Bringing Reliability to the Cloud: Regenerating Codes for Distributed Storage

Nihar B. Shah[1], K.V. Rashmi [1], P. Vijay Kumar[1], Kannan Ramchandran[2]

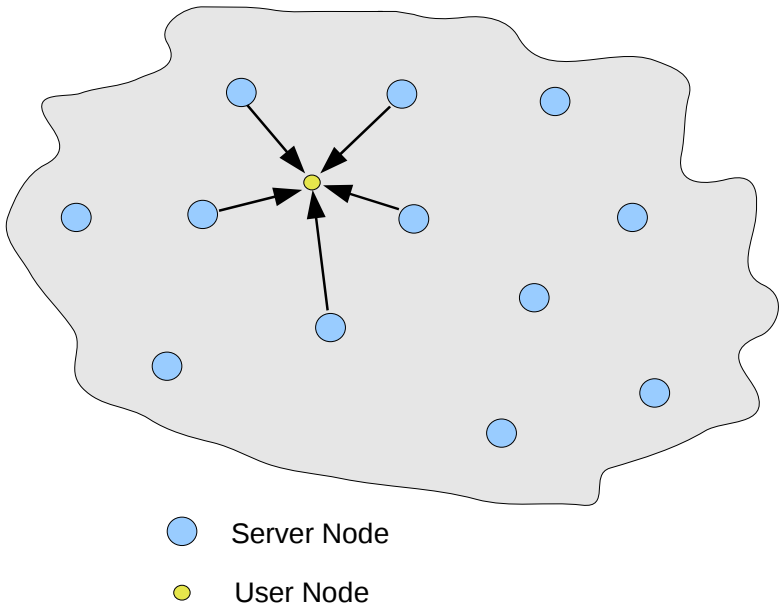
[1]Indian Institute Of Science, Bangalore, [2] University of California, Berkeley

Chinese University of Hong Kong
Hong Kong, Sep. 21, 2010

Outline

Outline

Distributed Data Storage Network



- Information pertaining to a file is dispersed across data centers in the network
- user taps into surrounding servers to retrieve data

Network Assumptions

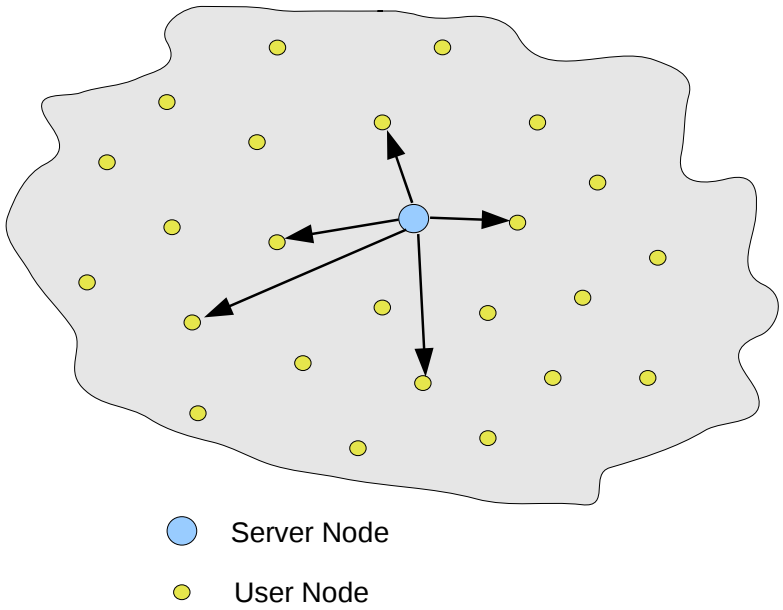
- Nodes are prone to failure (nodes down for maintenance, nodes coming and going)
- Would like to minimize the total amount of storage for a desired level of performance
- Desirable to avoid congestion across links in the network
- Restrict attention to linear operations at the individual nodes

An Example Node: A Large Data Center

File Size and Alphabet

- Data file B over an alphabet $\mathcal{A}$ of of size q
- Eg. $\mathcal{A} = \mathbb{F}_q$ for example

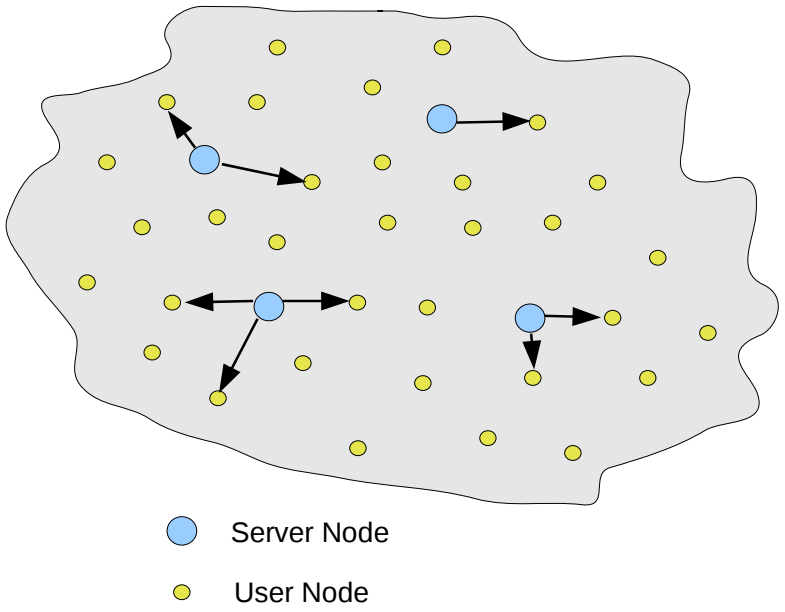
Single-Server Option



All data hosted on a single server

- vulnerable to even a single node failure
- links surrounding the server node will be subject to congestion

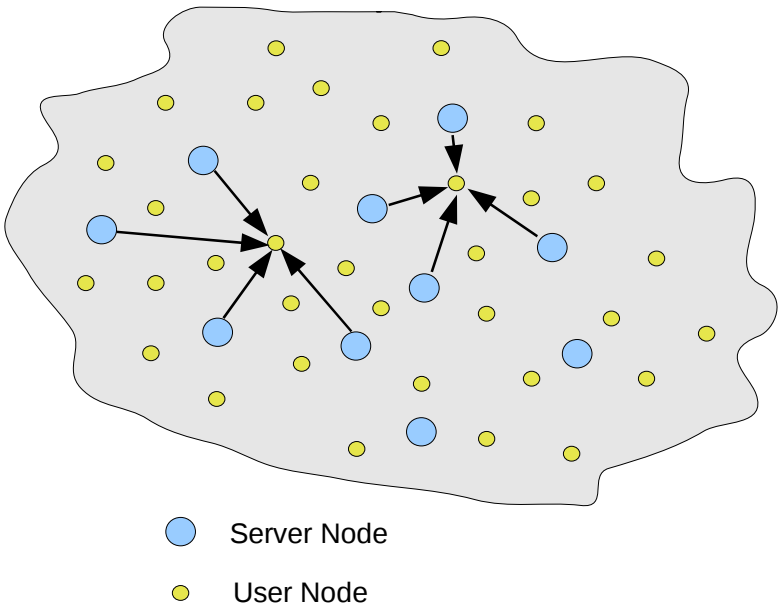
Replicated Server Option



Data replicated across ℓ nodes in the network

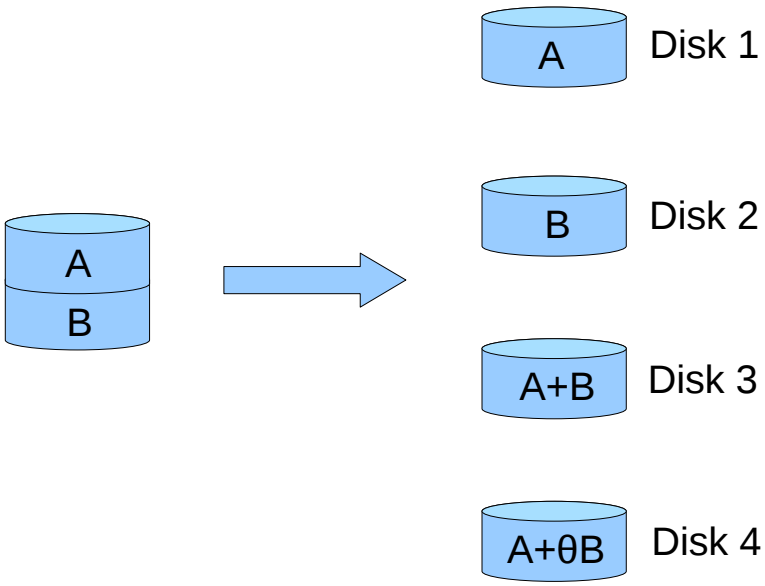
- But stores ℓB units of data
- Links surrounding each server may still be congested
- Can tolerate $\ell - 1$ node failures

Erasure Code Option



- Data split into k fragments and an $[n,\ k]$ MDS code used to generate n fragments which are stored in n nodes in the network
- each fragment represents a single symbol from the finite field $\mathbb{F}_q$
- Stores $\frac{n}{k}B$ units of data
- Links are less likely now to be congested as data is dispersed
- Can tolerate $n-k$ node failures – greater probability of network survival

RAID Example



(4, 2) MDS code

Used in RAID 6

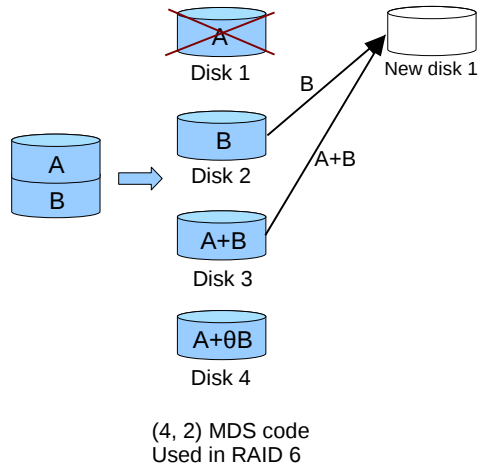
- each node stores one symbol
- then data stored across the network looks like a codeword dispersed in space

Outline

But How Does One Handle Node Failure ?

The obvious option:

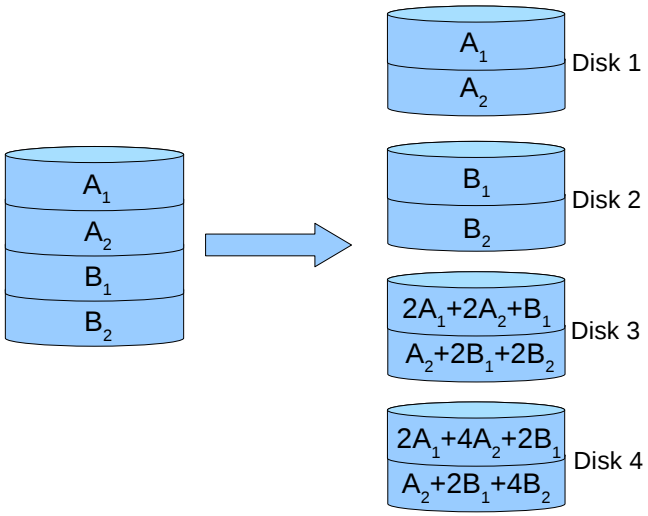
- Connect to any k nodes,
- reconstruct entire data file,
- then reconstruct data stored in the node



But downloading B units of data to revive a node that stores B/k units of data is wasteful!

Is there a better option?

Yes! Introducing Regenerating Codes



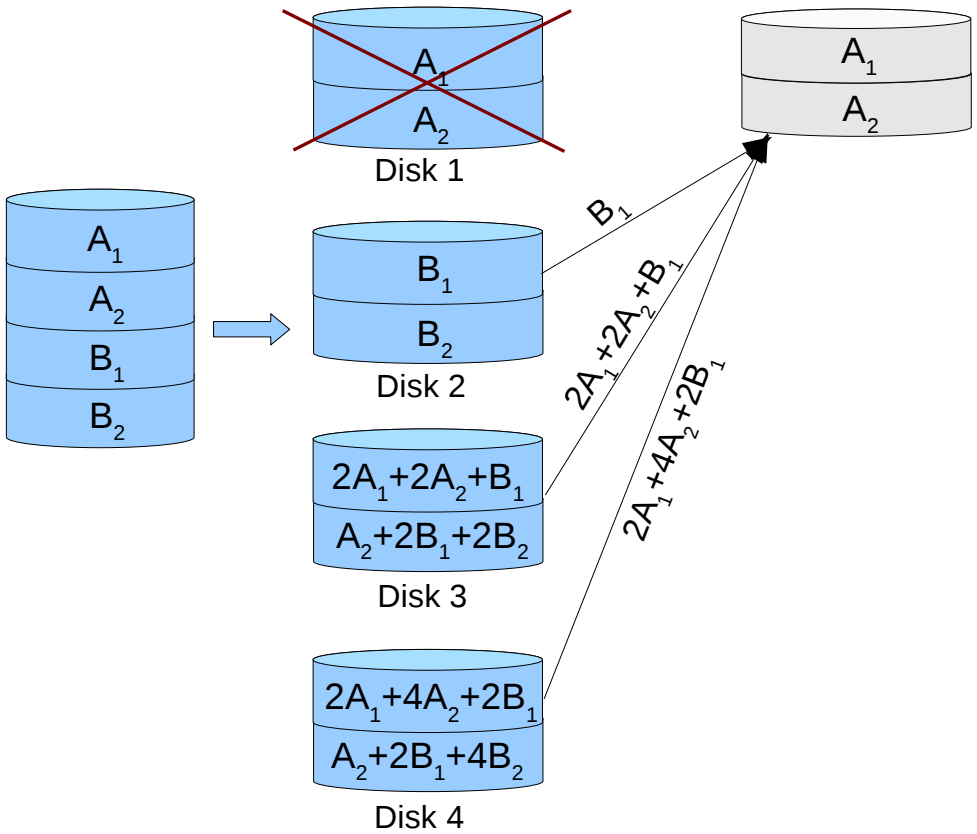
Store vectors as opposed to scalars at each node and use *vector MDS* codes. New alphabet:

$$\mathcal{A} = \mathbb{F}_q^{\alpha}, \quad \alpha > 1.$$

(in the figure, $\alpha = 2$ and $q = 5$)

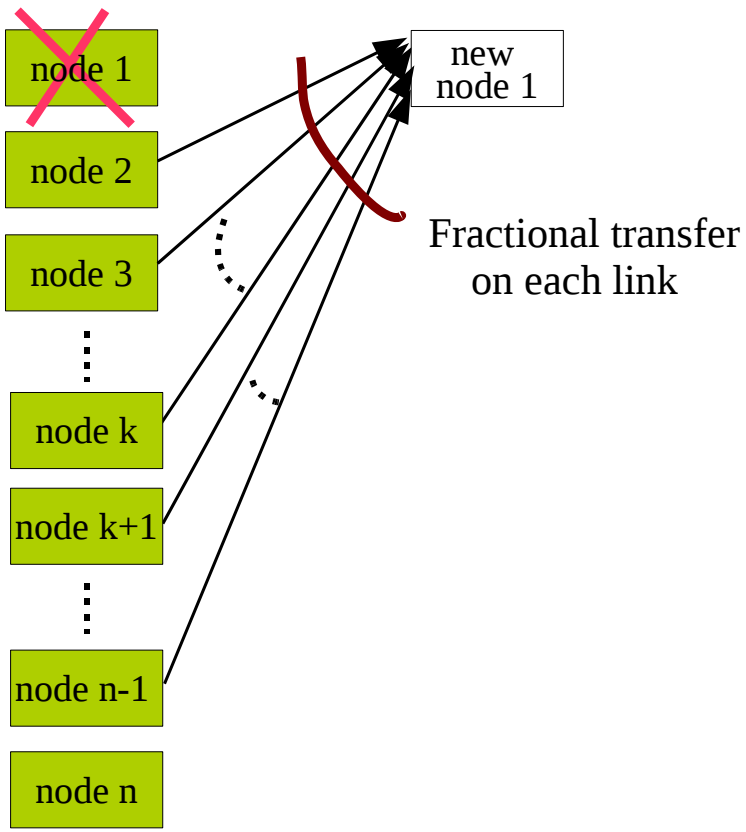
How Does a Vector Alphabet Help?

We illustrate with an example:



(download 3 half-symbols as opposed to 2 full-symbols)

Fractional Information Transfer

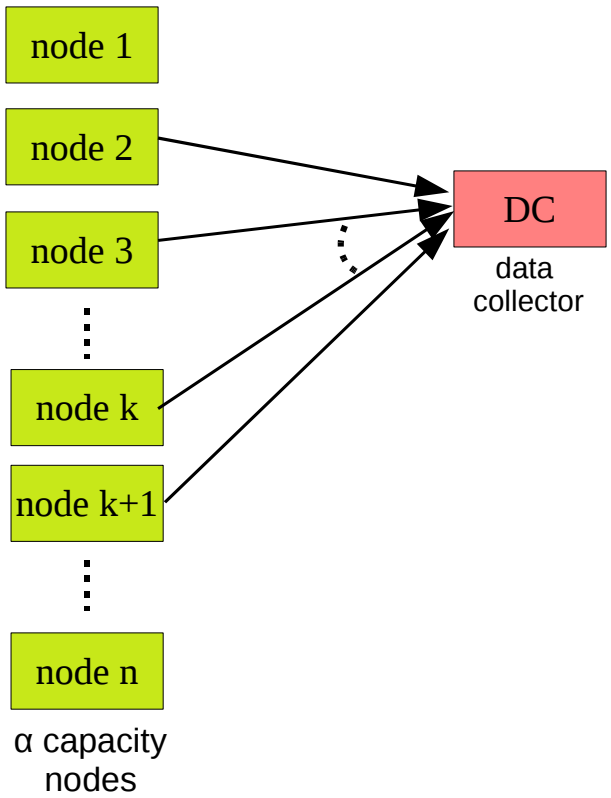


the key point is that when each node stores a vector with α components, it is possible to transfer a *fraction* of the information stored in a node to aid in node repair.

The potential savings can be quantified using principles of network coding

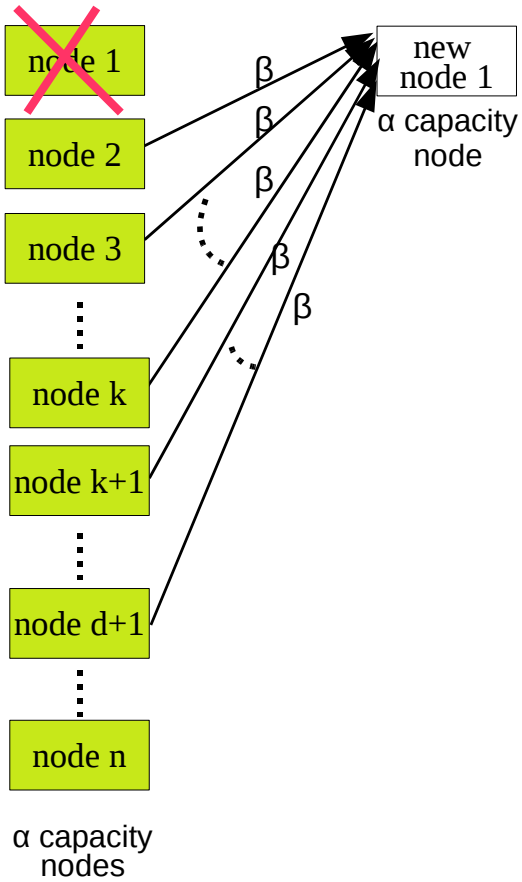
Recap On Notation

$\{\mathbb{F}_q,\ [n,k],\ [\alpha, B]\}$



- n = number of nodes in the network
- k= number of nodes to connect to for reconstructing file $\mathcal{B}$
- α = number of symbols stored per node
- B is the file size

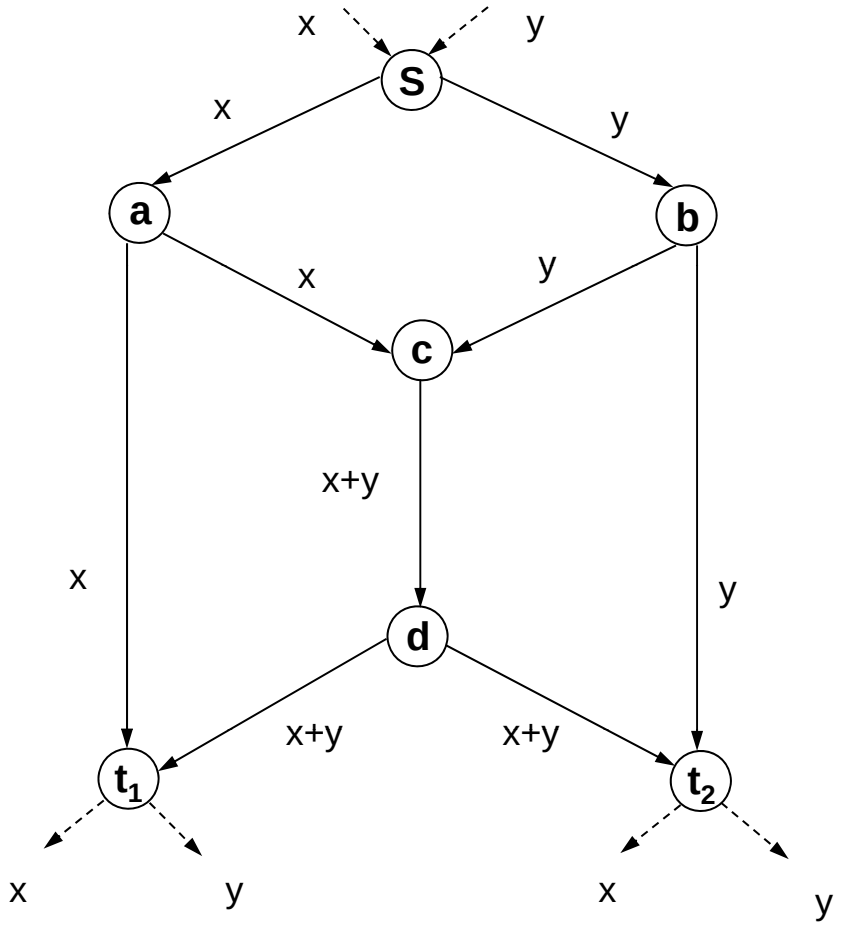
Two Additional Parameters: $(d,\ \beta)$



- d = number of nodes to connect to for regeneration of a failed node $(k \leq d \leq n-1)$
- β = number of symbols downloaded from each node for regeneration $(\beta < \alpha)$
- the quantity $d\beta$ is termed the "repair bandwidth"

Extended Parameter Set: $\{\mathbb{F}_q,\ [n,k,d],\ [\alpha,\beta,B]\}$

Network Coding Example

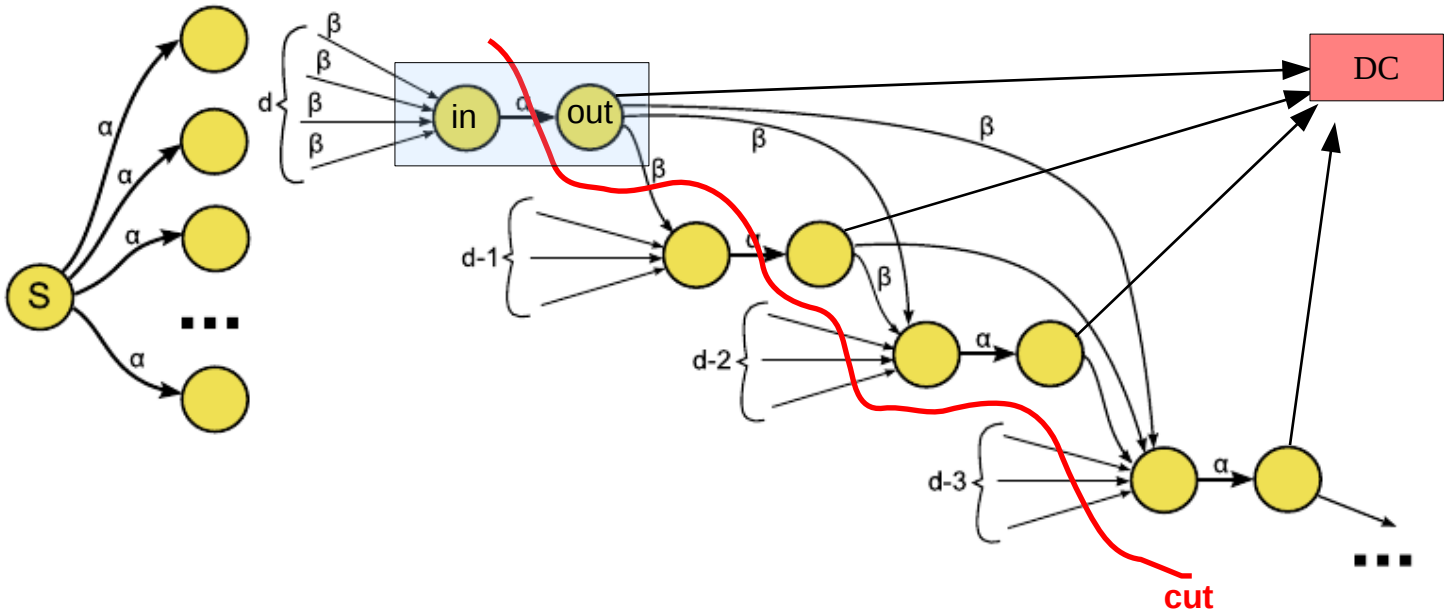


- both sinks want all the data – an example of a "multicast"
- min-cut capacity to each sink $= 2$
- MFMC theorem of commodity flow tells us each sink can get the information
- network coding tells us that in this situation, both sinks can recover the information!
- an example solution is illustrated

Cut-Set Bound for Regenerating Codes

The network here is the original set of n nodes plus all subsequent replacement nodes. Turns out, the min-cut capacity is given by:

$$\sum_{i=0}^{k-1} \min\{\alpha, (d-i)\beta\} \quad \geq \quad B$$



The Minimum-Storage Regeneration (MSR) Point (minimize α, then β)

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

$$B = \sum_{i=0}^{k-1} \alpha, \qquad (d-i)\beta \geq \alpha, \quad \text{all } i$$

$$\text{i.e., } \alpha = \frac{B}{k} \qquad \beta = \frac{\alpha}{d-k+1}$$

$$\text{or, } \alpha = \beta(d-k+1), \qquad B = \alpha k$$

(Note: both α, B are multiples of β)

The Minimum-Bandwidth Regeneration (MBR) Point (minimize β, then α)

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

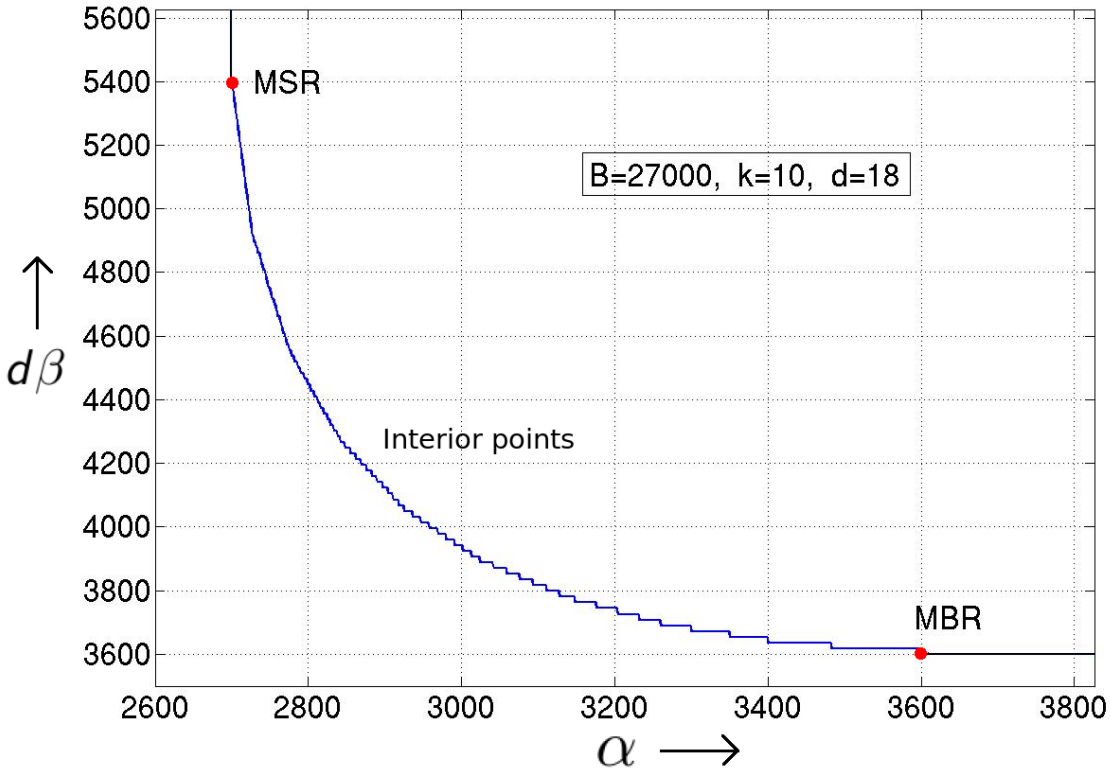
$$B = \sum_{i=0}^{k-1} (d-i)\beta, \qquad \alpha \geq (d-i)\beta, \quad \text{all } i$$

$$\text{i.e., } B = [dk - \binom{k}{2}]\beta \qquad \alpha = d\beta$$

(Note: again, both α, B are multiples of β)

(other operating points are possible; in general, there is a tradeoff)

The Storage-Repair Bandwidth Tradeoff Curve



- Repair bandwidth $d\beta$ versus storage α
- MSR and MBR are the two extreme points

Summarizing MSR and MBR Parameters

(storage per node, repair bandwidth)

MSR Point

$$\left(\alpha_{\text{min}} = \left(\frac{B}{k}\right), \quad d\beta_{\text{min}} = \left(\frac{B}{k}\right)\left(\frac{1}{1-(\frac{k-1}{d})}\right)\right)$$

(store least possible; download a little more)

MBR Point

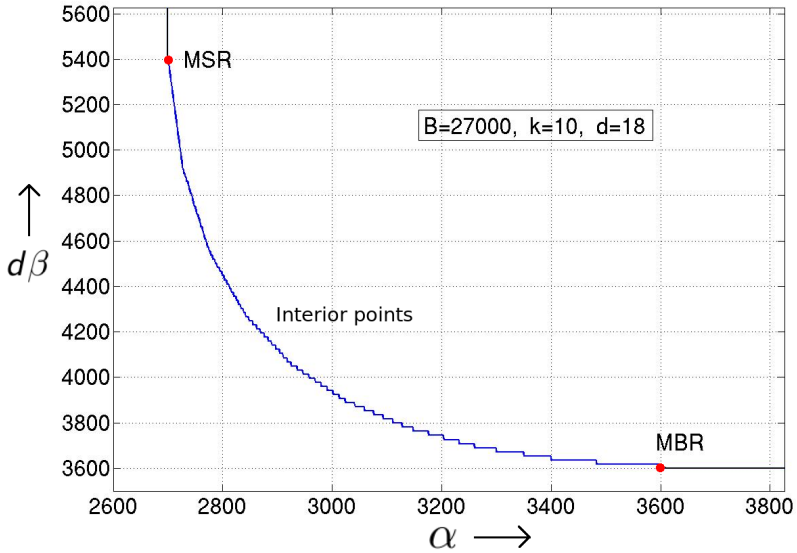
$$\left(\alpha_{\text{min}} = \left(\frac{B}{k}\right)\left(\frac{1}{1-\frac{(k-1)}{2d}}\right), \quad d\beta_{\text{min}} = \left(\frac{B}{k}\right)\left(\frac{1}{1-\frac{(k-1)}{2d}}\right)\right)$$

(store a little more; download only what you store)

$\beta = 1$ and "Striping"

- at both MSR, MBR endpoints, α, B are multiples of β
- β dictates smallest size of file for which a solution exists
- a solution for $\beta = 1$ can be replicated to give solutions for $\beta > 1$ ("striping" of data)
- thus constructions for $\beta = 1$ are of greatest interest

Exact versus Functional Regeneration



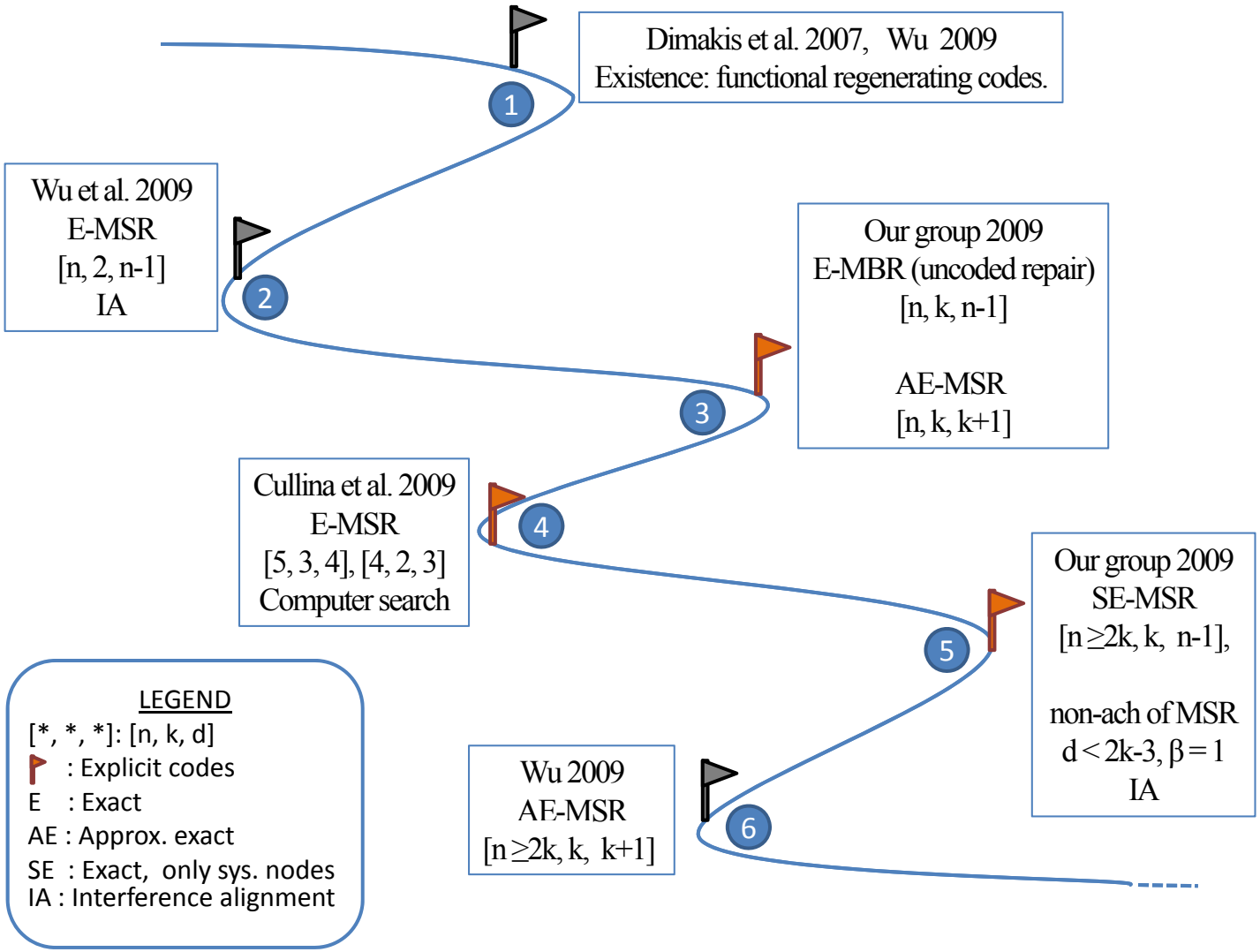
-

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

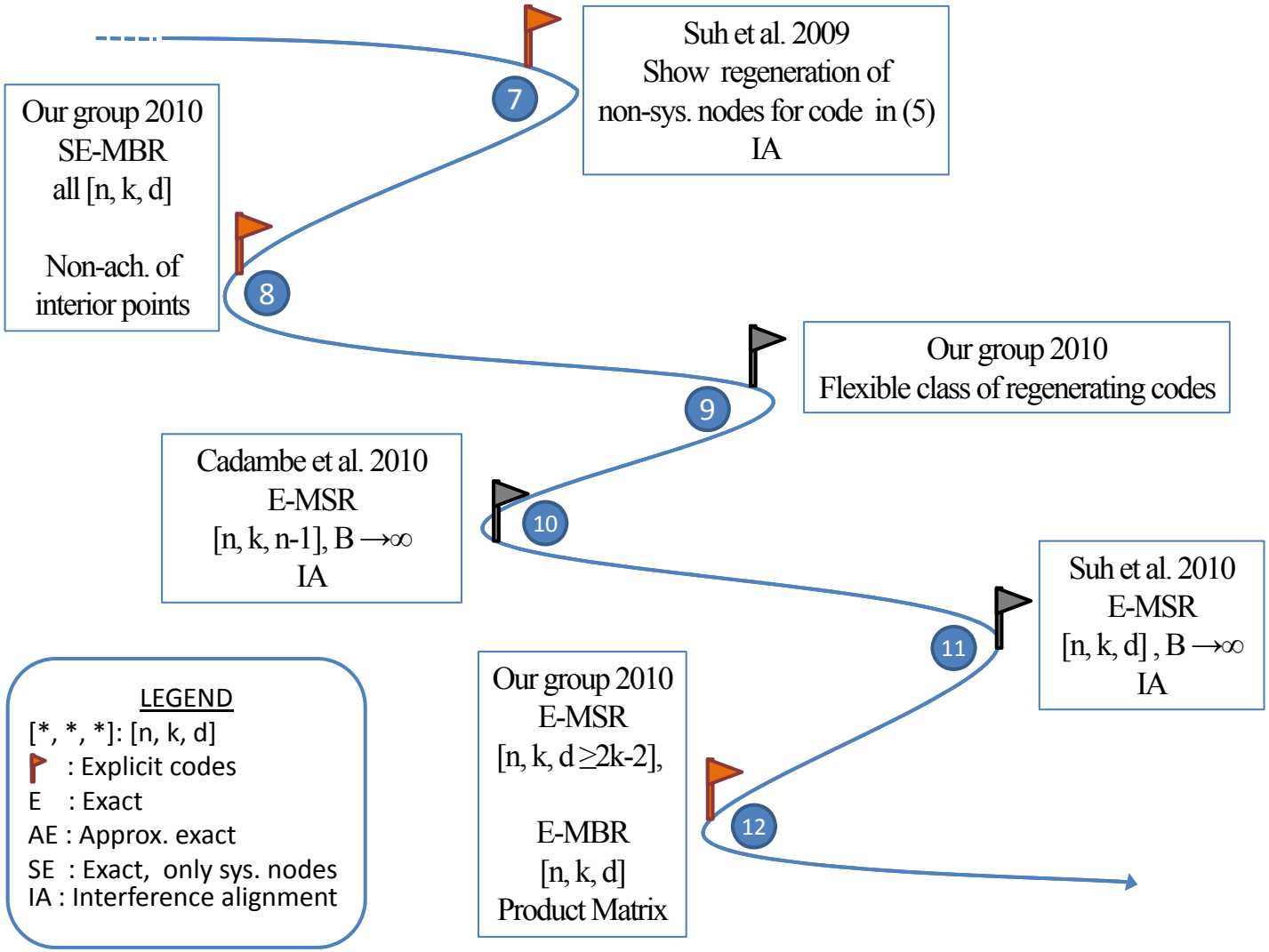
- Bound is known to be achievable for functional regeneration
- What if one demanded *exact* regeneration ?

Outline

Brief History



Brief History (2)



Our Contributions

- Constructions
  - E-MBR code $d = n - 1$
  - AE-MSR code $d = k + 1$
  - SE-MSR code, $d = n - 1$ exact regeneration of systematic nodes
  - Most recently, a unified product-matrix construction for MSR, MBR, all d (essentially)
- Non-achievability of interior points under exact regeneration
- non-existence of MSR code with $d < 2k - 3$ under exact regeneration when $\beta = 1$
- the cut-set bound, existence of codes and a preliminary construction for a more flexible regeneration code set up

Outline

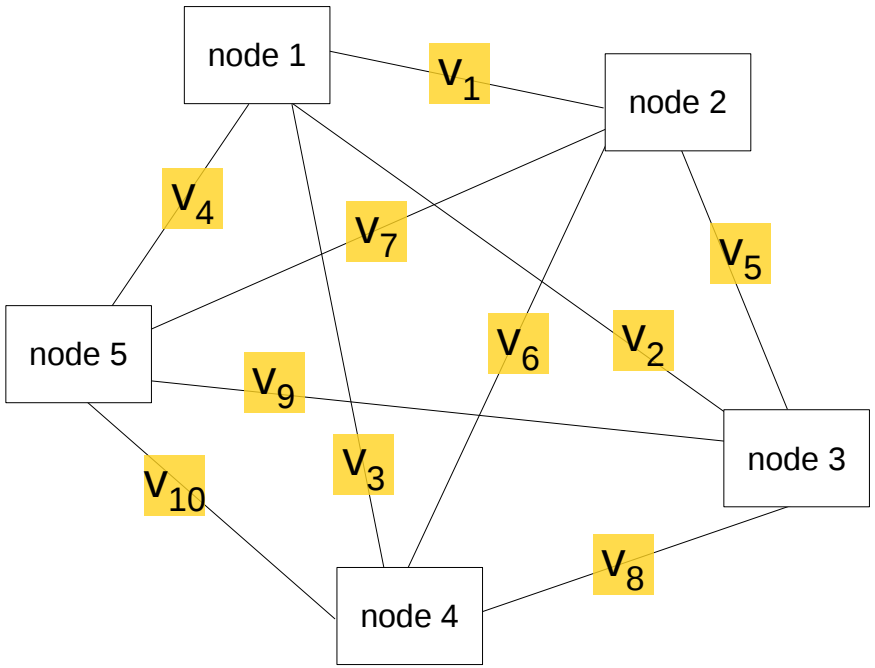
An Example Construction for the MBR $d = n - 1$ Point

- $n = 5,\ k = 3,\ d = 4$
- $B = 9,\ \alpha = 4,\ \beta = 1$

node 1	v_1	v_2	v_3	v_4
node 2	v_1	v_5	v_6	v_7
node 3	v_2	v_5	v_8	v_9
node 4	v_3	v_6	v_8	v_{10}
node 5	v_4	v_7	v_9	v_{10}

Gen Mx $[10, 9]$ MDS Code:

$$\underline{m}^t [\underline{v}_1\ \underline{v}_2\ \cdots\ \underline{v}_{10}]$$



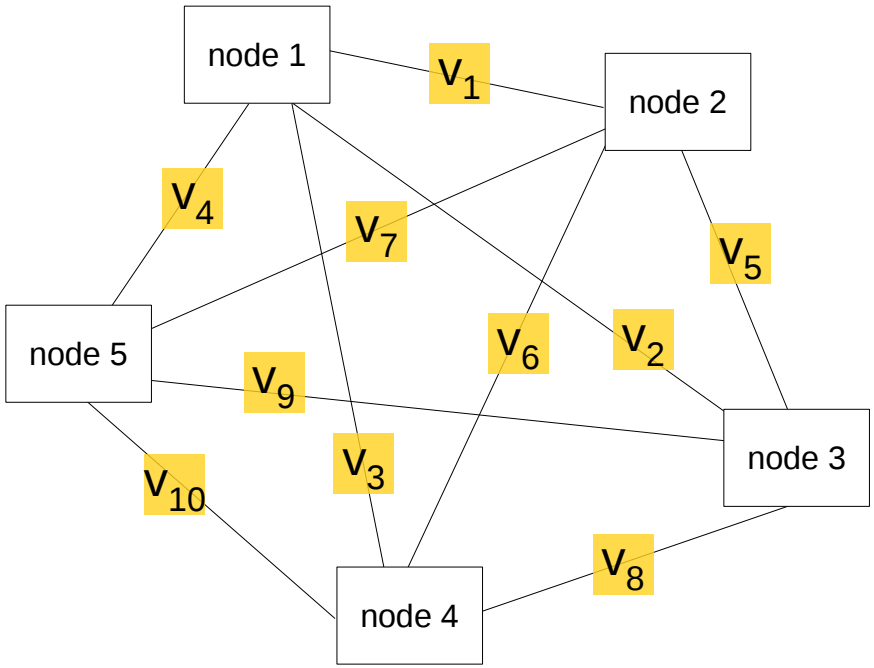
K. V. Rashmi, N. B. Shah, P. V. Kumar, and K. Ramchandran, "Explicit Construction of Optimal Exact Regenerating Codes for Distributed Storage," in *Proc. Allerton Conference on Control, Computing and Communication*, Urbana-Champaign, Sep. 2009.

The General Case: MBR $d = n - 1$ Code

For the general case, the MDS code would have

$$\text{length} = \binom{n}{2}$$

$$\text{dimension} = dk - \binom{k}{2}$$



MISER Code

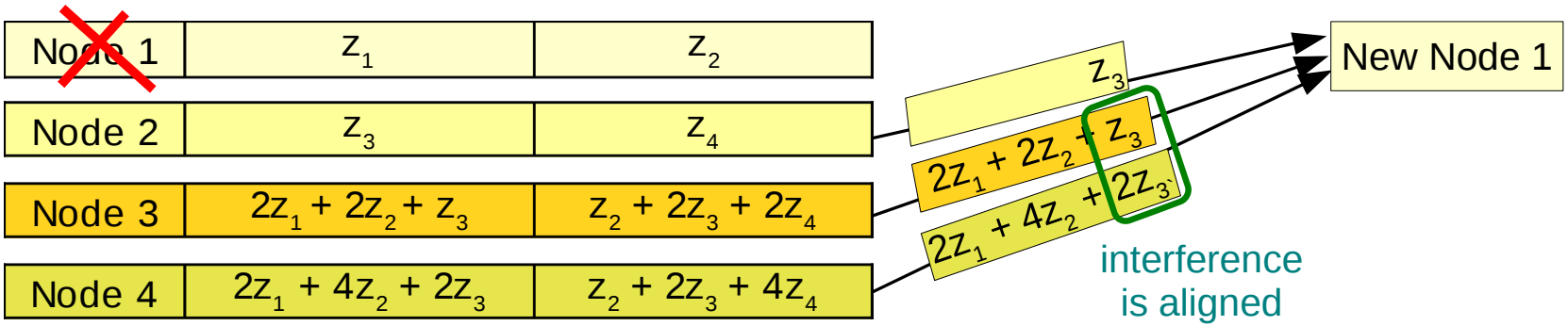
- Family of MSR codes for $d = n - 1 \geq 2k - 1$ that perform
  - reconstruction
  - optimal exact-repair of systematic nodes
- Based on the concept of Interference Alignment
- Suh-Ramchandran show that this code can perform optimal exact-repair of parity nodes as well

N. B. Shah, K. V. Rashmi, P. V. Kumar, and K. Ramchandran, "Explicit codes minimizing repair bandwidth for distributed storage," in *Proc. ITW,* Cairo, Jan. 2010.

N. B. Shah, K. V. Rashmi, P. V. Kumar, and K. Ramchandran, "Interference Alignment in Regenerating Codes for Distributed Storage: Necessity and Code Constructions," submitted to *IEEE Trans. on Information Theory*, available online at `arXiv:1005.1634v2 [cs.IT]`.

MISER Code: Toy Example

- $n = 4,\ k = 2,\ d = 3$
- $\alpha = d - k + 1 = 2,\ B = k\alpha = 4$
- Field of operation: $\mathbb{F}_5$



Product-Matrix Framework

$$\underbrace{C}_{n\times\alpha} = \underbrace{\Psi}_{n\times d}\underbrace{M}_{d\times\alpha}$$

- M : Message matrix
  - Contains message symbols with some message symbols repeated
  - Possesses a block-symmetry property
- Ψ : Encoding matrix
  - Used to disperse information across the nodes
  - Independent of message symbols
- C : Code matrix
  - Each row represents one node
  - i^{th} node stores: $\underline{\psi}_i^t M$

K. V. Rashmi, N. B. Shah, and P. V. Kumar, "Optimal Exact-Regenerating Codes for the MSR and MBR Points via a Product-Matrix Construction," submitted to *IEEE Transactions on Information Theory*, available online at `arxiv:1005.4178 [cs.IT]`.

The Product-Matrix MBR (PM-MBR) Code

- $\alpha = d$
- $B = kd - \binom{k}{2} \quad \rightarrow \quad B = \binom{k+1}{2} + k(d-k)$

- Let S be a $(k \times k)$ symmetric matrix with $\binom{k+1}{2}$ distinct message symbols
- Let T be a $(k \times (d-k))$ matrix with $k(d-k)$ distinct message symbols
- thus all message symbols are accounted for

Product-matrix MBR Code

- Message matrix $\underbrace{M}_{d\times d} = \begin{bmatrix} \underbrace{S}_{k\times k} & \underbrace{T}_{k\times(d-k)} \\ \underbrace{T^t}_{(d-k)\times k} & \underbrace{0}_{k\times(d-k)} \end{bmatrix}$ (symmetric)

- Encoding matrix $\underbrace{\Psi}_{n\times d} = \begin{bmatrix} \underbrace{\Phi}_{n\times k} & \underbrace{\Delta}_{n\times(d-k)} \end{bmatrix}$

  Φ: any k rows linearly independent
  Ψ: any d rows linearly independent

  e.g., Cauchy, Vandermonde matrix

Product-matrix MBR Code : Data Reconstruction

Node i passes: $\underline{\psi}_i^t M$

Aggregator $\downarrow$

$$\Psi_{DC} M$$

$(\Psi_{DC} = [\Phi_{DC} \quad \Delta_{DC}]$ is $(k \times d))$

$\downarrow$

Decoder

$$\begin{bmatrix} \Phi_{DC} S + \Delta_{DC} T^t & \Phi_{DC} T \end{bmatrix}$$

$\downarrow$

Φ_{DC} is $k \times k$, invertible

Decode T

$\downarrow$

Subtract $\Delta_{DC} T^t$, Decode S

$$M = \begin{bmatrix} S & T \\ T^t & 0 \end{bmatrix}$$

$$\Psi = \begin{bmatrix} \Phi & \Delta \end{bmatrix}$$

$$C = \Psi M$$

Product-matrix MBR Code : Exact Regeneration

Replacement node f needs: $\underline{\psi}_f^t M$

Helper node i, $1 \leq i \leq d$ stores: $\underline{\psi}_i^t M$

Helper node i passes: $\underline{\psi}_i^t M \underline{\psi}_f$

Aggregator $\downarrow$

$\Psi_{\text{repair}} M \underline{\psi}_f$

(Ψ_{repair} is $d \times d$, invertible)

Partial Decoder $\downarrow$

$M \underline{\psi}_f$

(M is symmetric)

Re-encoder $\downarrow$

$\underline{\psi}_f^t M$

$$M = \begin{bmatrix} S & T \\ T^t & 0 \end{bmatrix}$$

$$\Psi = \begin{bmatrix} \Phi & \Delta \end{bmatrix}$$

$$C = \Psi M$$

The Product-matrix MSR Code Parameters

- Here again $\beta = 1$
- $\alpha = d - k + 1$

The MSR point-Numerology

- $d < 2k - 3$ not possible with $\beta = 1$
- This code is designed for $d \geq 2k - 2$
- Choose $d = 2k - 2$ first, then extend to higher d

- Gives

$$\begin{aligned} k &= \alpha + 1 \\ d &= 2\alpha \\ B &= \alpha(\alpha + 1) \end{aligned}$$

- S_1, S_2: $(\alpha \times \alpha)$ symmetric matrices with $\frac{\alpha(\alpha+1)}{2}$ distinct message symbols each

The Product-Matrix MSR Code

- Message matrix $\underbrace{M}_{d\times\alpha} = \begin{bmatrix} \underbrace{S_1}_{\alpha\times\alpha} \\ \\ \underbrace{S_2}_{\alpha\times\alpha} \end{bmatrix}$

- Encoding matrix $\underbrace{\Psi}_{n\times d} = \begin{bmatrix} \underbrace{\Phi}_{n\times\alpha} & \underbrace{\Lambda\Phi}_{n\times\alpha} \end{bmatrix}$

Φ: any α rows linearly independent
Λ: $n \times n$ diagonal matrix with the diagonal elements distinct
Ψ: any d rows linearly independent
e.g., Vandermonde

The Product-Matrix MSR Code-Data Reconstruction

Node i passes: $\underline{\psi}_i^t M$

Aggregator $\downarrow$

$\Psi_{DC} M$

($\Psi_{DC} = [\Phi_{DC} \quad \Lambda_{DC}\Phi_{DC}]$ is $k \times d$)

$\downarrow$

$[\Phi_{DC} S_1 + \Lambda_{DC}\Phi_{DC} S_2]$

Decoder $\downarrow$

$[\Phi_{DC} S_1 \Phi_{DC}^t + \Lambda_{DC}\Phi_{DC} S_2 \Phi_{DC}^t]$

$\downarrow$

$[P + \Lambda_{DC} Q]$

(P and Q symmetric)

$\downarrow$

$(i,\ j) : P_{ij} + \lambda_i Q_{ij}, \quad (j,\ i) : P_{ij} + \lambda j Q_{ij}$

(Solve for P and Q)

$\downarrow$

Recover S_1 and S_2

$$M = \begin{bmatrix} S_1 \\ S_2 \end{bmatrix}$$

$$\Psi = \begin{bmatrix} \Phi & \Lambda\Phi \end{bmatrix}$$

$$C = \Psi M$$

The Product-Matrix MSR Code-Exact Regeneration

Replacement node f needs: $\underline{\psi}_f^t M$

Helper node i stores: $\underline{\psi}_i^t M$

Helper node i passes: $\underline{\psi}_i^t M \underline{\phi}_f$

Aggregator $\downarrow$

$$\Psi_{\text{rep}} M \underline{\phi}_f$$

(Ψ_{rep} is $d \times d$, invertible)

Partial Decoder $\downarrow$

$$M\underline{\phi}_f = \begin{bmatrix} S_1\underline{\phi}_f \\ S_2\underline{\phi}_f \end{bmatrix}$$

Re-encoder $\downarrow$

$$\underline{\phi}_f^t S_1 + \lambda_f \underline{\phi}_f^t S_2 \;=\; \underline{\psi}_f^t M$$

$$M = \begin{bmatrix} S_1 \\ S_2 \end{bmatrix}$$

$$\Psi = \begin{bmatrix} \Phi & \Lambda\Phi \end{bmatrix}$$

$$C = \Psi M$$

References

- A. G. Dimakis, P. B. Godfrey, Y. Wu, M. Wainwright, and K. Ramchandran, "Network Coding for Distributed Storage Systems," to appear in *IEEE Transactions on Information Theory*, available online at `arXiv:0803.0632 [cs.IT]`.
- Y. Wu and A. Dimakis, "Reducing Repair Traffic for Erasure Coding-Based Storage via Interference Alignment," in *Proc. IEEE International Symposium on Information Theory,* Jul. 2009.
- Y. Wu, "Existence and Construction of Capacity-Achieving Network Codes for Distributed Storage," in *Proc. IEEE International Symposium on Information Theory,* Seoul, Jul. 2009.
- K. V. Rashmi, N. B. Shah, P. V. Kumar, and K. Ramchandran, "Explicit Construction of Optimal Exact Regenerating Codes for Distributed Storage," in *Proc. Allerton Conference on Control, Computing and Communication*, Urbana-Champaign, Sep. 2009.
- D. Cullina, A. G. Dimakis and Tracey Ho, "Searching for Minimum Storage Regenerating Codes," in *Proc. Allerton Conference on Control, Computing and Communication,* Urbana-Champaign, September 2009.
- N. B. Shah, K. V. Rashmi, P. V. Kumar, and K. Ramchandran,"Explicit Codes Minimizing Repair Bandwidth for Distributed Storage," in *Proc. IEEE Information Theory Workshop,* Cairo, Jan. 2010.
- Y. Wu, "A Construction of Systematic MDS Codes with Minimum Repair Bandwidth," submitted to *IEEE Transactions on Information Theory*, available online at `arXiv:0910.2486v1 [cs.IT]`.
- K. V. Rashmi, N. B. Shah and P. V. Kumar, "Optimal Exact-Regenerating Codes for the MSR and MBR Points via a Product-Matrix Construction," submitted to *IEEE Transactions on Information Theory*, available online at `arxiv:1005.4178 [cs.IT]`.
- N. B. Shah, K. V. Rashmi, and P. V. Kumar "A Flexible Class of Regenerating Codes for Distributed Storage," in *Proc. IEEE International Symposium on Information Theory,* Austin, Jun. 2010.
- K. V. Rashmi, N. B. Shah, P. V. Kumar, and K. Ramchandran "Explicit and Optimal Exact-Regenerating Codes for the Minimum-Bandwidth Point in Distributed Storage," in *Proc. IEEE International Symposium on Information Theory,* Austin, Jun. 2010.

References

- C. Suh and K. Ramchandran, "Interference Alignment Based Exact Regeneration Codes for Distributed Storage," in *Proc. IEEE International Symposium on Information Theory,* Austin, Jun. 2010.
- S. Pawar, S. El Rouayheb and K. Ramchandran, "On Secure Distributed Data Storage Under Repair Dynamics," in *Proc. IEEE International Symposium on Information Theory,* Austin, Jun. 2010.
- V. R. Cadambe, S. A. Jafar, and H. Maleki, "Distributed Data Storage with Minimum Storage Regenerating Codes - Exact and Functional Repair are Asymptotically Equally Efficient," available online at `arXiv:1004.4299v1 [cs.IT]`.
- C. Suh and K. Ramchandran, "On the Existence of Optimal Exact-Repair MDS Codes for Distributed Storage," available online at `arXiv:1004.4663v1 [cs.IT]`.
- Dennis S. Bernstein, *Matrix mathematics: Theory, facts, and formulas with application to linear systems theory*, Princeton University Press, Princeton, NJ, p.119, 2005.
- S. Rhea, P. Eaton, D. Geels, H. Weatherspoon, B. Zhao, and J. Kubiatowicz, "Pond:the OceanStore prototype," in *Proc. USENIXFile and Storage Technologies (FAST)*, 2003.
- R. Bhagwan, K. Tati, Yu Chung Cheng, S. Savage, and G. M. Voelker, "Total recall: System support for automated availability management," in *NSDI*, 2004.
- R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking,* v.11 n.5, p.782-795, Oct. 2003.

Thanks!