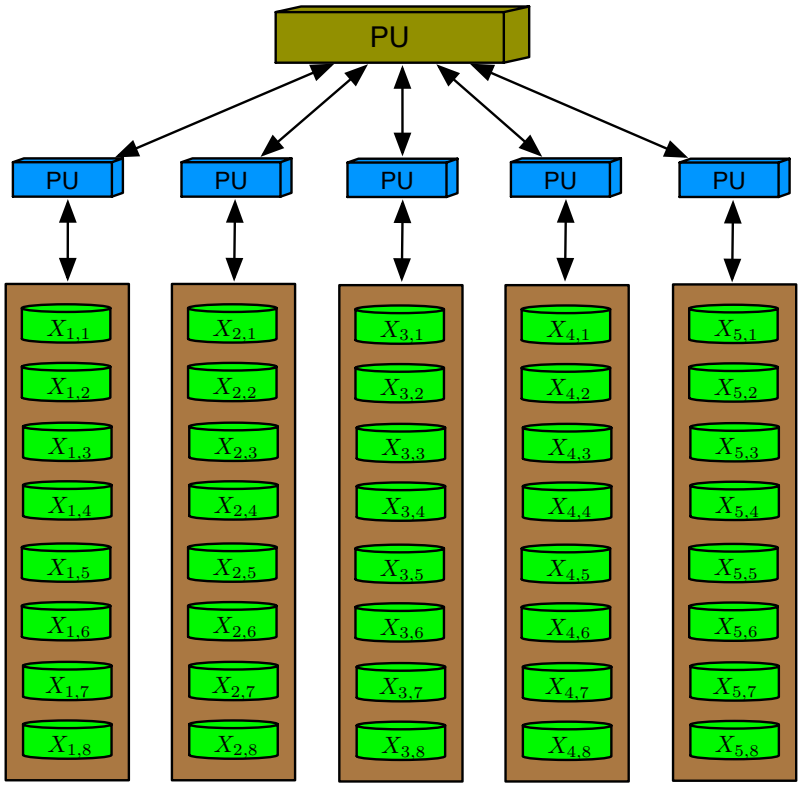
1. $\beta_j \in \Omega(\mathbf{H}, j)$, and
2. $\beta_j \cap \gamma = \emptyset$,

then there exists $c_{j,n}$ for $n \in \beta_j$ such that

$$X_{1,j} = \sum_{n \in \beta_j} c_{j,n} X_{1,n}.$$

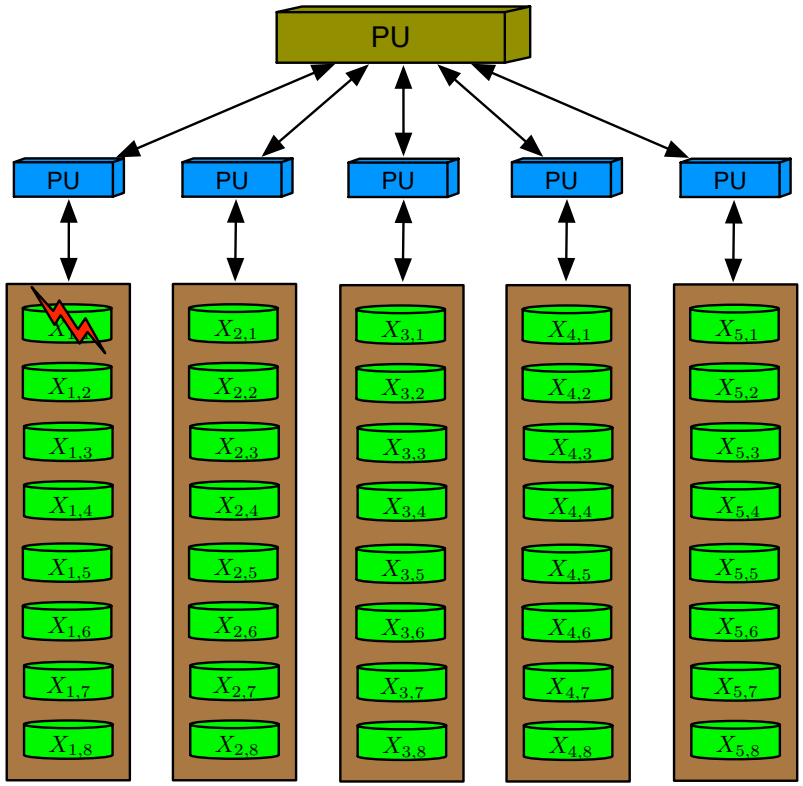
$\Omega(\mathbf{H}, j)$ is the collection of the supports of row spans of $\mathbf{H}$ (i.e., repair groups)

Multi-rack storage codes



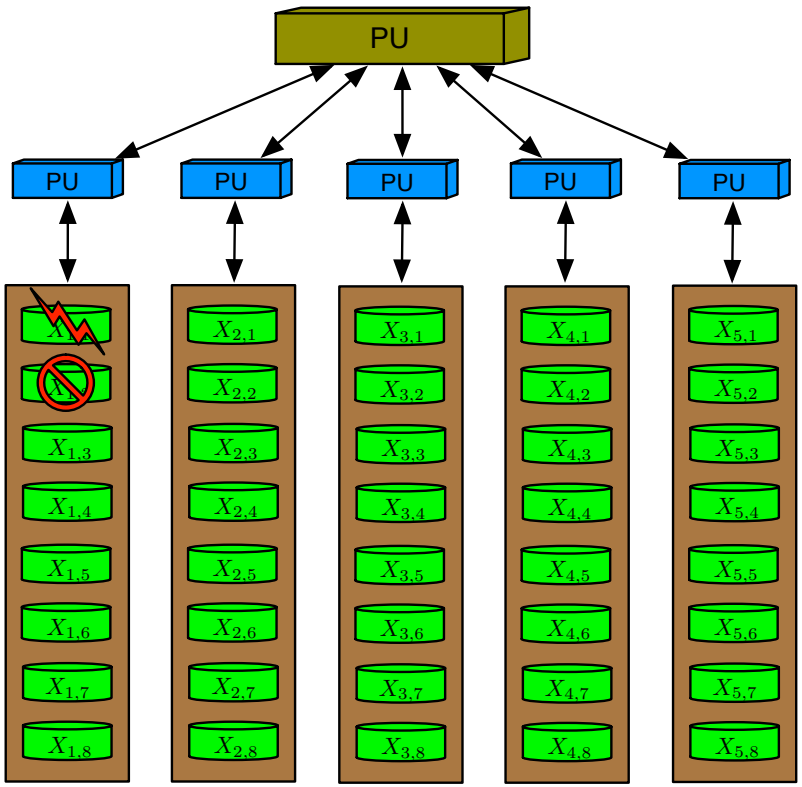
$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix},$$

Multi-rack storage codes



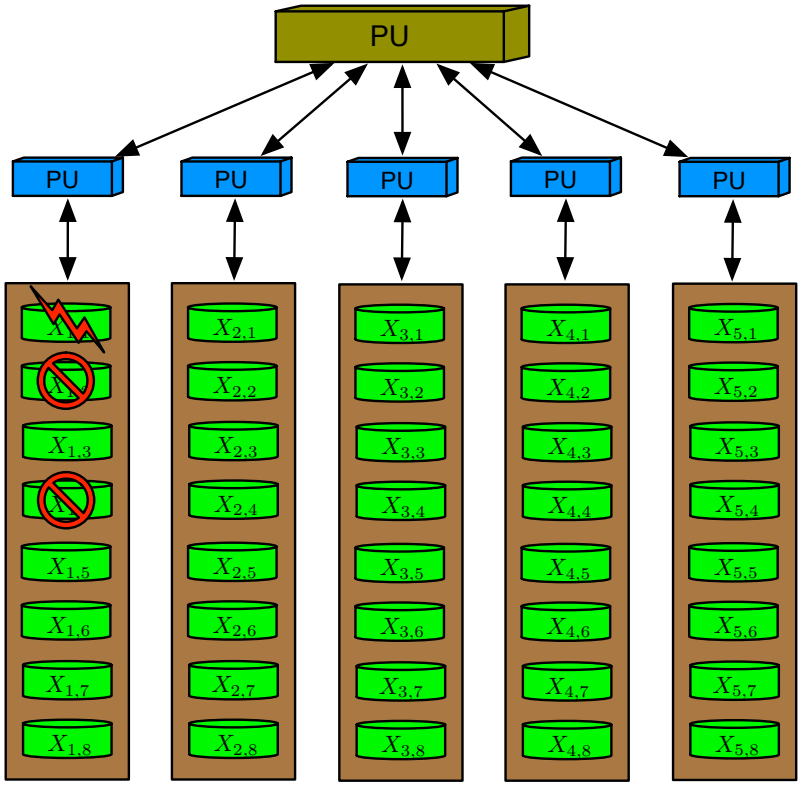
$$\Omega(\mathbf{H}, 1) = \Big\{\{3,4,8\}, \{2,7,8\}, \{2,4,6\}, \{3,6,7\}, \{2,3,5\}, \{4,5,7\}, \{5,6,8\}, \{2,3,4,5,6,7,8\}\Big\}.$$

Multi-rack storage codes



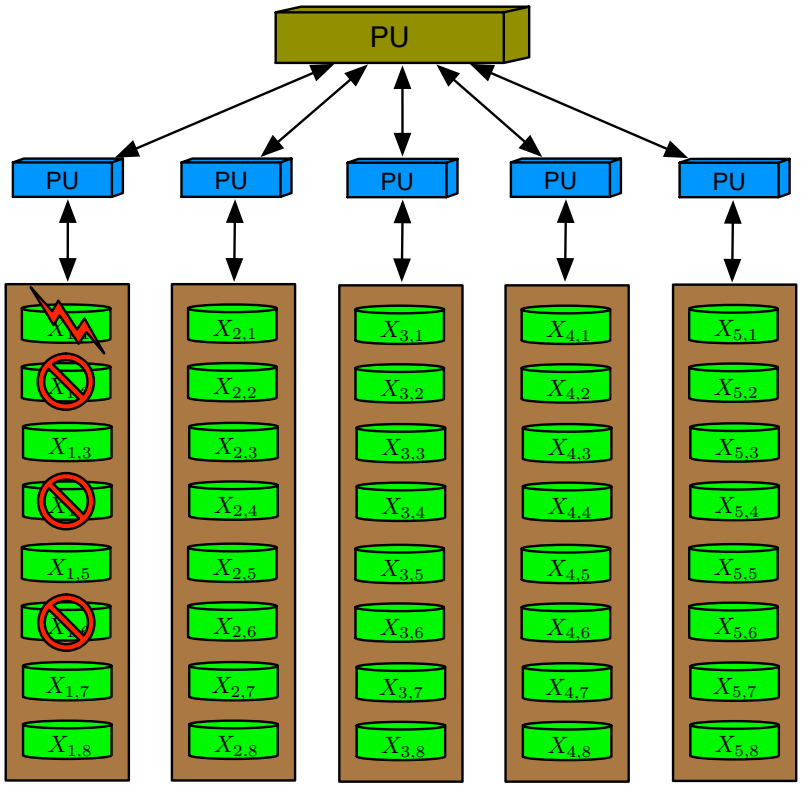
$$\beta_1 \in \Big\{\{3,4,8\},\{3,6,7\},\{4,5,7\},\{5,6,8\}\Big\}.$$

Multi-rack storage codes



$$\beta_1 \in \Big\{\{3,6,7\},\{5,6,8\}\Big\}.$$

Multi-rack storage codes



$$\beta_1 = \emptyset$$

Across rack repair process

Suppose that the node $X_{1,j}$ (i.e., the jth node in rack 1) fails. If $(\beta_j, \mu_j, \mathbf{r}_j, \tau)$ satisfies the following criteria,

1. $\mathbf{r}_j \in \langle \mathbf{K} \rangle$
2. $\mu_j = \{n \in \{1, \ldots, N\} : r_{j,n} \neq 0\}$
3. $\beta_j \in \Omega(\mathbf{H}, \mathbf{r}_j, j)$, and
4. $\beta_j \cap \gamma = \emptyset$,
5. $\tau \subseteq \{1, \ldots, M\} \in \Omega(\mathbf{G}, 1)$

then there exists $c_{j,n}$ for $n \in \beta_j$ such that

$$X_{1,j} = \sum_{m \in \tau} \left(\sum_{s \in \mu_j} d_{j,m,s} X_{m,s} \right) + \sum_{n \in \beta_j} c_{j,n} X_{1,n}.$$

Multi-rack storage codes

- Storage network with $M = 5$ racks each contains $N = 8$ storage nodes storing binary encoded data.

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

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

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

Multi-rack storage codes

- A set $\gamma = \{1, 2, 4, 6\}$ is failed in rack 1. The aim is to repair node 1 in rack 1. Rack 5 is not available during the repair process.

$$\mathbf{r}_1 = \begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

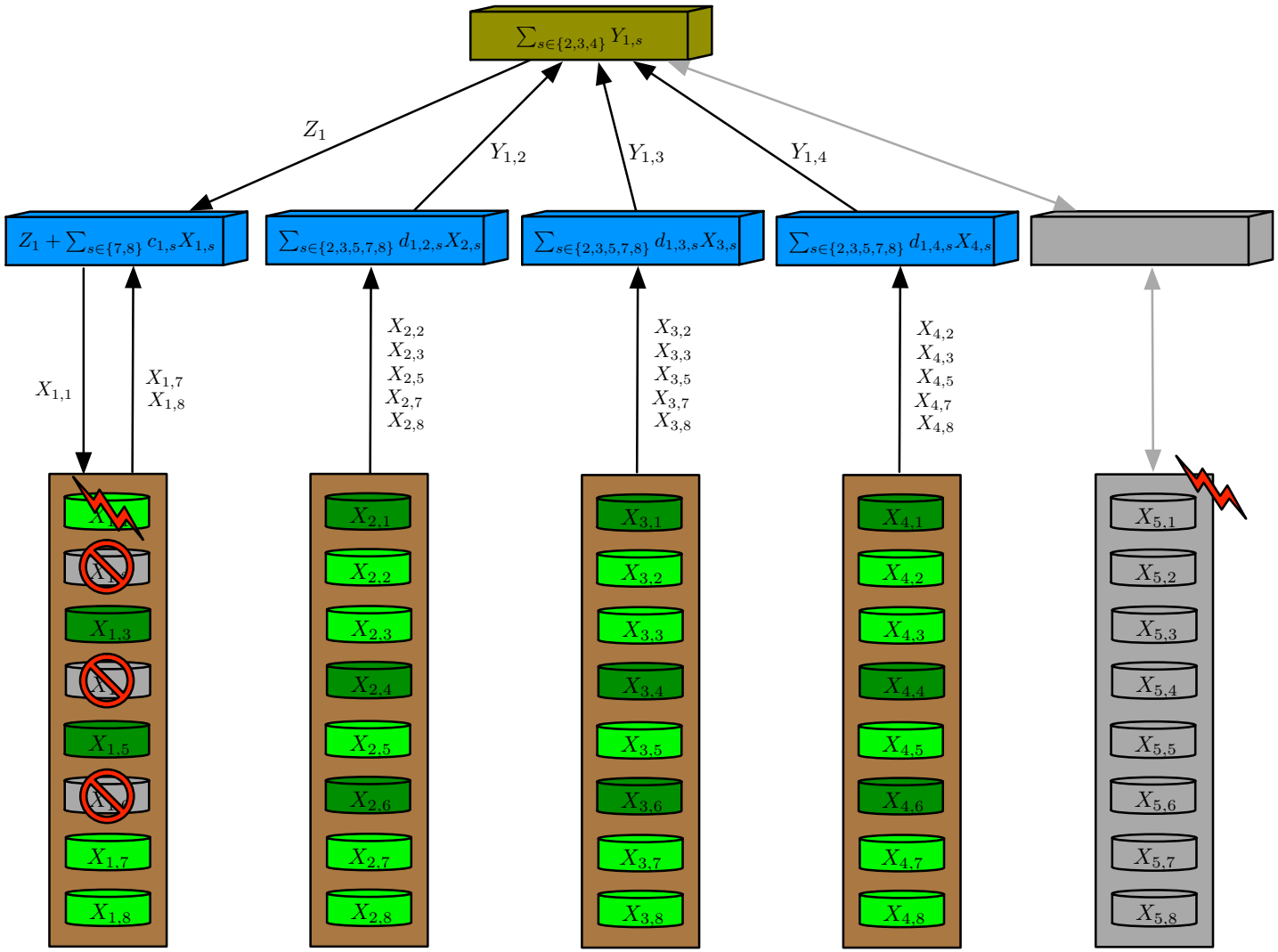
$$\mu_1 = \{2, 3, 5, 7, 8\}$$

$$\beta_1 = \{7, 8\}$$

$$\tau = \{2, 3, 4\}$$

$$X_{1,1} = \sum_{m \in \{2,3,4\}} \left(\sum_{s \in \{2,3,5,7,8\}} d_{1,m,s} X_{m,s} \right) + \sum_{n \in \{7,8\}} c_{1,n} X_{1,n}$$

Multi-rack storage codes



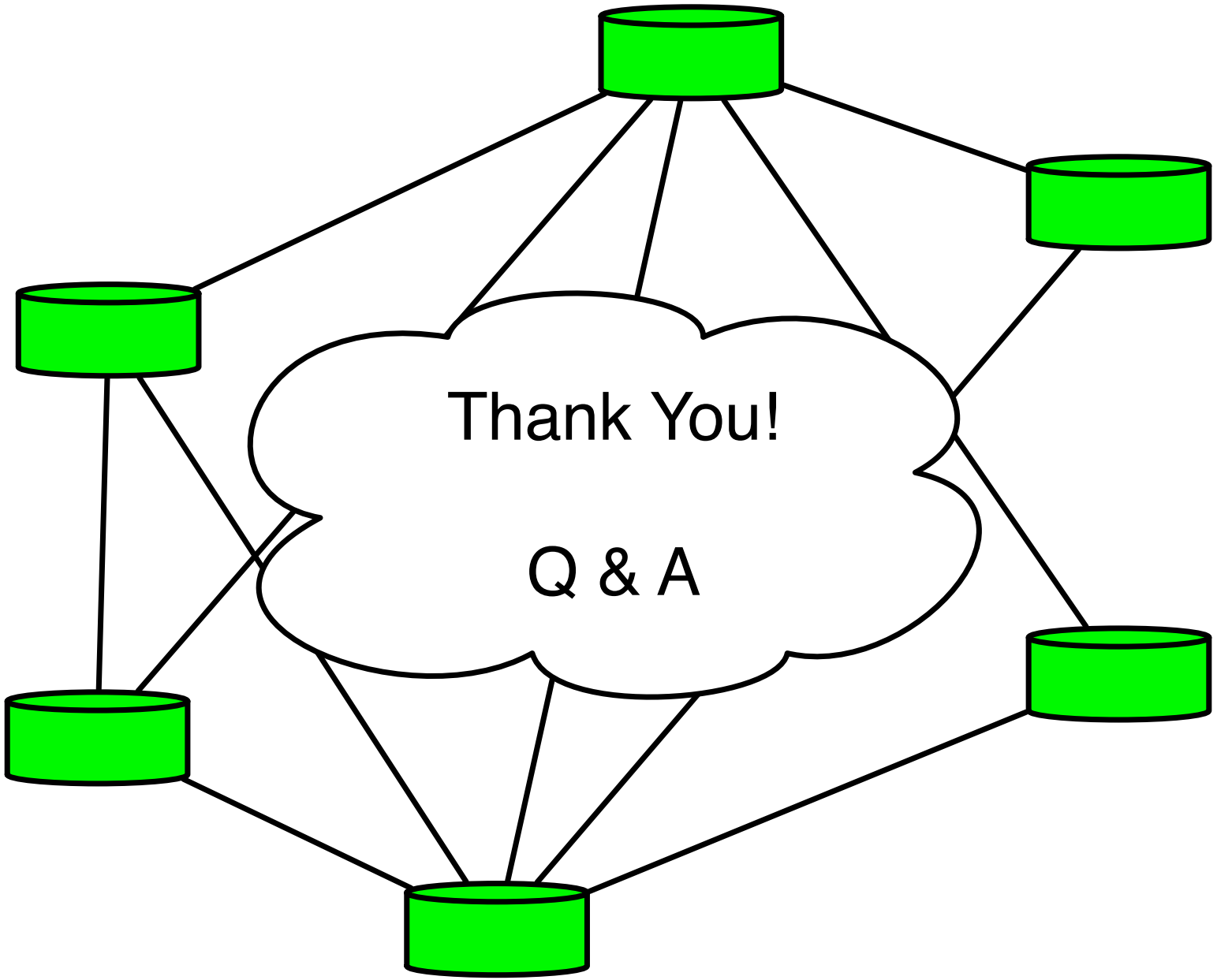
Multi-rack storage codes

Code rate

Let **H**, **K**, and **G** be respectively $S_1 \times N$, $S_2 \times N$, and $L \times M$ matrices. Then the rate of the generalised rack model code is

$$R \geq \frac{MN - MS_1 - LS_2}{MN}.$$

Equality holds if rows in **H** and **K** are linearly independent, and **G** is a full rank matrix.


Thank You!
Q & A