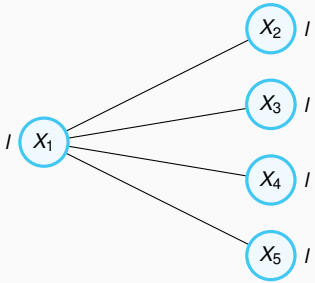
More results on Regenerating Codes on Graphs

International Symposium on Information Theory (ISIT) 2024, Athens

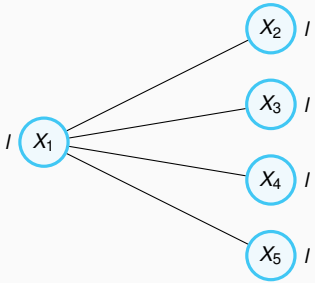
Adway Patra, Alexander Barg
University of Maryland, College Park

Introduction

Node repair in distributed storage

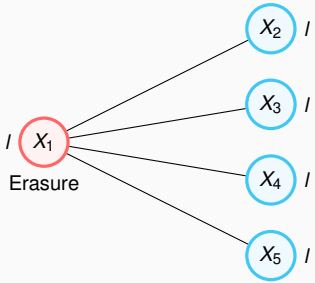


Node repair in distributed storage



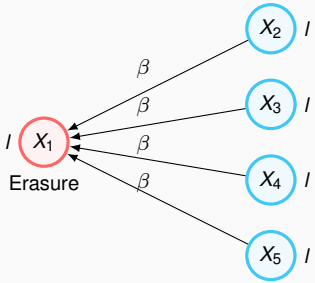
- An $[n, k, d, l, \beta, M]$ Regenerating Code $\mathcal{C} \subset \mathbb{F}^{nl}$, codewords viewed as $l \times n$ matrices over some finite field $\mathbb{F}$. Each codeword symbol stored in a node.

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

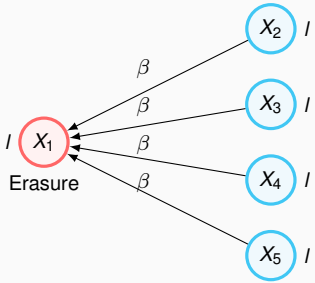


- An $[n, k, d, l, \beta, M]$ Regenerating Code $\mathcal{C} \subset \mathbb{F}^{nl}$, codewords viewed as $l \times n$ matrices over some finite field $\mathbb{F}$. Each codeword symbol stored in a node.
- Correct erasures while trying to minimize total data "moved".
- Total required transmission bounded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

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

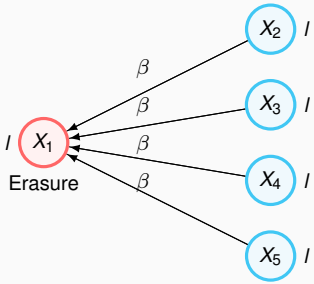
- Different pairs of (l, β) satisfying the above with equality give rise to different points on the storage-bandwidth trade-off; important ones are the MSR and MBR points.

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

Moving away from traditional setting

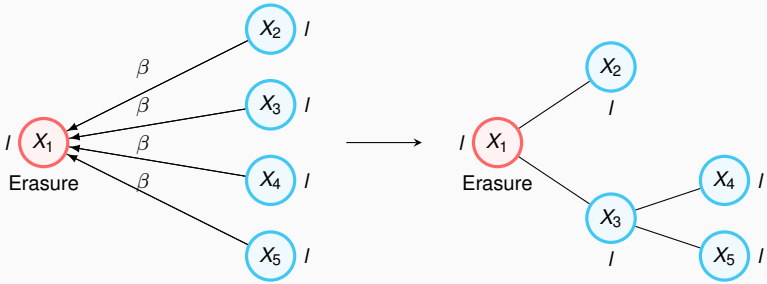
Moving away from traditional setting

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



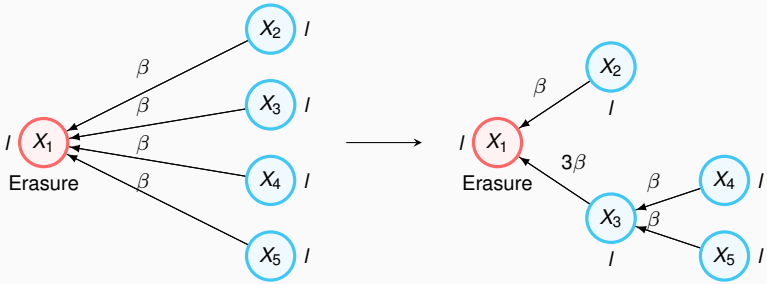
Moving away from traditional setting

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



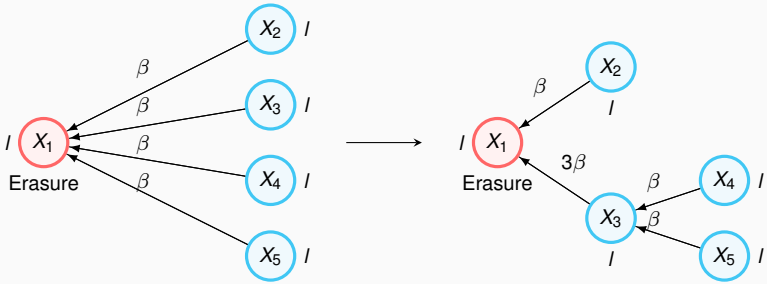
Moving away from traditional setting

- 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

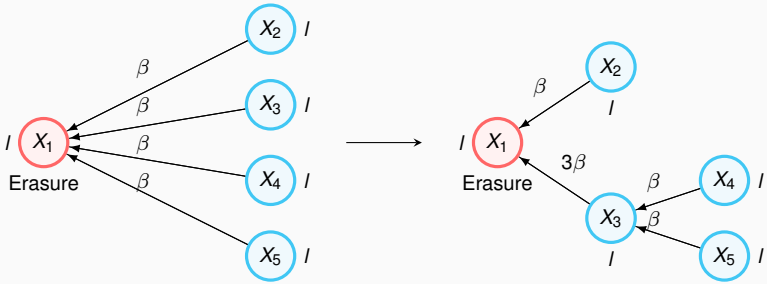
- General problem assumes d helper nodes are chosen from the direct neighbors of the failed node, i.e., high connectivity.



- Same data gets transmitted multiple times.

Moving away from traditional setting

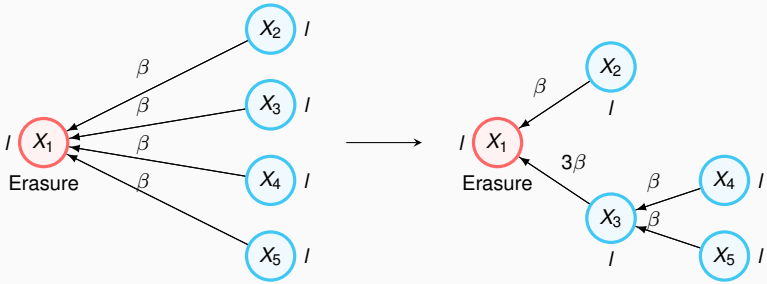
- General problem assumes d helper nodes are chosen from the direct neighbors of the failed node, i.e., high connectivity.



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

Moving away from traditional setting

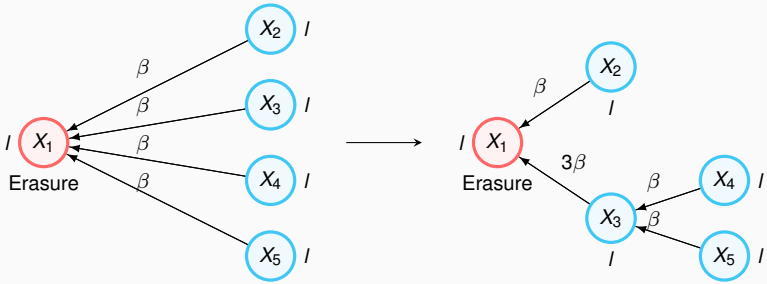
- General problem assumes d helper nodes are chosen from the direct neighbors of the failed node, i.e., high connectivity.



- 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?

Lower Bounds on Data Transmission

Lower Bounds on Data Transmission

- Node v_i holds random variable W_i.

Lower Bounds on Data Transmission

- 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.

Lower Bounds on Data Transmission

- 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: $H(W_f) = I$, $H(S_i^f) = \beta$.

Lower Bounds on Data Transmission

- 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: $H(W_f) = I$, $H(S_i^f) = \beta$.
- $H(S_i^f | W_i) = 0, H(W_f | S_1^f, \cdots, S_d^f) = 0$.

Lower Bounds on Data Transmission

- 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: $H(W_f) = I$, $H(S_i^f) = \beta$.
- $H(S_i^f | W_i) = 0, H(W_f | S_1^f, \cdots, S_d^f) = 0$.
- 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

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

Lower Bounds on Data Transmission

- 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: $H(W_f) = I$, $H(S_i^f) = \beta$.
- $H(S_i^f | W_i) = 0, H(W_f | S_1^f, \cdots, S_d^f) = 0$.
- 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

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

Lower Bounds on Data Transmission

- 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

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

Generalised Version [Patra and Barg, 2022]

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

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

Lower Bounds on Data Transmission

- 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: $H(W_f) = I$, $H(S_i^f) = \beta$.
- $H(S_i^f | W_i) = 0, H(W_f | S_1^f, \cdots, S_d^f) = 0$.
- 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

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

Generalised Version [Patra and Barg, 2022]

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

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

Corollary

For MSR Codes $H(R_E^f) \geqslant I$.

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

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

Achieving the bound: 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 bound: 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 bound: 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 bound: 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 bound: 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.

Are further savings possible in the above Graph Repair Setting?

Non-Uniform Contribution Model

Definitions

Definitions

Definition

Let $\mathcal{B} = \{\beta_i\}_{i=1}^{d}$ be a set of d positive integers. An $[n, k, d, l, \mathcal{B}, M]$ GRC encodes a file $\mathcal{F}$ of size M symbols over F by storing l symbols in each of the n nodes such that

1. (RECONSTRUCTION) by accessing any k out of n nodes, the original file can be recovered;
2. (REPAIR) the contents of any node $f \in [n]$ can be recovered by contacting a set $D \subseteq [n] \setminus \{f\}, |D| = d$ of nodes and downloading β_i symbols from node $\tau^{-1}(i)$ for any bijective mapping $\tau : D \to [d]$.

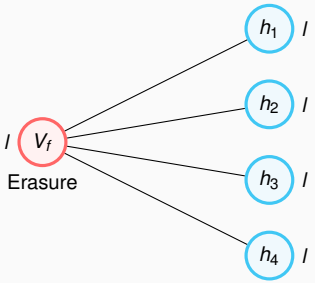
Definitions

Definition

Let $\mathcal{B} = \{\beta_i\}_{i=1}^{d}$ be a set of d positive integers. An $[n, k, d, l, \mathcal{B}, M]$ GRC encodes a file $\mathcal{F}$ of size M symbols over F by storing l symbols in each of the n nodes such that

1. (RECONSTRUCTION) by accessing any k out of n nodes, the original file can be recovered;
2. (REPAIR) the contents of any node $f \in [n]$ can be recovered by contacting a set $D \subseteq [n] \setminus \{f\}, |D| = d$ of nodes and downloading β_i symbols from node $\tau^{-1}(i)$ for any bijective mapping $\tau : D \to [d]$.

Example



Previous example:
$D = \{h_1, h_2, h_3, h_4\}, \mathcal{B} = \{\beta_1 \geqslant \beta_2 \geqslant \beta_3 \geqslant \beta_4\}, \tau \in S_4$.

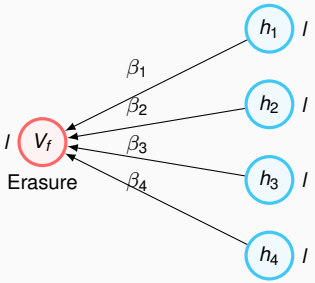
Definitions

Definition

Let $\mathcal{B} = \{\beta_i\}_{i=1}^{d}$ be a set of d positive integers. An $[n, k, d, l, \mathcal{B}, M]$ GRC encodes a file $\mathcal{F}$ of size M symbols over F by storing l symbols in each of the n nodes such that

1. (RECONSTRUCTION) by accessing any k out of n nodes, the original file can be recovered;
2. (REPAIR) the contents of any node $f \in [n]$ can be recovered by contacting a set $D \subseteq [n] \setminus \{f\}, |D| = d$ of nodes and downloading β_i symbols from node $\tau^{-1}(i)$ for any bijective mapping $\tau : D \to [d]$.

Example



Previous example:
$D = \{h_1, h_2, h_3, h_4\}, \mathcal{B} = \{\beta_1 \geqslant \beta_2 \geqslant \beta_3 \geqslant \beta_4\}, \tau \in S_4$.

$\tau := Id$

$1 \to 1$

$2 \to 2$

$3 \to 3$

$4 \to 4$

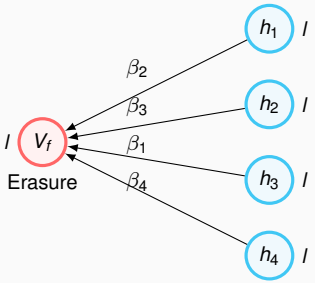
Definitions

Definition

Let $\mathcal{B} = \{\beta_i\}_{i=1}^{d}$ be a set of d positive integers. An $[n, k, d, l, \mathcal{B}, M]$ GRC encodes a file $\mathcal{F}$ of size M symbols over F by storing l symbols in each of the n nodes such that

1. (RECONSTRUCTION) by accessing any k out of n nodes, the original file can be recovered;
2. (REPAIR) the contents of any node $f \in [n]$ can be recovered by contacting a set $D \subseteq [n] \setminus \{f\}, |D| = d$ of nodes and downloading β_i symbols from node $\tau^{-1}(i)$ for any bijective mapping $\tau : D \to [d]$.

Example



Previous example:
$D = \{h_1, h_2, h_3, h_4\}, \mathcal{B} = \{\beta_1 \geqslant \beta_2 \geqslant \beta_3 \geqslant \beta_4\}, \tau \in S_4$.

$\tau := (123)$

$1 \to 2$

$2 \to 3$

$3 \to 1$

$4 \to 4$

Generalized Cut-set Bound

Generalized Cut-set Bound

Theorem

For an $[n, k, d, l, \mathcal{B}, M]$ *GRC,*

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

where $\Delta_r(\mathcal{B}) = \min_{R \subseteq [d], |R|=r} \sum_{i \in R} \beta_i$.

Generalized Cut-set Bound

Theorem

For an $[n, k, d, l, \mathcal{B}, M]$ *GRC,*

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

where $\Delta_r(\mathcal{B}) = \min_{R\subseteq[d],|R|=r} \sum_{i\in R} \beta_i$.

MSR Point

The *Minimum Storage* (MSR) point of the bound (1), is defined by $l = \Delta_{d-k+1}(\mathcal{B})$.

Generalized Cut-set Bound

Theorem

For an $[n, k, d, l, \mathcal{B}, M]$ *GRC,*

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

where $\Delta_r(\mathcal{B}) = \min_{R\subseteq[d],|R|=r} \sum_{i\in R} \beta_i$.

MSR Point

The *Minimum Storage* (MSR) point of the bound (1), is defined by $l = \Delta_{d-k+1}(\mathcal{B})$.

Lemma

Let $f \in [n]$ *be the failed node. For a (nonempty) subset of helper nodes* $E \subset D$*, let* R_E^f *be a function of* S_E^f *such that* $H(W_f|R_E^f, S_{D\setminus E}^f) = 0$*. Then if* $|E| \geqslant d - k + 1$*,*

$$H(R_E^f) \geqslant M - \sum_{i=0}^{k-2} \min\{l, \Delta_{d-i}(\mathcal{B})\}.$$

Generalized Cut-set Bound

Theorem

For an $[n, k, d, l, \mathcal{B}, M]$ *GRC,*

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

where $\Delta_r(\mathcal{B}) = \min_{R\subseteq[d],|R|=r} \sum_{i\in R} \beta_i$.

MSR Point

The *Minimum Storage* (MSR) point of the bound (1), is defined by $l = \Delta_{d-k+1}(\mathcal{B})$.

Lemma

Let $f \in [n]$ *be the failed node. For a (nonempty) subset of helper nodes* $E \subset D$*, let* R_E^f *be a function of* S_E^f *such that* $H(W_f|R_E^f, S_{D\setminus E}^f) = 0$*. Then if* $|E| \geqslant d - k + 1$*,*

$$H(R_E^f) \geqslant M - \sum_{i=0}^{k-2} \min\{l, \Delta_{d-i}(\mathcal{B})\}.$$

Corollary

For MSR codes, we have

$$H(R_E^f) \geqslant l = \Delta_{d-k+1}(\mathcal{B}). \tag{2}$$

Stacking Construction

Stacking Construction

Construction

1. *Take* $\mathcal{B} = \{\beta_j\}_{j=1}^{d}$ *non-decreasing,* $\mu_j = \mathbb{1}_{(\beta_j > \beta_{j-1})}$, $S := \{j : \mu_j = 1\}$.
2. *For each* $j \in S$ *take an MSR code* $\mathcal{C}_j$ *with parameters*

$$[n, k, d_j = d - j + 1,$$
$$l_j = (d - j - k + 2)(\beta_j - \beta_{j-1}),$$
$$(\beta_j - \beta_{j-1}), M_j = k l_j].$$

3. $[n, k, d, l, \mathcal{B}, M]$ *GRC code is formed by stacking the codes* $\{\mathcal{C}_j\}_{j \in S}$, *where* $l = \sum_{j \in S} l_j$ *and* $M = \sum_{j \in S} M_j$.

Stacking Construction

Construction

1. *Take* $\mathcal{B} = \{\beta_j\}_{j=1}^{d}$ *non-decreasing,* $\mu_j = \mathbb{1}_{(\beta_j > \beta_{j-1})}$, $S := \{j : \mu_j = 1\}$.
2. *For each* $j \in S$ *take an MSR code* $\mathcal{C}_j$ *with parameters*

$$[n, k, d_j = d - j + 1,$$
$$l_j = (d - j - k + 2)(\beta_j - \beta_{j-1}),$$
$$(\beta_j - \beta_{j-1}), M_j = k l_j].$$

3. $[n, k, d, l, \mathcal{B}, M]$ *GRC code is formed by stacking the codes* $\{\mathcal{C}_j\}_{j \in S}$, *where* $l = \sum_{j \in S} l_j$ *and* $M = \sum_{j \in S} M_j$.

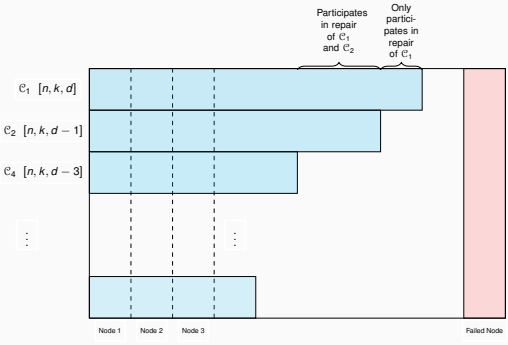


Figure 1: Stacking MSR codes

Stacking Construction

Construction

1. *Take* $\mathcal{B} = \{\beta_j\}_{j=1}^{d}$ *non-decreasing,* $\mu_j = \mathbb{1}_{(\beta_j > \beta_{j-1})}$, $S := \{j : \mu_j = 1\}$.
2. *For each* $j \in S$ *take an MSR code* $\mathcal{C}_j$ *with parameters*

$$[n, k, d_j = d - j + 1,$$
$$l_j = (d - j - k + 2)(\beta_j - \beta_{j-1}),$$
$$(\beta_j - \beta_{j-1}), M_j = kl_j].$$

3. $[n, k, d, l, \mathcal{B}, M]$ *GRC code is formed by stacking the codes* $\{\mathcal{C}_j\}_{j \in S}$*, where* $l = \sum_{j \in S} l_j$ *and* $M = \sum_{j \in S} M_j$.

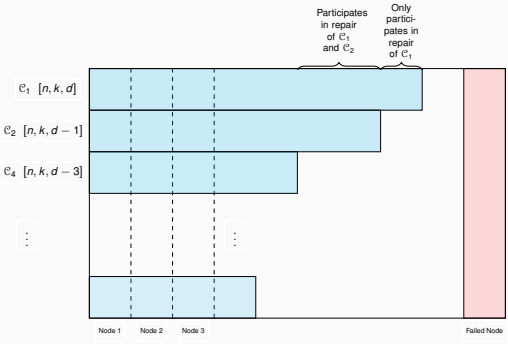


Figure 1: Stacking MSR codes

Remarks

Stacking Construction

Construction

1. *Take* $\mathcal{B} = \{\beta_j\}_{j=1}^{d}$ *non-decreasing,* $\mu_j = \mathbb{1}_{(\beta_j > \beta_{j-1})}$, $S := \{j : \mu_j = 1\}$.
2. *For each* $j \in S$ *take an MSR code* $\mathcal{C}_j$ *with parameters*

$$[n, k, d_j = d - j + 1,$$
$$l_j = (d - j - k + 2)(\beta_j - \beta_{j-1}),$$
$$(\beta_j - \beta_{j-1}), M_j = k l_j].$$

3. $[n, k, d, l, \mathcal{B}, M]$ *GRC code is formed by stacking the codes* $\{\mathcal{C}_j\}_{j \in S}$*, where* $l = \sum_{j \in S} l_j$ *and* $M = \sum_{j \in S} M_j$.

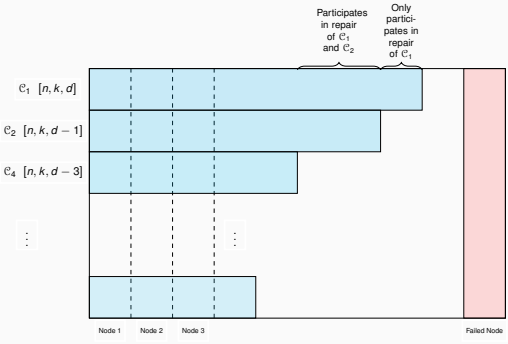


Figure 1: Stacking MSR codes

Remarks

- Node $\tau^{-1}(j)$ participates in the recovery of node f only in the component codes $\{\mathcal{C}_p : p \leqslant j\}$, $\sum_{p=1}^{j}(\beta_p - \beta_{p-1}) = \beta_j$ symbols.

Stacking Construction

Construction

1. *Take* $\mathcal{B} = \{\beta_j\}_{j=1}^{d}$ *non-decreasing,* $\mu_j = \mathbb{1}_{(\beta_j > \beta_{j-1})}$, $S := \{j : \mu_j = 1\}$.
2. *For each* $j \in S$ *take an MSR code* $\mathcal{C}_j$ *with parameters*

$$[n, k, d_j = d - j + 1,$$
$$l_j = (d - j - k + 2)(\beta_j - \beta_{j-1}),$$
$$(\beta_j - \beta_{j-1}), M_j = k l_j].$$

3. $[n, k, d, l, \mathcal{B}, M]$ *GRC code is formed by stacking the codes* $\{\mathcal{C}_j\}_{j \in S}$*, where* $l = \sum_{j \in S} l_j$ *and* $M = \sum_{j \in S} M_j$.

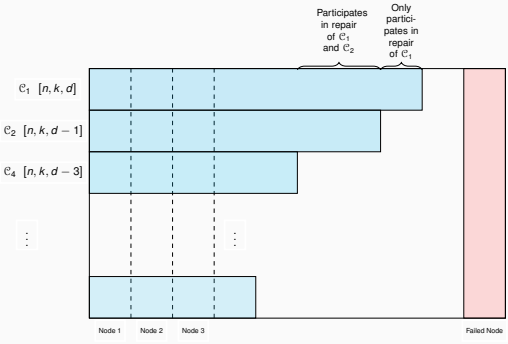


Figure 1: Stacking MSR codes

Remarks

- Node $\tau^{-1}(j)$ participates in the recovery of node f only in the component codes $\{\mathcal{C}_p : p \leqslant j\}$, $\sum_{p=1}^{j}(\beta_p - \beta_{p-1}) = \beta_j$ symbols.
- Meets the Generalized Cut-set bound with equality at the MSR point.

Stacking Construction

Construction

1. *Take* $\mathcal{B} = \{\beta_j\}_{j=1}^{d}$ *non-decreasing,* $\mu_j = \mathbb{1}_{(\beta_j > \beta_{j-1})}$, $S := \{j : \mu_j = 1\}$.
2. *For each* $j \in S$ *take an MSR code* $\mathcal{C}_j$ *with parameters*

$$[n, k, d_j = d - j + 1,$$
$$l_j = (d - j - k + 2)(\beta_j - \beta_{j-1}),$$
$$(\beta_j - \beta_{j-1}), M_j = k l_j].$$

3. $[n, k, d, l, \mathcal{B}, M]$ *GRC code is formed by stacking the codes* $\{\mathcal{C}_j\}_{j \in S}$*, where* $l = \sum_{j \in S} l_j$ *and* $M = \sum_{j \in S} M_j$.

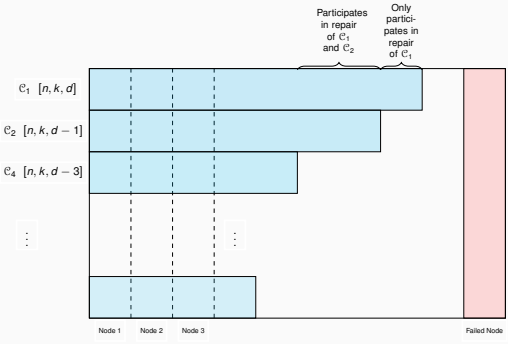


Figure 1: Stacking MSR codes

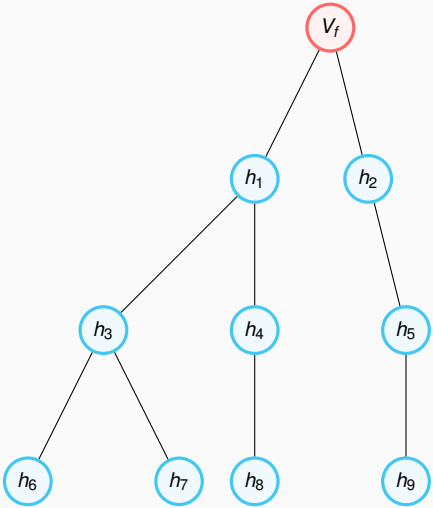
Remarks

- Node $\tau^{-1}(j)$ participates in the recovery of node f only in the component codes $\{\mathcal{C}_p : p \leqslant j\}$, $\sum_{p=1}^{j} (\beta_p - \beta_{p-1}) = \beta_j$ symbols.
- Meets the Generalized Cut-set bound with equality at the MSR point.
- Meets the IP repair bound with equality at the MSR point.

Graph Repair

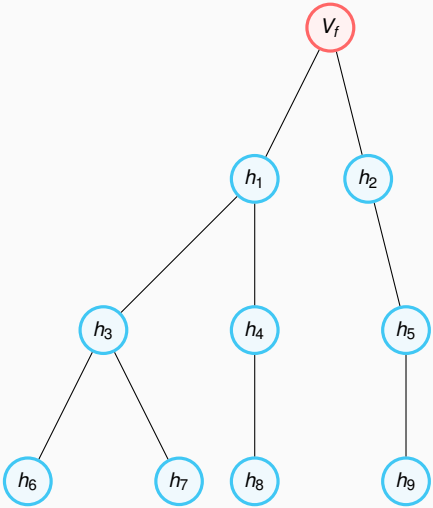
Non-Uniform Download from layers

Non-Uniform Download from layers



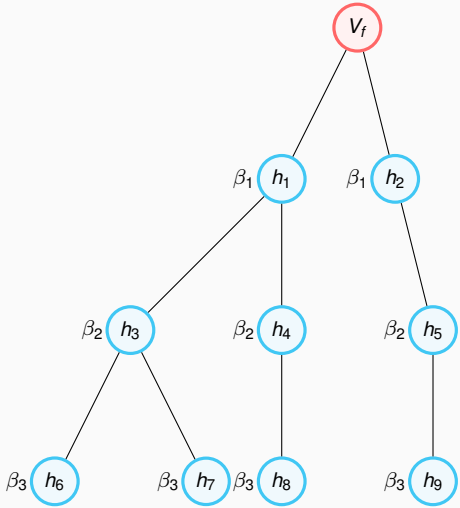
Non-Uniform Download from layers

- Helper nodes in layer i generate β_i symbols each for repair, $\beta_i \leqslant \beta_{i-1}$.

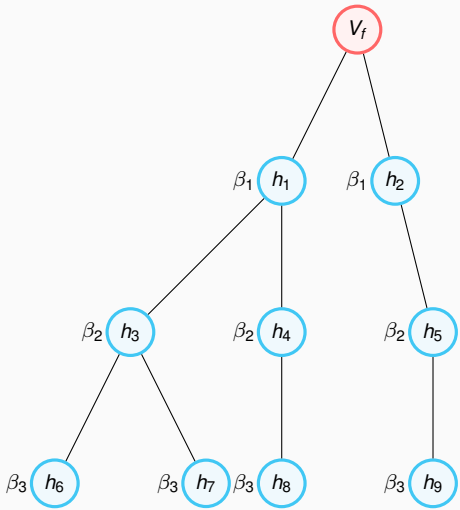


Non-Uniform Download from layers

- Helper nodes in layer i generate β_i symbols each for repair, $\beta_i \leqslant \beta_{i-1}$.

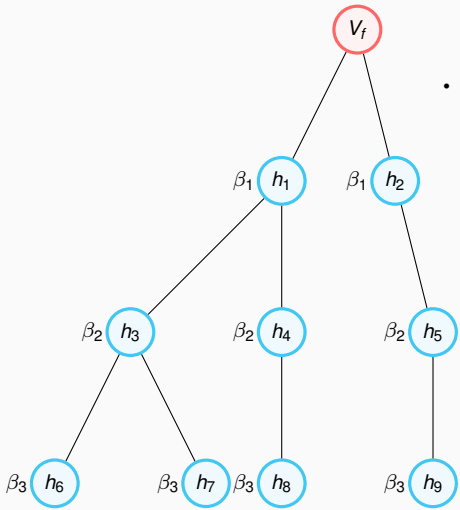


Non-Uniform Download from layers



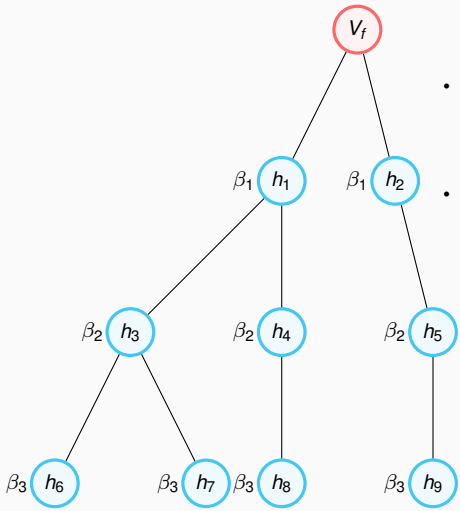
- Helper nodes in layer i generate β_i symbols each for repair, $\beta_i \leqslant \beta_{i-1}$.
- Let $J :=$ Set of nodes with at least $(d - k + 1)$ children in the tree, and for $i \notin J$ let $\mathcal{P}(i)$ denote the nearest parent of node i in J,

Non-Uniform Download from layers



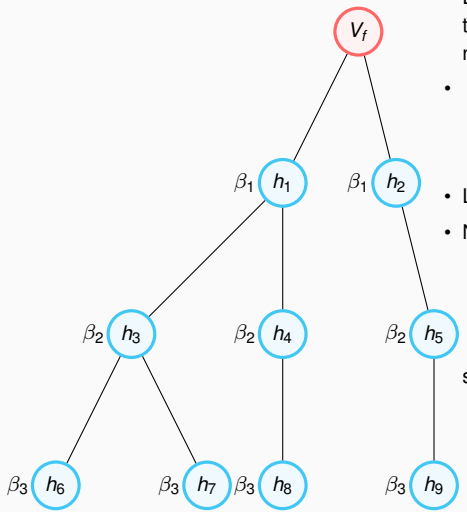
- Helper nodes in layer i generate β_i symbols each for repair, $\beta_i \leqslant \beta_{i-1}$.
- Let $J :=$ Set of nodes with at least $(d - k + 1)$ children in the tree, and for $i \notin J$ let $\mathcal{P}(i)$ denote the nearest parent of node i in J,
- $$\Lambda_{\text{NU}}^{\mathcal{T}} = \sum_{i \in J \setminus \{f\}} I + \sum_{i=1}^{t} \sum_{j \in D_i \setminus J_i} \rho(j, \mathcal{P}(j)) \beta_i.$$

Non-Uniform Download from layers



- Helper nodes in layer i generate β_i symbols each for repair, $\beta_i \leqslant \beta_{i-1}$.
- Let $J :=$ Set of nodes with at least $(d - k + 1)$ children in the tree, and for $i \notin J$ let $\mathcal{P}(i)$ denote the nearest parent of node i in J,
- $$\Lambda_{\text{NU}}^{\mathcal{T}} = \sum_{i \in J \setminus \{f\}} I + \sum_{i=1}^{t} \sum_{j \in D_i \setminus J_i} \rho(j, \mathcal{P}(j)) \beta_i.$$
- Let $\delta_i = \beta - \beta_i$.

Non-Uniform Download from layers



- Helper nodes in layer i generate β_i symbols each for repair, $\beta_i \leqslant \beta_{i-1}$.
- Let $J :=$ Set of nodes with at least $(d-k+1)$ children in the tree, and for $i \notin J$ let $\mathcal{P}(i)$ denote the nearest parent of node i in J,
- $$\Lambda_{\mathrm{NU}}^{\mathcal{T}} = \sum_{i \in J \setminus \{f\}} I + \sum_{i=1}^{t} \sum_{j \in D_i \setminus J_i} \rho(j, \mathcal{P}(j)) \beta_i.$$
- Let $\delta_i = \beta - \beta_i$.
- Non-Uniform Download is advantageous whenever:

$$\sum_{i=1}^{t} \sum_{j \in D_i \setminus J_i} \rho(j, \mathcal{P}(j)) \delta_i > 0$$

subject to

$$\sum_{i=t}^{t'+1} d_i \delta_i + \left(d - k + 1 - \sum_{i=t}^{t'+1} d_i\right) \delta_{t'} = 0.$$

where t' is the largest number such that $\sum_{i=t}^{t'} d_i \geqslant d - k + 1$.

Varying the Repair Degree

Varying the Repair Degree

- Lowering the contributions of some nodes at the expense of others may reduce total complexity.

Varying the Repair Degree

- Lowering the contributions of some nodes at the expense of others may reduce total complexity.
- In the limit some nodes will stop contributing $\rightarrow$ reducing the repair degree.

Varying the Repair Degree

- Lowering the contributions of some nodes at the expense of others may reduce total complexity.
- In the limit some nodes will stop contributing $\rightarrow$ reducing the repair degree.
- Code families exist that support multiple values of repair degree simultaneously: Universal d codes [Ye and Barg, 2017].

Varying the Repair Degree

- Lowering the contributions of some nodes at the expense of others may reduce total complexity.
- In the limit some nodes will stop contributing $\rightarrow$ reducing the repair degree.
- Code families exist that support multiple values of repair degree simultaneously: Universal d codes [Ye and Barg, 2017].
- Finding the optimal $d \rightarrow$ Solve the following optimization problem [Li, Mow, Deng and Wu, 2022]:

$$\min \sum_{i \in D \setminus J} b_i \beta_i$$

$$\text{subject to} \sum_{i \in A} \beta_i \geqslant I, \quad \forall A \subseteq [n-1], |A| = n-k$$

$$0 \leqslant \beta_i \leqslant I, \quad i \in [n-1],$$

where the costs b_i can be calculated from the structure of the graph.

Varying the Repair Degree

- Lowering the contributions of some nodes at the expense of others may reduce total complexity.
- In the limit some nodes will stop contributing $\rightarrow$ reducing the repair degree.
- Code families exist that support multiple values of repair degree simultaneously: Universal d codes [Ye and Barg, 2017].
- Finding the optimal $d \rightarrow$ Solve the following optimization problem [Li, Mow, Deng and Wu, 2022]:

$$\min \sum_{i \in D \setminus J} b_i \beta_i$$

$$\text{subject to} \sum_{i \in A} \beta_i \geqslant I, \quad \forall A \subseteq [n-1], |A| = n-k$$

$$0 \leqslant \beta_i \leqslant I, \quad i \in [n-1],$$

  where the costs b_i can be calculated from the structure of the graph.

- **Theorem:** There exists an optimal solution with uniform downloads for some repair degree d.

Varying the Repair Degree: Example

Example

Varying the Repair Degree: Example

Example

- Consider a r regular graph with t repair layers: $d_i = r(r-1)^{i-1}, 1 \leqslant i \leqslant (t-1), d_t = d - \sum_{i=1}^{t-1} d_i$.

Varying the Repair Degree: Example

Example

- Consider a r regular graph with t repair layers: $d_i = r(r-1)^{i-1}, 1 \leqslant i \leqslant (t-1), d_t = d - \sum_{i=1}^{t-1} d_i$.
- Suppose further that $d_t + d_{t-1} \geqslant (d - k + 1)$, and no IP.

Varying the Repair Degree: Example

Example

- Consider a r regular graph with t repair layers: $d_i = r(r-1)^{i-1}, 1 \leqslant i \leqslant (t-1), d_t = d - \sum_{i=1}^{t-1} d_i$.
- Suppose further that $d_t + d_{t-1} \geqslant (d - k + 1)$, and no IP.
- Nodes in layer i contribute β_i symbols each, $\beta_i \leqslant \beta_{i-1}$.

Varying the Repair Degree: Example

Example

- Consider a r regular graph with t repair layers: $d_i = r(r-1)^{i-1}, 1 \leqslant i \leqslant (t-1), d_t = d - \sum_{i=1}^{t-1} d_i$.
- Suppose further that $d_t + d_{t-1} \geqslant (d - k + 1)$, and no IP.
- Nodes in layer i contribute β_i symbols each, $\beta_i \leqslant \beta_{i-1}$.
- The savings with the non-uniform contributions scheme: (Recall $\delta_i = \beta - \beta_i$)

$$\begin{aligned}\Lambda_{\mathrm{U}}^{\mathfrak{T}} - \Lambda_{\mathrm{NU}}^{\mathfrak{T}} &= \sum_{i=1}^{t} i d_i \delta_i \\ &= \frac{d_t \delta_t}{d - k + 1 - d_t} \Big(\sum_{i=1}^{t-1} (t-i) d_i - t(k-1) \Big)\end{aligned}$$

with

$$d_t \delta_t + (d - k + 1 - d_t)\delta_{t-1} = 0.$$

Varying the Repair Degree: Example

Example

- Consider a r regular graph with t repair layers: $d_i = r(r-1)^{i-1}, 1 \leqslant i \leqslant (t-1), d_t = d - \sum_{i=1}^{t-1} d_i$.
- Suppose further that $d_t + d_{t-1} \geqslant (d-k+1)$, and no IP.
- Nodes in layer i contribute β_i symbols each, $\beta_i \leqslant \beta_{i-1}$.
- The savings with the non-uniform contributions scheme: (Recall $\delta_i = \beta - \beta_i$)

$$\begin{aligned}\Lambda_{\mathrm{U}}^{\mathfrak{T}} - \Lambda_{\mathrm{NU}}^{\mathfrak{T}} &= \sum_{i=1}^{t} i d_i \delta_i \\ &= \frac{d_t \delta_t}{d-k+1-d_t}\Big(\sum_{i=1}^{t-1}(t-i)d_i - t(k-1)\Big)\end{aligned}$$

with

$$d_t\delta_t + (d-k+1-d_t)\delta_{t-1} = 0.$$

- Extreme case:

$$\begin{aligned}&\delta_t = \beta \implies \beta_t = 0 \\ \implies &\delta_{t-1} = -\frac{d_t\beta}{d-k+1-d_t} \\ \implies &\beta_{t-1} = \beta + \frac{d_t\beta}{d-k+1-d_t} = \frac{l}{d'-k+1}, \quad d' = d - d_t\end{aligned}$$

Conclusions

Conclusions

- If the repair degree is fixed by design and the graph is sufficiently regular, **non-uniform download schemes based on the stacking construction** may reduce the communication complexity of repair.

Conclusions

- If the repair degree is fixed by design and the graph is sufficiently regular, **non-uniform download schemes based on the stacking construction** may reduce the communication complexity of repair.

- If we are free to adjust the repair degree in different instances, use **universal-*d*** code families with uniform download.

Conclusions

- If the repair degree is fixed by design and the graph is sufficiently regular, **non-uniform download schemes based on the stacking construction** may reduce the communication complexity of repair.
- If we are free to adjust the repair degree in different instances, use **universal-*d*** code families with uniform download.
- In both cases, **Intermediate Processing** of information further reduces the communication complexity.

Questions?