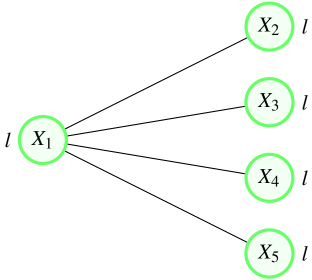
Regenerating Codes on Graphs

Adway Patra & Alexander Barg

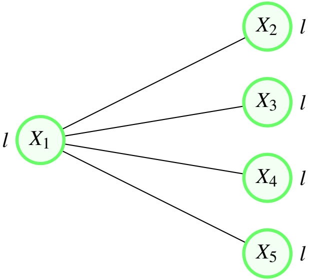
(University of Maryland)

ISIT 2021

Node repair in distributed storage

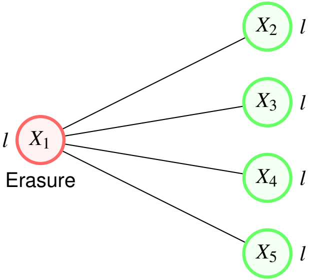


Node repair in distributed storage



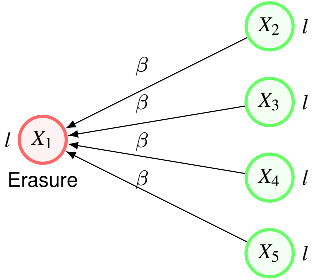
- Each codeword symbol stored in a node.

Node repair in distributed storage



- Each codeword symbol stored in a node.
- Correct erasures while trying to minimize total data "moved".

Node repair in distributed storage



- 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

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

Moving away from traditional setting

- General problem assumes degree at least d for all nodes, i.e., high connectivity.

Moving away from traditional setting

- General problem assumes degree at least d for all nodes, i.e., high connectivity.
- What can be done for sparsely connected graphs?

Moving away from traditional setting

- General problem assumes degree at least d for all nodes, i.e., high connectivity.
- What can be done for sparsely connected graphs?
  - Simple answer: Relaying of repair data.

Moving away from traditional setting

- General problem assumes degree at least d for all nodes, i.e., high connectivity.
- What can be done for sparsely connected graphs?
  - Simple answer: Relaying of repair data.
  - Choose a suitable spanning tree, having the failed node at the root.

Moving away from traditional setting

- General problem assumes degree at least d for all nodes, i.e., high connectivity.
- What can be done for sparsely connected graphs?
    - Simple answer: Relaying of repair data.
    - Choose a suitable spanning tree, having the failed node at the root.
    - Each node forwards the helper data of its descendants, possibly supplementing with its own.

Moving away from traditional setting

- General problem assumes degree at least d for all nodes, i.e., high connectivity.
- What can be done for sparsely connected graphs?
  - Simple answer: Relaying of repair data.
  - Choose a suitable spanning tree, having the failed node at the root.
  - Each node forwards the helper data of its descendants, possibly supplementing with its own.
- Same data gets transmitted multiple times.

Moving away from traditional setting

- General problem assumes degree at least d for all nodes, i.e., high connectivity.
- What can be done for sparsely connected graphs?
  - Simple answer: Relaying of repair data.
  - Choose a suitable spanning tree, having the failed node at the root.
  - Each node forwards the helper data of its descendants, possibly supplementing with its own.
- Same data gets transmitted multiple times.
- Total required communication depends on the structure of the tree.

Moving away from traditional setting

- General problem assumes degree at least d for all nodes, i.e., high connectivity.
- What can be done for sparsely connected graphs?
  - Simple answer: Relaying of repair data.
  - Choose a suitable spanning tree, having the failed node at the root.
  - Each node forwards the helper data of its descendants, possibly supplementing with its own.
- Same data gets transmitted multiple times.
- Total required communication depends on the structure of the tree.

Question 1 : Is it possible to process the data to reduce communication?

Moving away from traditional setting

- General problem assumes degree at least d for all nodes, i.e., high connectivity.
- What can be done for sparsely connected graphs?
  - Simple answer: Relaying of repair data.
  - Choose a suitable spanning tree, having the failed node at the root.
  - Each node forwards the helper data of its descendants, possibly supplementing with its own.
- Same data gets transmitted multiple times.
- Total required communication depends on the structure of the tree.

Question 1 : Is it possible to process the data to reduce communication?

Question 2 : If so, then to what extent?

The bounds: How much can we process?

- Node v_i holds random variable X_i.

The bounds: How much can we process?

- Node v_i holds random variable X_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 X_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 the MSR point: $H(X_f) = l$, $H(S_i^f) = \beta = \frac{l}{d-k+1}$.

The bounds: How much can we process?

- Node v_i holds random variable X_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 the MSR point: $H(X_f) = l$, $H(S_i^f) = \beta = \frac{l}{d-k+1}$.
- $H(S_i^f|X_i) = 0, H(X_f|S_1^f, \cdots, S_d^f) = 0.$

The bounds: How much can we process?

- Node v_i holds random variable X_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 the MSR point: $H(X_f) = l$, $H(S_i^f) = \beta = \frac{l}{d-k+1}$.
- $H(S_i^f | X_i) = 0, H(X_f | S_1^f, \cdots, S_d^f) = 0.$

Lemma

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

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

1) If $|E| \geqslant d-k+1$, then

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

2) If $|E| \leqslant d-k$, then

$$H(R_E^f) \geqslant \frac{|E| l}{d-k+1}.$$

Lower bound on communication

Lower bound on communication

Definition

Repair graph: The subgraph spanned by the failed node and d helper nodes closest to it in terms of graph distance.

$\Gamma_i(v_f)$: set of helper nodes at distance i from the failed node.

Lower bound on communication

Definition

Repair graph: The subgraph spanned by the failed node and d helper nodes closest to it in terms of graph distance.

$\Gamma_i(v_f)$: set of helper nodes at distance i from the failed node.

Proposition

Let R_j^f be the random variable denoting the information flow from the j-th layer to the $(j-1)$-th layer. Then

$$H(R_j^f) \geqslant \min\left\{l, \frac{|\cup_{i=j}^{t}\Gamma_i(v_f)| \cdot l}{d-k+1}\right\}$$

Lower bound on communication

Definition

Repair graph: The subgraph spanned by the failed node and d helper nodes closest to it in terms of graph distance.

$\Gamma_i(v_f)$: set of helper nodes at distance i from the failed node.

Proposition

Let R_j^f be the random variable denoting the information flow from the j-th layer to the $(j-1)$-th layer. Then

$$H(R_j^f) \geqslant \min\left\{l, \frac{|\cup_{i=j}^t \Gamma_i(v_f)| \cdot l}{d-k+1}\right\}$$

Proof.

Take $E = \cup_{i=j}^t \Gamma_i(v_f)$. □

Lower bound on communication

If the repair graph is a tree, then

Lower bound on communication

If the repair graph is a tree, then

Proposition

Let $J_f = \{v \in V(T_f)\backslash\{v_f\} : |D^*(v)| \geqslant d-k+2\}$. The total communication complexity β for the repair of node v_f on the repair tree T_f is bounded as

$$\beta \geqslant \sum_{v \in J_f} l + \sum_{v \in V(T_f)\backslash(\{v_f\} \cup J_f)} \frac{|D^*(v)| l}{d-k+1}.$$

where $D^*(v)$: set of descendants of v including itself.

Lower bound on communication

If the repair graph is a tree, then

Proposition

Let $J_f = \{v \in V(T_f) \backslash \{v_f\} : |D^*(v)| \geqslant d - k + 2\}$. The total communication complexity β for the repair of node v_f on the repair tree T_f is bounded as

$$\beta \geqslant \sum_{v \in J_f} l + \sum_{v \in V(T_f) \backslash (\{v_f\} \cup J_f)} \frac{|D^*(v)| l}{d - k + 1}.$$

where $D^*(v)$: set of descendants of v including itself.

Proof.

□

Lower bound on communication

If the repair graph is a tree, then

Proposition

Let $J_f = \{v \in V(T_f) \backslash \{v_f\} : |D^*(v)| \geqslant d - k + 2\}$. The total communication complexity β for the repair of node v_f on the repair tree T_f is bounded as

$$\beta \geqslant \sum_{v \in J_f} l + \sum_{v \in V(T_f) \backslash (\{v_f\} \cup J_f)} \frac{|D^*(v)| l}{d - k + 1}.$$

where $D^*(v)$: set of descendants of v including itself.

Proof.

- For every non-root node $v \notin J_f$, we have $|D^*(v)| \leqslant d - k$.

□

Lower bound on communication

If the repair graph is a tree, then

Proposition

Let $J_f = \{v \in V(T_f) \backslash \{v_f\} : |D^*(v)| \geqslant d - k + 2\}$*. The total communication complexity* β *for the repair of node* v_f *on the repair tree* T_f *is bounded as*

$$\beta \geqslant \sum_{v \in J_f} l + \sum_{v \in V(T_f) \backslash (\{v_f\} \cup J_f)} \frac{|D^*(v)| l}{d - k + 1}.$$

where $D^*(v)$ *: set of descendants of* v *including itself.*

Proof.

- For every non-root node $v \notin J_f$, we have $|D^*(v)| \leqslant d - k$.
- Any outflow of information out of the subtree spanned by $D^*(v)$ passes through the node v.

□

Lower bound on communication

If the repair graph is a tree, then

Proposition

Let $J_f = \{v \in V(T_f) \backslash \{v_f\} : |D^*(v)| \geqslant d - k + 2\}$. *The total communication complexity* β *for the repair of node* v_f *on the repair tree* T_f *is bounded as*

$$\beta \geqslant \sum_{v \in J_f} l + \sum_{v \in V(T_f) \backslash (\{v_f\} \cup J_f)} \frac{|D^*(v)| l}{d - k + 1}.$$

where $D^*(v)$: *set of descendants of* v *including itself.*

Proof.

- For every non-root node $v \notin J_f$, we have $|D^*(v)| \leqslant d - k$.
- Any outflow of information out of the subtree spanned by $D^*(v)$ passes through the node v.
- Needs to transmit at least $|D^*(v)| \cdot l/(d - k + 1)$ symbols to its immediate parent in T_f.

□

Lower bound on communication

If the repair graph is a tree, then

Proposition

Let $J_f = \{v \in V(T_f) \backslash \{v_f\} : |D^*(v)| \geqslant d - k + 2\}$*. The total communication complexity* β *for the repair of node* v_f *on the repair tree* T_f *is bounded as*

$$\beta \geqslant \sum_{v \in J_f} l + \sum_{v \in V(T_f) \backslash (\{v_f\} \cup J_f)} \frac{|D^*(v)| l}{d - k + 1}.$$

where $D^*(v)$: *set of descendants of* v *including itself.*

Proof.

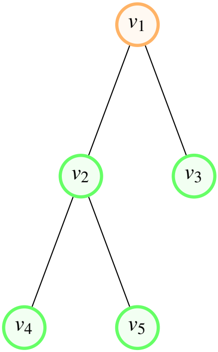
- For every non-root node $v \notin J_f$, we have $|D^*(v)| \leqslant d - k$.
- Any outflow of information out of the subtree spanned by $D^*(v)$ passes through the node v.
- Needs to transmit at least $|D^*(v)| \cdot l/(d - k + 1)$ symbols to its immediate parent in T_f.
- Every node $v \in J_f$ needs to transmit at least l symbols to its immediate parent.

□

Achieving the bounds: Using MSR Product Matrix codes

Example

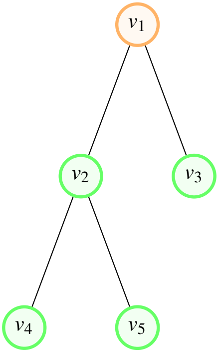
- Take an $[n = 5, k = 3, d = 4, l = 2, \beta = 1, B = 6]$ MSR Product Matrix code.



Achieving the bounds: Using MSR Product Matrix codes

Example

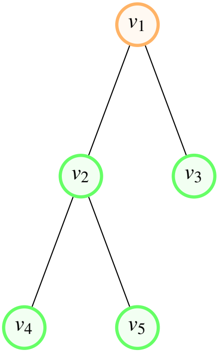
- Take an $[n = 5, k = 3, d = 4, l = 2, \beta = 1, B = 6]$ MSR Product Matrix code.
- $M = [S_1, S_2]^T = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \\ m_{21} & m_{22} \\ m_{22} & m_{23} \end{bmatrix}, \Psi_{5\times 4} = [\Phi, \Lambda\Phi], C = \Psi M.$



Achieving the bounds: Using MSR Product Matrix codes

Example

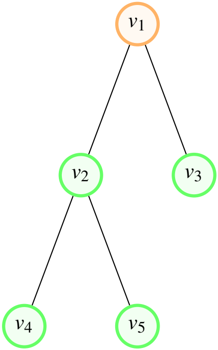
- Take an $[n = 5, k = 3, d = 4, l = 2, \beta = 1, B = 6]$ MSR Product Matrix code.
- $M = [S_1, S_2]^T = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \\ m_{21} & m_{22} \\ m_{22} & m_{23} \end{bmatrix}, \Psi_{5\times 4} = \left[\Phi, \Lambda\Phi\right], C = \Psi M.$
- Node i sends $y_i = \Psi_i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



Achieving the bounds: Using MSR Product Matrix codes

Example

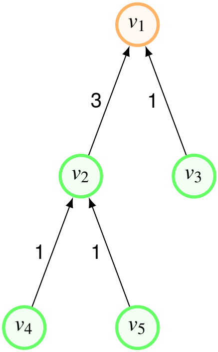
- Take an $[n=5, k=3, d=4, l=2, \beta=1, B=6]$ MSR Product Matrix code.
- $M = [S_1, S_2]^T = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \\ m_{21} & m_{22} \\ m_{22} & m_{23} \end{bmatrix}, \Psi_{5\times 4} = [\Phi, \Lambda\Phi], C = \Psi M.$
- Node i sends $y_i = \Psi_i M \Phi_1^T = (\Phi_i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



Achieving the bounds: Using MSR Product Matrix codes

Example

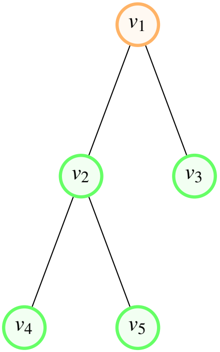
- Take an $[n=5, k=3, d=4, l=2, \beta=1, B=6]$ MSR Product Matrix code.
- $M = [S_1, S_2]^T = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \\ m_{21} & m_{22} \\ m_{22} & m_{23} \end{bmatrix}, \Psi_{5\times 4} = [\Phi, \Lambda\Phi], C = \Psi M.$
- Node i sends $y_i = \Psi_i M \Phi_1^T = (\Phi_i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



Achieving the bounds: Using MSR Product Matrix codes

Example

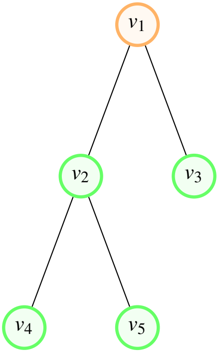
- Take an $[n=5, k=3, d=4, l=2, \beta=1, B=6]$ MSR Product Matrix code.
- $M = [S_1, S_2]^T = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \\ m_{21} & m_{22} \\ m_{22} & m_{23} \end{bmatrix}, \Psi_{5\times 4} = \left[\Phi, \Lambda\Phi\right], C = \Psi M.$
- Node i sends $y_i = \Psi_i M \Phi_1^T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



Achieving the bounds: Using MSR Product Matrix codes

Example

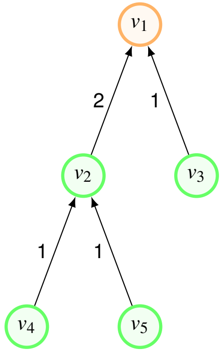
- Take an $[n = 5, k = 3, d = 4, l = 2, \beta = 1, B = 6]$ MSR Product Matrix code.
- $M = [S_1, S_2]^T = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \\ m_{21} & m_{22} \\ m_{22} & m_{23} \end{bmatrix}, \Psi_{5\times 4} = [\Phi, \Lambda\Phi], C = \Psi M.$
- Node i sends $y_i = \Psi_i M \Phi_1^T = (\Phi_i S_1 + \lambda_i \Phi_i S_2)\Phi_1^T$.
- 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 $U_4^{1,D} y_4 + U_5^{1,D} y_5 + U_2^{1,D} y_2$.



Achieving the bounds: Using MSR Product Matrix codes

Example

- Take an $[n = 5, k = 3, d = 4, l = 2, \beta = 1, B = 6]$ MSR Product Matrix code.
- $M = [S_1, S_2]^T = \begin{bmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \\ m_{21} & m_{22} \\ m_{22} & m_{23} \end{bmatrix}$, $\Psi_{5\times 4} = [\Phi, \Lambda\Phi]$, $C = \Psi M$.
- 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 $U_4^{1,D} y_4 + U_5^{1,D} y_5 + U_2^{1,D} y_2$.



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

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

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

In general, for any $\mathbb{F}$-linear MSR 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 MSR 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 MSR 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 MSR 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
  - $\sum_{j \in E} \mathcal{U}_j y_{i_j} + \mathcal{U}_x y_{i_x}$ ($l = (d - k + 1)\beta$ transmissions) in stead of

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

In general, for any $\mathbb{F}$-linear MSR 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
  - $\sum_{j \in E} \mathcal{U}_j y_{i_j} + \mathcal{U}_x y_{i_x}$ ($l = (d-k+1)\beta$ transmissions) in stead of
  - $\{y_{i_j} : j \in E \cup \{x\}\}$ ($|E \cup \{x\}|\beta \geqslant (d-k+2)\beta$ transmissions).

Application to random graphs

Application to random graphs

- Erdös-Rényi random graph $\mathcal{G}_{n,p}$: n nodes, each edge present with probability p independently of others.

Application to random graphs

- Erdös-Rényi random graph $\mathcal{G}_{n,p}$: n nodes, each edge present with probability p independently of others.
- An $[n, k, d]$ MSR code defined with C_i stored in node i.

Application to random graphs

- Erdös-Rényi random graph $\mathcal{G}_{n,p}$: n nodes, each edge present with probability p independently of others.
- An $[n, k, d]$ MSR code defined with C_i stored in node i.
- For asymptotically positive rates, $d = \Theta(n)$.

Application to random graphs

- Erdös-Rényi random graph $\mathcal{G}_{n,p}$: n nodes, each edge present with probability p independently of others.
- An $[n, k, d]$ MSR code defined with C_i stored in node i.
- For asymptotically positive rates, $d = \Theta(n)$.
- Operate in $p \gg \frac{\log n}{n}$ region, i.e., connected region.

Application to random graphs

- Erdös-Rényi random graph $\mathcal{G}_{n,p}$: n nodes, each edge present with probability p independently of others.
- An $[n, k, d]$ MSR code defined with C_i stored in node i.
- For asymptotically positive rates, $d = \Theta(n)$.
- Operate in $p \gg \frac{\log n}{n}$ region, i.e., connected region.
  - Node repair possible with high probability.

Application to random graphs

- Erdös-Rényi random graph $\mathcal{G}_{n,p}$: n nodes, each edge present with probability p independently of others.
- An $[n, k, d]$ MSR code defined with C_i stored in node i.
- For asymptotically positive rates, $d = \Theta(n)$.
- Operate in $p \gg \frac{\log n}{n}$ region, i.e., connected region.
  - Node repair possible with high probability.
- Threshold behavior: We say that *t-layer repair* of the failed node v *is possible* if

$$P(|N_t(v)| \geqslant d) \to 1 \text{ as } n \to \infty.$$

Application to random graphs: Repair threshold

Proposition

Let $d = \delta n, 0 < \delta < 1$ be a constant and let t be a fixed integer. Then t is the threshold depth for repair if

$$(np)^{t-1} = o(n), \quad p^t n^{t-1} - 2\log n \to \infty.$$

Application to random graphs: Repair threshold

Proposition

Let $d = \delta n, 0 < \delta < 1$ be a constant and let t be a fixed integer. Then t is the threshold depth for repair if

$$(np)^{t-1} = o(n), \quad p^t n^{t-1} - 2\log n \to \infty.$$

- Proof by classical results:

Application to random graphs: Repair threshold

Proposition

Let $d = \delta n, 0 < \delta < 1$ be a constant and let t be a fixed integer. Then t is the threshold depth for repair if

$$(np)^{t-1} = o(n), \quad p^t n^{t-1} - 2\log n \to \infty.$$

- Proof by classical results:
  - If the above conditions hold, then $\mathrm{diam}(G) = t$ with high probability [See Bollobas '81; Frieze and Karoński 2015].

Application to random graphs: Repair threshold

Proposition

Let $d = \delta n, 0 < \delta < 1$ be a constant and let t be a fixed integer. Then t is the threshold depth for repair if

$$(np)^{t-1} = o(n), \quad p^t n^{t-1} - 2\log n \to \infty.$$

- Proof by classical results:
  - If the above conditions hold, then $\mathrm{diam}(G) = t$ with high probability [See Bollobas '81; Frieze and Karoński 2015].
  - If $(np)^{t-1} = o(n)$ and $d = \Theta(n)$, then with high probability, d nodes are not reached in $t-1$ layers [Chung et. al. 2001].

Application to random graphs: Repair bandwidth

- How much savings do we get by using the "Intermediate Processing" technique compared to simple "Accumulate and Forward" relaying?

Application to random graphs: Repair bandwidth

- How much savings do we get by using the "Intermediate Processing" technique compared to simple "Accumulate and Forward" relaying?

Proposition

The repair bandwidth β_{AF} *satisfies* $\mathbb{P}(\beta_{\mathrm{AF}} \geqslant td - o(n)) \to 1$

Application to random graphs: Repair bandwidth

- How much savings do we get by using the "Intermediate Processing" technique compared to simple "Accumulate and Forward" relaying?

Proposition

The repair bandwidth β_{AF} *satisfies* $\mathbb{P}(\beta_{\mathrm{AF}} \geqslant td - o(n)) \to 1$

- Most nodes are at distance t. For the AF setting, required bandwidth scaled by t compared to the directly connected scenario.

Application to random graphs: Repair bandwidth

- How much savings do we get by using the "Intermediate Processing" technique compared to simple "Accumulate and Forward" relaying?

Proposition

The repair bandwidth β_{AF} *satisfies* $\mathbb{P}(\beta_{\mathrm{AF}} \geqslant td - o(n)) \to 1$

- Most nodes are at distance t. For the AF setting, required bandwidth scaled by t compared to the directly connected scenario.

Theorem

Let t *be the threshold for repair. For* $d = \Theta(n)$ *let* $d - k = \chi(n)$ *be a function of* n *such that* $\chi(n)n^{s-1}p^s \to 0$ *where* $s \leqslant t - 1$ *is the largest integer for which this condition holds. Then* $\mathbb{P}(\beta_{\mathrm{IP}} \leqslant (t-s)d + o(n)) \to 1.$

Application to random graphs: Repair bandwidth

- How much savings do we get by using the "Intermediate Processing" technique compared to simple "Accumulate and Forward" relaying?

Proposition

The repair bandwidth β_{AF} *satisfies* $\mathbb{P}(\beta_{\mathrm{AF}} \geqslant td - o(n)) \to 1$

- Most nodes are at distance t. For the AF setting, required bandwidth scaled by t compared to the directly connected scenario.

Theorem

Let t *be the threshold for repair. For* $d = \Theta(n)$ *let* $d - k = \chi(n)$ *be a function of* n *such that* $\chi(n)n^{s-1}p^s \to 0$ *where* $s \leqslant t - 1$ *is the largest integer for which this condition holds. Then* $\mathbb{P}(\beta_{\mathrm{IP}} \leqslant (t-s)d + o(n)) \to 1.$

- Using Intermediate Processing, the scaling of the bandwidth can be brought down from t to $t - s$.

Concluding remarks

- In sparsely connected graphs, it is possible to do better than simple relaying.

Concluding remarks

- In sparsely connected graphs, it is possible to do better than simple relaying.
- The intermediate processing technique is applicable to all $\mathbb{F}$-linear MSR codes and achieves the minimum possible communication in some cases.

Concluding remarks

- In sparsely connected graphs, it is possible to do better than simple relaying.
- The intermediate processing technique is applicable to all $\mathbb{F}$-linear MSR codes and achieves the minimum possible communication in some cases.
- For random graphs $\mathcal{G}_{n,p}$, in certain regimes, intermediate processing can give significant reductions in communication overhead.