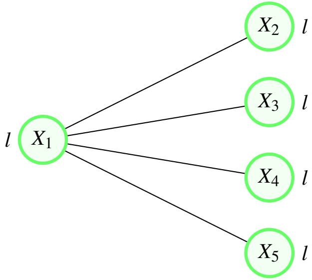
Node repair for Adversarial Graphical Networks

Adway Patra & Alexander Barg

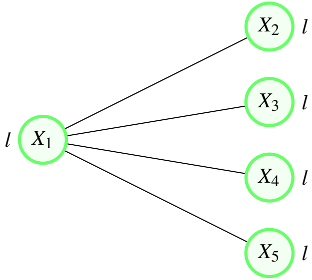
(University of Maryland, College Park)

International Symposium on Information Theory (ISIT), June 2023
Taipei City, Taiwan

Node repair in distributed storage

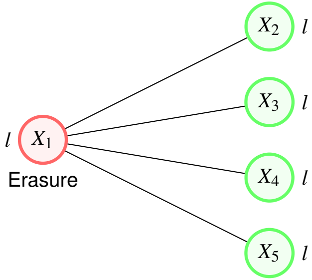


Node repair in distributed storage



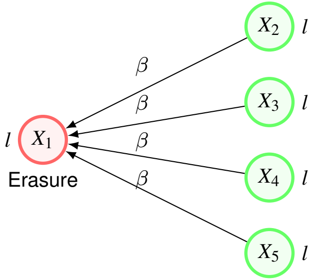
- An $[n, k, d, l, \beta, M]$ Regenerating Code $\mathcal{C} \subset \mathbb{F}^{nl}$, codewords viewed as $l \times n$ matrices over some finite field $\mathbb{F}$. Each codeword symbol stored in a node.

Node repair in distributed storage



- An $[n, k, d, l, \beta, M]$ Regenerating Code $\mathcal{C} \subset \mathbb{F}^{nl}$, codewords viewed as $l \times n$ matrices over some finite field $\mathbb{F}$. Each codeword symbol stored in a node.
- Correct erasures while trying to minimize total data "moved".

Node repair in distributed storage

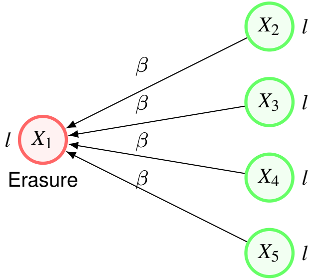


- An $[n, k, d, l, \beta, M]$ Regenerating Code $\mathcal{C} \subset \mathbb{F}^{nl}$, codewords viewed as $l \times n$ matrices over some finite field $\mathbb{F}$. Each codeword symbol stored in a node.
- Correct erasures while trying to minimize total data "moved".
- Total required transmission bounded by the Cut-set bound[1]

$$M \leqslant \sum_{i=0}^{k-1} \min\{l, (d-i)\beta\}$$

[1] Dimakis, Godfrey, Wu, Wainwright, Ramchandran, 2010

Node repair in distributed storage



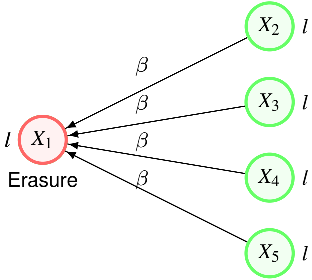
- An $[n, k, d, l, \beta, M]$ Regenerating Code $\mathcal{C} \subset \mathbb{F}^{nl}$, codewords viewed as $l \times n$ matrices over some finite field $\mathbb{F}$. Each codeword symbol stored in a node.
- Correct erasures while trying to minimize total data "moved".
- Total required transmission bounded by the Cut-set bound[1]

$$M \leqslant \sum_{i=0}^{k-1} \min\{l, (d-i)\beta\}$$

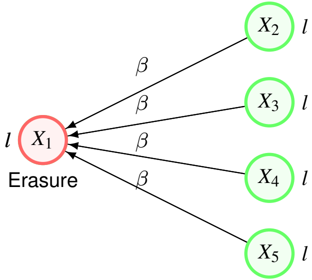
- Different pairs of (l, β) satisfying the above with equality give rise to different points on the storage-bandwidth trade-off.

[1] Dimakis, Godfrey, Wu, Wainwright, Ramchandran, 2010

Node repair in distributed storage

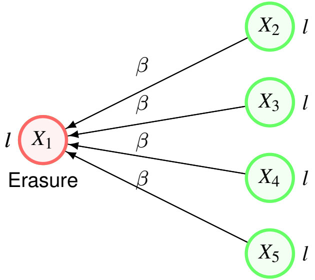


Node repair in distributed storage



- Two extreme points of the trade-off curve are

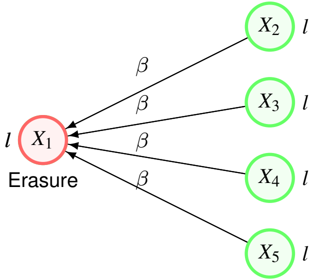
Node repair in distributed storage



- Two extreme points of the trade-off curve are
  - The Minimum Storage Regenerating (MSR) point characterized by

$$l = (d - k + 1)\beta.$$

Node repair in distributed storage



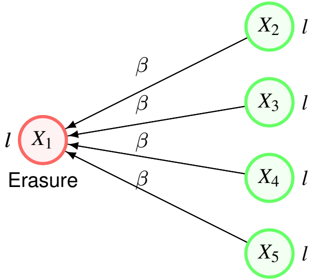
- Two extreme points of the trade-off curve are
  - The Minimum Storage Regenerating (MSR) point characterized by

$$l = (d - k + 1)\beta.$$

  - The Minimum Bandwidth Regenerating (MBR) point characterized by

$$l = d\beta.$$

Node repair in distributed storage



- Two extreme points of the trade-off curve are
  - The Minimum Storage Regenerating (MSR) point characterized by

$$l = (d - k + 1)\beta.$$

  - The Minimum Bandwidth Regenerating (MBR) point characterized by

$$l = d\beta.$$

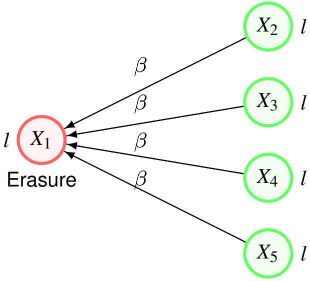
- For interior points

$$d\beta > l > (d - k + 1)\beta.$$

Moving away from traditional setting

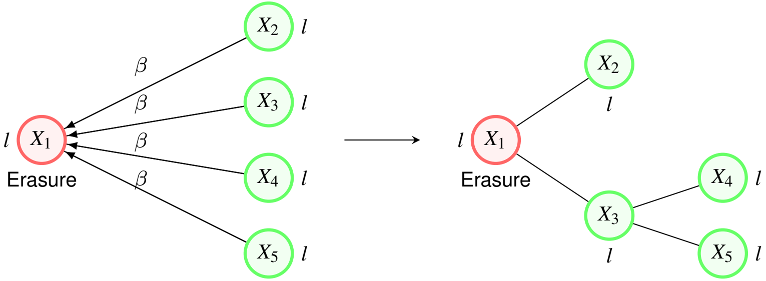
Moving away from traditional setting

- General problem assumes d helper nodes are chosen from the direct neighbors of the failed node, i.e., high connectivity.



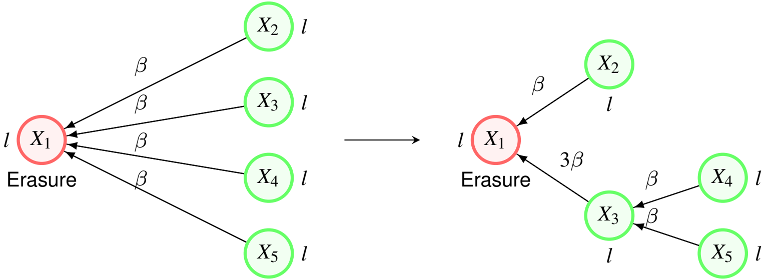
Moving away from traditional setting

- General problem assumes d helper nodes are chosen from the direct neighbors of the failed node, i.e., high connectivity.



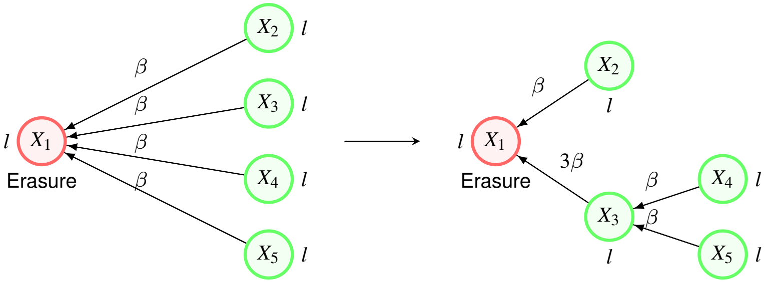
Moving away from traditional setting

- General problem assumes d helper nodes are chosen from the direct neighbors of the failed node, i.e., high connectivity.



Moving away from traditional setting

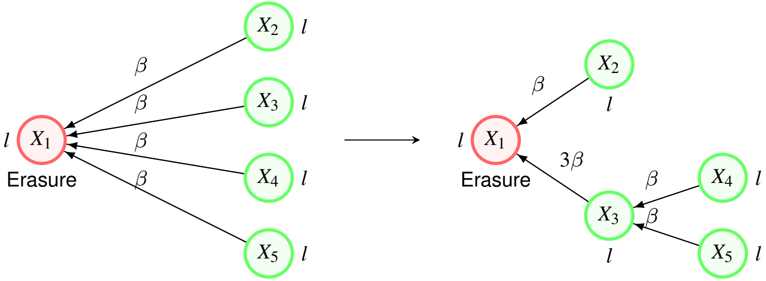
- General problem assumes d helper nodes are chosen from the direct neighbors of the failed node, i.e., high connectivity.



- Same data gets transmitted multiple times.

Moving away from traditional setting

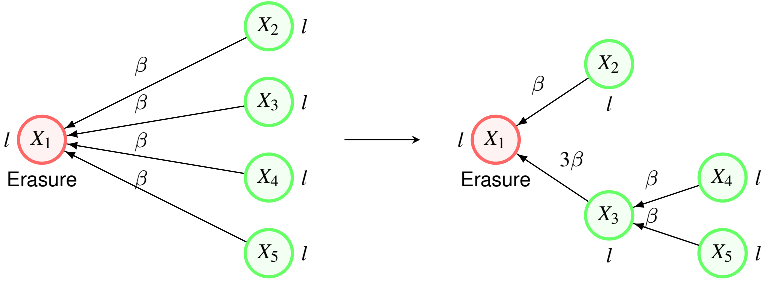
- General problem assumes d helper nodes are chosen from the direct neighbors of the failed node, i.e., high connectivity.



- Same data gets transmitted multiple times.
- Total required communication depends on the structure of the tree.

Moving away from traditional setting

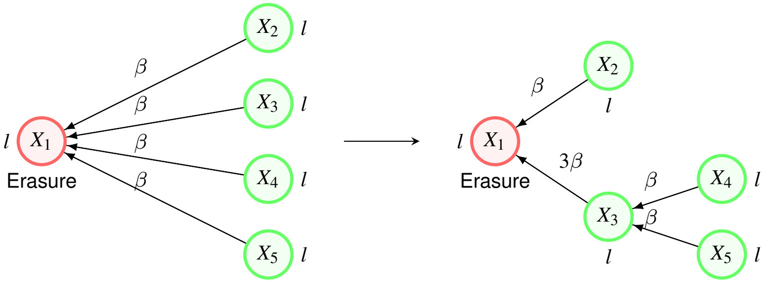
- General problem assumes d helper nodes are chosen from the direct neighbors of the failed node, i.e., high connectivity.



- Same data gets transmitted multiple times.
- Total required communication depends on the structure of the tree.
  - For example, if the helpers are on a line, the failed node being at the end, then $\frac{d(d+1)\beta}{2} = \Theta(d^2)$ transmission required.

Moving away from traditional setting

- General problem assumes d helper nodes are chosen from the direct neighbors of the failed node, i.e., high connectivity.



- Same data gets transmitted multiple times.
- Total required communication depends on the structure of the tree.
  - For example, if the helpers are on a line, the failed node being at the end, then $\frac{d(d+1)\beta}{2} = \Theta(d^2)$ transmission required.

Question : Is it possible to process the data to reduce communication?

The bounds: How much can we process?

The bounds: How much can we process?

- Node v_i holds random variable W_i.

The bounds: How much can we process?

- Node v_i holds random variable W_i.
- For failed node v_f, d helper nodes $v_1, \cdots, v_d$. Helper node v_i would have sent S_i^f to v_f in case of direct connectivity.

The bounds: How much can we process?

- Node v_i holds random variable W_i.
- For failed node v_f, d helper nodes $v_1, \cdots, v_d$. Helper node v_i would have sent S_i^f to v_f in case of direct connectivity.
- Operating at an arbitrary point on the trade-off curve: $H(W_f) = l$, $H(S_i^f) = \beta$.

The bounds: How much can we process?

- Node v_i holds random variable W_i.
- For failed node v_f, d helper nodes $v_1, \cdots, v_d$. Helper node v_i would have sent S_i^f to v_f in case of direct connectivity.
- Operating at an arbitrary point on the trade-off curve: $H(W_f) = l$, $H(S_i^f) = \beta$.
- $H(S_i^f | W_i) = 0, H(W_f | S_1^f, \cdots, S_d^f) = 0$.

The bounds: How much can we process?

- Node v_i holds random variable W_i.
- For failed node v_f, d helper nodes $v_1, \cdots, v_d$. Helper node v_i would have sent S_i^f to v_f in case of direct connectivity.
- Operating at an arbitrary point on the trade-off curve: $H(W_f) = l$, $H(S_i^f) = \beta$.
- $H(S_i^f | W_i) = 0, H(W_f | S_1^f, \cdots, S_d^f) = 0$.
- Let $v_f, f \in [n]$ be the failed node. For a subset of the helper nodes $E \subset D$ let R_E^f be a function of S_E^f such that

$$H(W_f | R_E^f, S_{D\setminus E}^f) = 0.$$

The bounds: How much can we process?

- Node v_i holds random variable W_i.
- For failed node v_f, d helper nodes $v_1, \cdots, v_d$. Helper node v_i would have sent S_i^f to v_f in case of direct connectivity.
- Operating at an arbitrary point on the trade-off curve: $H(W_f) = l$, $H(S_i^f) = \beta$.
- $H(S_i^f|W_i) = 0, H(W_f|S_1^f, \cdots, S_d^f) = 0$.
- Let $v_f, f \in [n]$ be the failed node. For a subset of the helper nodes $E \subset D$ let R_E^f be a function of S_E^f such that

$$H(W_f|R_E^f, S_{D\setminus E}^f) = 0.$$

In our prior work [Patra and Barg, 2022]

For MSR codes, if $|E| \geqslant d-k+1$,

$$H(R_E^f) \geqslant l$$

The bounds: How much can we process?

- Node v_i holds random variable W_i.
- For failed node v_f, d helper nodes $v_1, \cdots, v_d$. Helper node v_i would have sent S_i^f to v_f in case of direct connectivity.
- Operating at an arbitrary point on the trade-off curve: $H(W_f) = l$, $H(S_i^f) = \beta$.
- $H(S_i^f|W_i) = 0, H(W_f|S_1^f, \cdots, S_d^f) = 0.$
- Let $v_f, f \in [n]$ be the failed node. For a subset of the helper nodes $E \subset D$ let R_E^f be a function of S_E^f such that

$$H(W_f|R_E^f, S_{D\setminus E}^f) = 0.$$

In our prior work [Patra and Barg, 2022]

For MSR codes, if $|E| \geqslant d - k + 1$,

$$H(R_E^f) \geqslant l$$

Generalised Version

For any Regenerating Code, if $|E| \geqslant d - k + 1$, then

$$H(R_E^f) \geqslant M - \sum_{i=1}^{k-1} \min\{l, (d-i+1)\beta\}.$$

Proof of Lemma

- Given $X_{D\setminus E}$ the information contained in R_E^f is sufficient to repair v_f, i.e.,

$$H(W_f|R_E^f, W_{D\setminus E}) = 0.$$

Proof of Lemma

- Given $X_{D\setminus E}$ the information contained in R_E^f is sufficient to repair v_f, i.e.,

$$H(W_f|R_E^f, W_{D\setminus E}) = 0.$$

- Take a set $A \subset E$ with $|A| = k - 1 - |D\setminus E|$. Now,

$$H(R_E^f, W_{D\setminus E}, W_A) = H(R_E^f, W_{D\setminus E}, W_f, W_A) \geqslant M$$

by the recoverability property.

Proof of Lemma

- Given $X_{D\setminus E}$ the information contained in R_E^f is sufficient to repair v_f, i.e.,

$$H(W_f|R_E^f, W_{D\setminus E}) = 0.$$

- Take a set $A \subset E$ with $|A| = k - 1 - |D\setminus E|$. Now,

$$H(R_E^f, W_{D\setminus E}, W_A) = H(R_E^f, W_{D\setminus E}, W_f, W_A) \geqslant M$$

  by the recoverability property.

-

$$\begin{aligned} H(R_E^f) + H(W_{D\setminus E}, W_A) &\geqslant H(R_E^f, W_{D\setminus E}, W_A) \\ H(R_E^f) &\geqslant M - H(W_{D\setminus E}, W_A) \\ &\geqslant M - \sum_{i=1}^{k-1} \min\{l, (d-i+1)\beta\} \end{aligned}$$

  using $H(W_i|W_X) \leqslant \min\{l, (d-|X|)\beta\}$ for any $i \notin X$.

Proof of Lemma

- Given $X_{D\backslash E}$ the information contained in R_E^f is sufficient to repair v_f, i.e.,

$$H(W_f | R_E^f, W_{D\backslash E}) = 0.$$

- Take a set $A \subset E$ with $|A| = k - 1 - |D\backslash E|$. Now,

$$H(R_E^f, W_{D\backslash E}, W_A) = H(R_E^f, W_{D\backslash E}, W_f, W_A) \geqslant M$$

  by the recoverability property.

- $$\begin{aligned} H(R_E^f) + H(W_{D\backslash E}, W_A) &\geqslant H(R_E^f, W_{D\backslash E}, W_A) \\ H(R_E^f) &\geqslant M - H(W_{D\backslash E}, W_A) \\ &\geqslant M - \sum_{i=1}^{k-1} \min\{l, (d-i+1)\beta\} \end{aligned}$$

  using $H(W_i | W_X) \leqslant \min\{l, (d - |X|)\beta\}$ for any $i \notin X$.

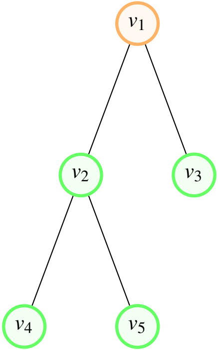
- For MSR case, $l = (d - i + 1)\beta$ and $M = kl$, hence

$$H(R_E^f) \geqslant l$$

Achieving the bounds: Using MSR Product Matrix codes[2]

Example

- Take an $[n = 5, k = 3, d = 4, l = 2, \beta = 1, M = 6]$ MSR Product Matrix code.

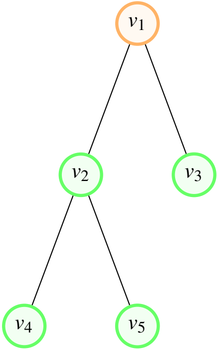


[2]Rashmi, Shah, Vijay Kumar, 2011

Achieving the bounds: Using MSR Product Matrix codes[2]

Example

- Take an $[n=5, k=3, d=4, l=2, \beta=1, M=6]$ MSR Product Matrix code.
- $M = [S_1, S_2]^T = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \\ m_{21} & m_{22} \\ m_{22} & m_{23} \end{bmatrix}, \Psi_{5\times 4} = \begin{bmatrix} \Phi & \Lambda\Phi \end{bmatrix}, C = \Psi M.$

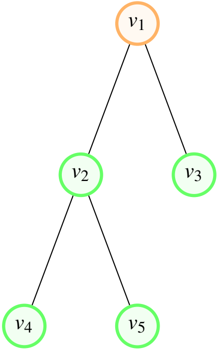


[2]Rashmi, Shah, Vijay Kumar, 2011

Achieving the bounds: Using MSR Product Matrix codes[2]

Example

- Take an $[n = 5, k = 3, d = 4, l = 2, \beta = 1, M = 6]$ MSR Product Matrix code.
- $M = [S_1, S_2]^T = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \\ m_{21} & m_{22} \\ m_{22} & m_{23} \end{bmatrix}, \Psi_{5\times 4} = \begin{bmatrix} \Phi & \Lambda\Phi \end{bmatrix}, C = \Psi M.$
- Node i sends $y_i = \Psi_i M \Phi_1^T = (\Phi_i S_1 + \lambda_i \Phi_i S_2)\Phi_1^T$.

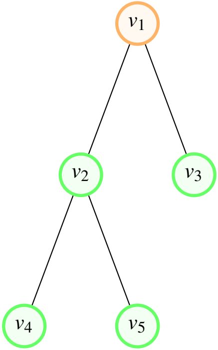


[2]Rashmi, Shah, Vijay Kumar, 2011

Achieving the bounds: Using MSR Product Matrix codes[2]

Example

- Take an $[n = 5, k = 3, d = 4, l = 2, \beta = 1, M = 6]$ MSR Product Matrix code.
- $M = [S_1, S_2]^T = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \\ m_{21} & m_{22} \\ m_{22} & m_{23} \end{bmatrix}, \Psi_{5\times 4} = \begin{bmatrix} \Phi & \Lambda\Phi \end{bmatrix}, C = \Psi M.$
- Node i sends $y_i = \Psi_i M \Phi_1^T = (\Phi_i S_1 + \lambda_i \Phi_i S_2)\Phi_1^T$.
- Node 1 inverts matrix Ψ_D to get $M\Phi_i^t = \begin{bmatrix} S_1\Phi_1^T \\ S_2\Phi_1^T \end{bmatrix} = \Psi_D^{-1} y_D$ and calculates $\Phi_1 S_1 + \lambda_1 \Phi_1 S_2$.

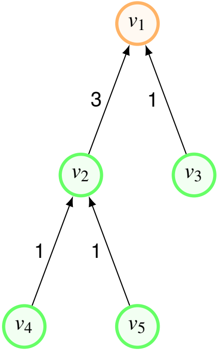


[2] Rashmi, Shah, Vijay Kumar, 2011

Achieving the bounds: Using MSR Product Matrix codes[2]

Example

- Take an $[n = 5, k = 3, d = 4, l = 2, \beta = 1, M = 6]$ MSR Product Matrix code.
- $M = [S_1, S_2]^T = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \\ m_{21} & m_{22} \\ m_{22} & m_{23} \end{bmatrix}, \Psi_{5\times 4} = \begin{bmatrix} \Phi & \Lambda\Phi \end{bmatrix}, C = \Psi M.$
- Node i sends $y_i = \Psi_i M \Phi_1^T = (\Phi_i S_1 + \lambda_i \Phi_i S_2)\Phi_1^T$.
- Node 1 inverts matrix Ψ_D to get $M\Phi_i^t = \begin{bmatrix} S_1\Phi_1^T \\ S_2\Phi_1^T \end{bmatrix} = \Psi_D^{-1} y_D$ and calculates $\Phi_1 S_1 + \lambda_1 \Phi_1 S_2$.

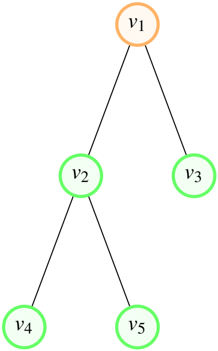


[2]Rashmi, Shah, Vijay Kumar, 2011

Achieving the bounds: Using MSR Product Matrix codes[2]

Example

- Take an $[n = 5, k = 3, d = 4, l = 2, \beta = 1, M = 6]$ MSR Product Matrix code.
- $M = [S_1, S_2]^T = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \\ m_{21} & m_{22} \\ m_{22} & m_{23} \end{bmatrix}$, $\Psi_{5\times 4} = \begin{bmatrix} \Phi & \Lambda\Phi \end{bmatrix}$, $C = \Psi M$.
- Node i sends $y_i = \Psi_i M \Phi_1^T = (\Phi_i S_1 + \lambda_i \Phi_i S_2)\Phi_1^T$.
- Node 1 inverts matrix Ψ_D to get $M\Phi_i^t = \begin{bmatrix} S_1\Phi_1^T \\ S_2\Phi_1^T \end{bmatrix} = \Psi_D^{-1} y_D$ and calculates $\Phi_1 S_1 + \lambda_1 \Phi_1 S_2$.
- Observe $C_1^T = \begin{bmatrix} I_2 & \lambda_1 I_2 \end{bmatrix} M\Phi_i^t = \mathcal{R}_1 \Psi_D^{-1} y_D = U^{1,D} y_D$

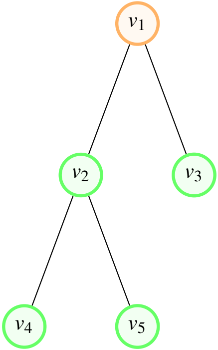


[2] Rashmi, Shah, Vijay Kumar, 2011

Achieving the bounds: Using MSR Product Matrix codes[2]

Example

- Take an $[n = 5, k = 3, d = 4, l = 2, \beta = 1, M = 6]$ MSR Product Matrix code.
- $M = [S_1, S_2]^T = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \\ m_{21} & m_{22} \\ m_{22} & m_{23} \end{bmatrix}, \Psi_{5\times 4} = \begin{bmatrix} \Phi & \Lambda\Phi \end{bmatrix}, C = \Psi M.$
- Node i sends $y_i = \Psi_i M \Phi_1^T = (\Phi_i S_1 + \lambda_i \Phi_i S_2)\Phi_1^T$.
- Node 1 inverts matrix Ψ_D to get $M\Phi_i^t = \begin{bmatrix} S_1\Phi_1^T \\ S_2\Phi_1^T \end{bmatrix} = \Psi_D^{-1} y_D$ and calculates $\Phi_1 S_1 + \lambda_1 \Phi_1 S_2$.
- Observe $C_1^T = \begin{bmatrix} I_2 & \lambda_1 I_2 \end{bmatrix} M\Phi_i^t = \mathcal{R}_1 \Psi_D^{-1} y_D = U^{1,D} y_D$
- So v_2 can transmit instead $U_4^{1,D} y_4 + U_5^{1,D} y_5 + U_2^{1,D} y_2$, $\rightarrow$ *Intermediate Processing (IP)*.

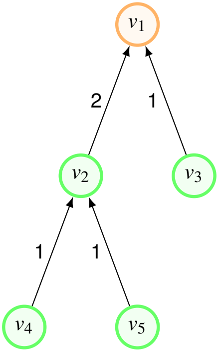


[2] Rashmi, Shah, Vijay Kumar, 2011

Achieving the bounds: Using MSR Product Matrix codes[2]

Example

- Take an $[n = 5, k = 3, d = 4, l = 2, \beta = 1, M = 6]$ MSR Product Matrix code.
- $M = [S_1, S_2]^T = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \\ m_{21} & m_{22} \\ m_{22} & m_{23} \end{bmatrix}, \Psi_{5\times 4} = \begin{bmatrix} \Phi & \Lambda\Phi \end{bmatrix}, C = \Psi M.$
- Node i sends $y_i = \Psi_i M \Phi_1^T = (\Phi_i S_1 + \lambda_i \Phi_i S_2)\Phi_1^T$.
- Node 1 inverts matrix Ψ_D to get $M\Phi_i^t = \begin{bmatrix} S_1\Phi_1^T \\ S_2\Phi_1^T \end{bmatrix} = \Psi_D^{-1} y_D$ and calculates $\Phi_1 S_1 + \lambda_1 \Phi_1 S_2$.
- Observe $C_1^T = \begin{bmatrix} I_2 & \lambda_1 I_2 \end{bmatrix} M\Phi_i^t = \mathcal{R}_1 \Psi_D^{-1} y_D = U^{1,D} y_D$
- So v_2 can transmit instead $U_4^{1,D} y_4 + U_5^{1,D} y_5 + U_2^{1,D} y_2$, $\rightarrow$ *Intermediate Processing (IP)*.



[2]Rashmi, Shah, Vijay Kumar, 2011

Achieving the bounds: Any $\mathbb{F}$-linear code

In general, for any $\mathbb{F}$-linear code:

Achieving the bounds: Any $\mathbb{F}$-linear code

In general, for any $\mathbb{F}$-linear code:

- Say helper node i needs to send $y_i \in \mathbb{F}^{\beta}$ to v_1 in the non-constrained setting.

Achieving the bounds: Any $\mathbb{F}$-linear code

In general, for any $\mathbb{F}$-linear code:

- Say helper node i needs to send $y_i \in \mathbb{F}^\beta$ to v_1 in the non-constrained setting.
- There exists $l \times d\beta$ matrix $\mathcal{U}$:

$$\begin{bmatrix} c_{1,1} \\ c_{1,2} \\ \vdots \\ c_{1,l} \end{bmatrix} = \begin{bmatrix} \mathcal{U}_1 & \mathcal{U}_2 & \cdots & \mathcal{U}_d \end{bmatrix} \begin{bmatrix} y_{i_1} \\ y_{i_2} \\ \vdots \\ y_{i_d} \end{bmatrix}$$

Achieving the bounds: Any $\mathbb{F}$-linear code

In general, for any $\mathbb{F}$-linear code:

- Say helper node i needs to send $y_i \in \mathbb{F}^\beta$ to v_1 in the non-constrained setting.
- There exists $l \times d\beta$ matrix $\mathcal{U}$:

$$\begin{bmatrix} c_{1,1} \\ c_{1,2} \\ \vdots \\ c_{1,l} \end{bmatrix} = \begin{bmatrix} \mathcal{U}_1 & \mathcal{U}_2 & \cdots & \mathcal{U}_d \end{bmatrix} \begin{bmatrix} y_{i_1} \\ y_{i_2} \\ \vdots \\ y_{i_d} \end{bmatrix}$$

- Node v_x that receives y_is from at least $d-k+1$ other nodes E can send

Achieving the bounds: Any $\mathbb{F}$-linear code

In general, for any $\mathbb{F}$-linear code:

- Say helper node i needs to send $y_i \in \mathbb{F}^\beta$ to v_1 in the non-constrained setting.
- There exists $l \times d\beta$ matrix $\mathcal{U}$:

$$\begin{bmatrix} c_{1,1} \\ c_{1,2} \\ \vdots \\ c_{1,l} \end{bmatrix} = \begin{bmatrix} \mathcal{U}_1 & \mathcal{U}_2 & \cdots & \mathcal{U}_d \end{bmatrix} \begin{bmatrix} y_{i_1} \\ y_{i_2} \\ \vdots \\ y_{i_d} \end{bmatrix}$$

- Node v_x that receives y_is from at least $d-k+1$ other nodes E can send
  - $\sum_{j \in E} \mathcal{U}_j y_{i_j} + \mathcal{U}_x y_{i_x} \longrightarrow l$ symbols instead of

Achieving the bounds: Any $\mathbb{F}$-linear code

In general, for any $\mathbb{F}$-linear code:

- Say helper node i needs to send $y_i \in \mathbb{F}^\beta$ to v_1 in the non-constrained setting.
- There exists $l \times d\beta$ matrix $\mathcal{U}$:

$$\begin{bmatrix} c_{1,1} \\ c_{1,2} \\ \vdots \\ c_{1,l} \end{bmatrix} = \begin{bmatrix} \mathcal{U}_1 & \mathcal{U}_2 & \cdots & \mathcal{U}_d \end{bmatrix} \begin{bmatrix} y_{i_1} \\ y_{i_2} \\ \vdots \\ y_{i_d} \end{bmatrix}$$

- Node v_x that receives y_is from at least $d-k+1$ other nodes E can send
  - $\sum_{j \in E} \mathcal{U}_j y_{i_j} + \mathcal{U}_x y_{i_x} \longrightarrow l$ symbols instead of
  - $\{y_{i_j} : j \in E \cup \{x\}\} \longrightarrow |E \cup \{x\}|\beta$ symbols.

Adversarial Setting

Question : What if a part of the network is not trustworthy anymore?

Adversarial Setting

Question : What if a part of the network is not trustworthy anymore?

What we know

- In the fully connected setting, some results are known [Rashmi, Shah, Ramchandran, Vijay Kumar, 2012], [Ye and Barg, 2017], [Silberstein, Rawat, Vishwanath, 2015].

Adversarial Setting

Question : What if a part of the network is not trustworthy anymore?

What we know

- In the fully connected setting, some results are known [Rashmi, Shah, Ramchandran, Vijay Kumar, 2012], [Ye and Barg, 2017], [Silberstein, Rawat, Vishwanath, 2015].
- More complicated in the graph setting:

Adversarial Setting

Question : What if a part of the network is not trustworthy anymore?

What we know

- In the fully connected setting, some results are known [Rashmi, Shah, Ramchandran, Vijay Kumar, 2012], [Ye and Barg, 2017], [Silberstein, Rawat, Vishwanath, 2015].
- More complicated in the graph setting:
  - Different types of adversarial models possible: edge-controlling adversary, node-controlling adversary or both.

Adversarial Setting

Question : What if a part of the network is not trustworthy anymore?

What we know

- In the fully connected setting, some results are known [Rashmi, Shah, Ramchandran, Vijay Kumar, 2012], [Ye and Barg, 2017], [Silberstein, Rawat, Vishwanath, 2015].
- More complicated in the graph setting:
  - Different types of adversarial models possible: edge-controlling adversary, node-controlling adversary or both.
  - A small adversarial portion of the network can act as a bottleneck and thwart the entire repair task.

Adversarial Setting

Question : What if a part of the network is not trustworthy anymore?

What we know

- In the fully connected setting, some results are known [Rashmi, Shah, Ramchandran, Vijay Kumar, 2012], [Ye and Barg, 2017], [Silberstein, Rawat, Vishwanath, 2015].
- More complicated in the graph setting:
  - Different types of adversarial models possible: edge-controlling adversary, node-controlling adversary or both.
  - A small adversarial portion of the network can act as a bottleneck and thwart the entire repair task.
  - Errors spread due to the linear nature of current IP protocols.

Adversarial Setting

Question : What if a part of the network is not trustworthy anymore?

What we know

- In the fully connected setting, some results are known [Rashmi, Shah, Ramchandran, Vijay Kumar, 2012], [Ye and Barg, 2017], [Silberstein, Rawat, Vishwanath, 2015].
- More complicated in the graph setting:
  - Different types of adversarial models possible: edge-controlling adversary, node-controlling adversary or both.
  - A small adversarial portion of the network can act as a bottleneck and thwart the entire repair task.
  - Errors spread due to the linear nature of current IP protocols.
- Always possible to fall back to simple relaying (not do any Intermediate Processing) and use the existing constructions for full connectivity.

Adversarial Setting

Question : What if a part of the network is not trustworthy anymore?

What we know

- In the fully connected setting, some results are known [Rashmi, Shah, Ramchandran, Vijay Kumar, 2012], [Ye and Barg, 2017], [Silberstein, Rawat, Vishwanath, 2015].
- More complicated in the graph setting:
  - Different types of adversarial models possible: edge-controlling adversary, node-controlling adversary or both.
  - A small adversarial portion of the network can act as a bottleneck and thwart the entire repair task.
  - Errors spread due to the linear nature of current IP protocols.
- Always possible to fall back to simple relaying (not do any Intermediate Processing) and use the existing constructions for full connectivity.

Goal : Account for adversarial behavior without sacrificing the benefits of IP

Adversarial Setting: Solutions

Adversarial Setting: Solutions

- Edge-controlling adversary can be handled using local encoding and decoding at every node of the graph $\longrightarrow$ IP still possible with a multiplicative bandwidth overhead due to local encoding at every node.

Adversarial Setting: Solutions

- Edge-controlling adversary can be handled using local encoding and decoding at every node of the graph $\longrightarrow$ IP still possible with a multiplicative bandwidth overhead due to local encoding at every node.
- What about node-controlling adversary?

Adversarial Setting: Solutions

- Edge-controlling adversary can be handled using local encoding and decoding at every node of the graph $\longrightarrow$ IP still possible with a multiplicative bandwidth overhead due to local encoding at every node.
- What about node-controlling adversary?
- Consider a limited power adversary: *Can only corrupt the data stored in a helper node, can not influence the computations due to IP.*

Adversarial Setting: Solutions

- Edge-controlling adversary can be handled using local encoding and decoding at every node of the graph $\longrightarrow$ IP still possible with a multiplicative bandwidth overhead due to local encoding at every node.
- What about node-controlling adversary?
- Consider a limited power adversary: *Can only corrupt the data stored in a helper node, can not influence the computations due to IP.*
- Lower bounds and achievability?

Adversarial Setting: Lower Bounds

Adversarial Setting: Lower Bounds

Lemma

Suppose the data is encoded using an $[n, k, d, l, \beta, M]$ *MSR code on a graph. Let* v_f *be the failed node and* D *be the helper node set. Suppose that at most* t *nodes are controlled by a limited-power adversary and let* $E \subseteq D$ *be a subset of helper nodes containing them. If* $|E| \geqslant d - k + 1 + 2t$ *then*

$$c(E, v_f \cup D \setminus E) \geqslant l + 2t\beta.$$

Adversarial Setting: Lower Bounds

Lemma

Suppose the data is encoded using an $[n, k, d, l, \beta, M]$ MSR code on a graph. Let v_f be the failed node and D be the helper node set. Suppose that at most t nodes are controlled by a limited-power adversary and let $E \subseteq D$ be a subset of helper nodes containing them. If $|E| \geqslant d - k + 1 + 2t$ then

$$c(E, v_f \cup D \setminus E) \geqslant l + 2t\beta.$$

Proof Idea

Adversarial Setting: Lower Bounds

Lemma

Suppose the data is encoded using an $[n, k, d, l, \beta, M]$ MSR code on a graph. Let v_f be the failed node and D be the helper node set. Suppose that at most t nodes are controlled by a limited-power adversary and let $E \subseteq D$ be a subset of helper nodes containing them. If $|E| \geqslant d - k + 1 + 2t$ then

$$c(E, v_f \cup D \setminus E) \geqslant l + 2t\beta.$$

Proof Idea

- Use the network Singleton Bound.

Adversarial Setting: Lower Bounds

Lemma

Suppose the data is encoded using an $[n, k, d, l, \beta, M]$ MSR code on a graph. Let v_f be the failed node and D be the helper node set. Suppose that at most t nodes are controlled by a limited-power adversary and let $E \subseteq D$ be a subset of helper nodes containing them. If $|E| \geqslant d - k + 1 + 2t$ then

$$c(E, v_f \cup D \setminus E) \geqslant l + 2t\beta.$$

Proof Idea

- Use the network Singleton Bound.
- Form a directed acyclic graph from the given repair task.

Adversarial Setting: Lower Bounds

Lemma

Suppose the data is encoded using an $[n, k, d, l, \beta, M]$ *MSR code on a graph. Let* v_f *be the failed node and* D *be the helper node set. Suppose that at most* t *nodes are controlled by a limited-power adversary and let* $E \subseteq D$ *be a subset of helper nodes containing them. If* $|E| \geqslant d - k + 1 + 2t$ *then*

$$c(E, v_f \cup D \setminus E) \geqslant l + 2t\beta.$$

Proof Idea

- Use the network Singleton Bound.
- Form a directed acyclic graph from the given repair task.
- The failed node is the sink.

Adversarial Setting: Lower Bounds

Lemma

Suppose the data is encoded using an $[n, k, d, l, \beta, M]$ MSR code on a graph. Let v_f be the failed node and D be the helper node set. Suppose that at most t nodes are controlled by a limited-power adversary and let $E \subseteq D$ be a subset of helper nodes containing them. If $|E| \geqslant d - k + 1 + 2t$ then

$$c(E, v_f \cup D \setminus E) \geqslant l + 2t\beta.$$

Proof Idea

- Use the network Singleton Bound.
- Form a directed acyclic graph from the given repair task.
- The failed node is the sink.
- The set E jointly is the source.

Adversarial Setting: Lower Bounds

Lemma

Suppose the data is encoded using an $[n, k, d, l, \beta, M]$ MSR code on a graph. Let v_f be the failed node and D be the helper node set. Suppose that at most t nodes are controlled by a limited-power adversary and let $E \subseteq D$ be a subset of helper nodes containing them. If $|E| \geqslant d - k + 1 + 2t$ then

$$c(E, v_f \cup D \setminus E) \geqslant l + 2t\beta.$$

Proof Idea

- Use the network Singleton Bound.
- Form a directed acyclic graph from the given repair task.
- The failed node is the sink.
- The set E jointly is the source.
- The adversary can introduce at most $t\beta$ errors.

Adversarial Setting: Lower Bounds

Lemma

Suppose the data is encoded using an $[n, k, d, l, \beta, M]$ *MSR code on a graph. Let* v_f *be the failed node and* D *be the helper node set. Suppose that at most* t *nodes are controlled by a limited-power adversary and let* $E \subseteq D$ *be a subset of helper nodes containing them. If* $|E| \geqslant d - k + 1 + 2t$ *then*

$$c(E, v_f \cup D \setminus E) \geqslant l + 2t\beta.$$

Proof Idea

- Use the network Singleton Bound.
- Form a directed acyclic graph from the given repair task.
- The failed node is the sink.
- The set E jointly is the source.
- The adversary can introduce at most $t\beta$ errors.
- The set E needs to convey the message R_E^f to the sink with $H(R_E^f) \geqslant l$ by the previous lemma.

Adversarial Setting: Code Construction

Adversarial Setting: Code Construction

- Goal: Correct errors while also doing IP.

Adversarial Setting: Code Construction

- Goal: Correct errors while also doing IP.
- Idea: A receiving node does IP by doing linear combinations of symbols received and then forwards it. In the limited power adversary model, the transformations are done faithfully. Hence, an adversarial node may introduce at most β errors which may get linearly combined along the way $\longrightarrow$ Rank metric codes is the way to go.

Adversarial Setting: Code Construction

- Goal: Correct errors while also doing IP.
- Idea: A receiving node does IP by doing linear combinations of symbols received and then forwards it. In the limited power adversary model, the transformations are done faithfully. Hence, an adversarial node may introduce at most β errors which may get linearly combined along the way $\longrightarrow$ Rank metric codes is the way to go.
- Take an $[N, K, D]$ Gabidulin Code $\mathcal{C}_1$. Take an $[n, k, d, l = N, \beta, M]$ systematic MSR code $\mathcal{C}_2$. Encode each coordinate by $\mathcal{C}_1$ and then encode overall by $\mathcal{C}_2$.

Adversarial Setting: Code Construction

- Goal: Correct errors while also doing IP.
- Idea: A receiving node does IP by doing linear combinations of symbols received and then forwards it. In the limited power adversary model, the transformations are done faithfully. Hence, an adversarial node may introduce at most β errors which may get linearly combined along the way $\longrightarrow$ Rank metric codes is the way to go.
- Take an $[N, K, D]$ Gabidulin Code $\mathcal{C}_1$. Take an $[n, k, d, l = N, \beta, M]$ systematic MSR code $\mathcal{C}_2$. Encode each coordinate by $\mathcal{C}_1$ and then encode overall by $\mathcal{C}_2$.
- Each systematic node now stores an N-length Gabidulin code-word.

Adversarial Setting: Code Construction

- Goal: Correct errors while also doing IP.
- Idea: A receiving node does IP by doing linear combinations of symbols received and then forwards it. In the limited power adversary model, the transformations are done faithfully. Hence, an adversarial node may introduce at most β errors which may get linearly combined along the way $\longrightarrow$ Rank metric codes is the way to go.
- Take an $[N, K, D]$ Gabidulin Code $\mathcal{C}_1$. Take an $[n, k, d, l = N, \beta, M]$ systematic MSR code $\mathcal{C}_2$. Encode each coordinate by $\mathcal{C}_1$ and then encode overall by $\mathcal{C}_2$.
- Each systematic node now stores an N-length Gabidulin code-word.
- Hence if $D \geqslant 2t\beta + 1$, the failed node (or any faithful node along the way) receives a Gabidulin code-word with at-most $t\beta$ rank errors, it will be able to correct them.

Adversarial Setting: Continued

Performance Analysis

Adversarial Setting: Continued

Performance Analysis

- The resulting code is not MSR anymore.

Adversarial Setting: Continued

Performance Analysis

- The resulting code is not MSR anymore.
- Rate of the code becomes $\frac{kK}{nN}$ instead of the previous $\frac{k}{n}$.

Adversarial Setting: Continued

Performance Analysis

- The resulting code is not MSR anymore.
- Rate of the code becomes $\frac{kK}{nN}$ instead of the previous $\frac{k}{n}$.
- For the same value of $[n, k, d, M]$ let (l_{eff}, β_{eff}) be the values that meets the storage bandwidth trade-off with equality. Then

$$l_{eff} = l \cdot R_1, \quad \beta_{eff} = \beta \cdot R_1 \quad \text{where} \;\; R_1 = \frac{K}{N}.$$

Adversarial Setting: Continued

Performance Analysis

- The resulting code is not MSR anymore.
- Rate of the code becomes $\frac{kK}{nN}$ instead of the previous $\frac{k}{n}$.
- For the same value of $[n, k, d, M]$ let (l_{eff}, β_{eff}) be the values that meets the storage bandwidth trade-off with equality. Then

$$l_{eff} = l \cdot R_1, \quad \beta_{eff} = \beta \cdot R_1 \quad \text{where} \;\; R_1 = \frac{K}{N}.$$

- Any set E of size at least $d - k + 1 + 2t$ transmits l symbols: $l = l_{eff} + 2t\beta = l_{eff} + 2t\beta_{eff} \cdot \frac{1}{R_1}$.

Adversarial Setting: Continued

Performance Analysis

- The resulting code is not MSR anymore.
- Rate of the code becomes $\frac{kK}{nN}$ instead of the previous $\frac{k}{n}$.
- For the same value of $[n, k, d, M]$ let (l_{eff}, β_{eff}) be the values that meets the storage bandwidth trade-off with equality. Then

$$l_{eff} = l \cdot R_1, \quad \beta_{eff} = \beta \cdot R_1 \quad \text{where} \quad R_1 = \frac{K}{N}.$$

- Any set E of size at least $d - k + 1 + 2t$ transmits l symbols: $l = l_{eff} + 2t\beta = l_{eff} + 2t\beta_{eff} \cdot \frac{1}{R_1}$.
- The overhead in communication complexity is a constant multiple $(\frac{1}{R_1})$ of the optimal overhead.

Adversarial Setting: Continued

Performance Analysis

- The resulting code is not MSR anymore.
- Rate of the code becomes $\frac{kK}{nN}$ instead of the previous $\frac{k}{n}$.
- For the same value of $[n, k, d, M]$ let (l_{eff}, β_{eff}) be the values that meets the storage bandwidth trade-off with equality. Then

$$l_{eff} = l \cdot R_1, \quad \beta_{eff} = \beta \cdot R_1 \quad \text{where} \quad R_1 = \frac{K}{N}.$$

- Any set E of size at least $d - k + 1 + 2t$ transmits l symbols: $l = l_{eff} + 2t\beta = l_{eff} + 2t\beta_{eff} \cdot \frac{1}{R_1}$.
- The overhead in communication complexity is a constant multiple $(\frac{1}{R_1})$ of the optimal overhead.

All Powerful Adversary

Adversarial Setting: Continued

Performance Analysis

- The resulting code is not MSR anymore.
- Rate of the code becomes $\frac{kK}{nN}$ instead of the previous $\frac{k}{n}$.
- For the same value of $[n, k, d, M]$ let (l_{eff}, β_{eff}) be the values that meets the storage bandwidth trade-off with equality. Then

$$l_{eff} = l \cdot R_1, \quad \beta_{eff} = \beta \cdot R_1 \quad \text{where} \quad R_1 = \frac{K}{N}.$$

- Any set E of size at least $d - k + 1 + 2t$ transmits l symbols: $l = l_{eff} + 2t\beta = l_{eff} + 2t\beta_{eff} \cdot \frac{1}{R_1}$.
- The overhead in communication complexity is a constant multiple $(\frac{1}{R_1})$ of the optimal overhead.

All Powerful Adversary

- More difficult to handle because an adversary of this type can change all symbols being transmitted through it.

Adversarial Setting: Continued

Performance Analysis

- The resulting code is not MSR anymore.
- Rate of the code becomes $\frac{kK}{nN}$ instead of the previous $\frac{k}{n}$.
- For the same value of $[n, k, d, M]$ let (l_{eff}, β_{eff}) be the values that meets the storage bandwidth trade-off with equality. Then

$$l_{eff} = l \cdot R_1, \quad \beta_{eff} = \beta \cdot R_1 \quad \text{where} \quad R_1 = \frac{K}{N}.$$

- Any set E of size at least $d - k + 1 + 2t$ transmits l symbols: $l = l_{eff} + 2t\beta = l_{eff} + 2t\beta_{eff} \cdot \frac{1}{R_1}$.
- The overhead in communication complexity is a constant multiple $(\frac{1}{R_1})$ of the optimal overhead.

All Powerful Adversary

- More difficult to handle because an adversary of this type can change all symbols being transmitted through it.
- If the total number of such nodes is limited, above construction still works with sufficiently large rank-metric distance.


ANY
QUESTIONS