Fault Tolerance and Attack Resilience on Big Data Storage

Yunghsiang S. Han (韓永祥)

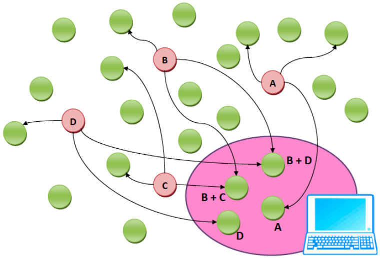
National Taiwan University of Science and Technology, Taiwan

June 25, 2014

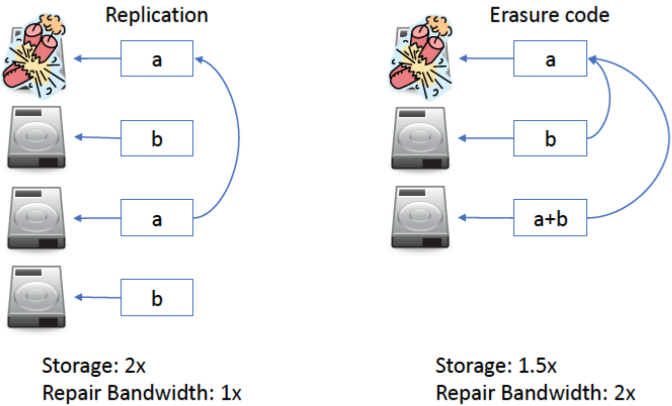
Outline

Distributed Networked Storage

- To store data reliably by adding redundancy
- k data are distributed to $n > k$ storage nodes to improve reliability
- Two common methods: Replication (Google), Erasure coding (Oceanstore, TotalRecall)
- In erasure coding, access any k storage nodes to recover data



Replication VS. Erasure Coding



How to reduce the repair bandwidth: **regenerating codes** is the solution

Real Systems using Erasure Codes

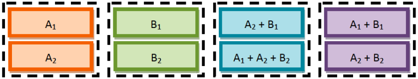
- All for big data storage
- Windows Azure, (Cheng et al. USENIX 2012) (Local Reconstruction Codes (LRC)) (Microsoft)
- CORE (Li et al. MSST 2013) (Exact-repair MSR (EMSR) Code)
- NCCloud (Hu et al. USENIX FAST 2012) (Functional MSR Code)
- ClusterDFS (Pamies Juarez et al.) (Self-Repairing Codes)
- StorageCore (Esmaili et al.) (over Hadoop HDFS) HDFS Xorbas (Sathiamoorthy et al. VLDB 2013) (over Hadoop HDFS) (LRC code) (Facebook)

Some Metrics of Interest for Erasure Codes

- Storage Space: Quantity of data stored in the system
- Repair Bandwidth: Amount of data transmitted in the network to repair a failed node (disk)
- Locality: The number of nodes accessed to repair a failed node
- Fault-Tolerance: How many nodes can be failed
- Attack Resilience: How many compromised nodes the system can tolerate

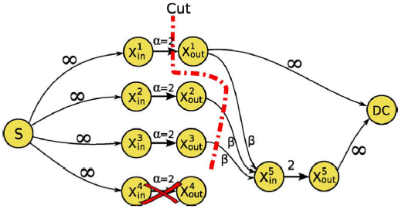
A (4, 2) binary regenerting code

- The original data blocks: $A_1, \ A_2, \ B_1, \ B_2$
- The total size of original data blocks: $B = 4$
- Each storage node stores two blocks ($\alpha = 2$)
- $n = 4$ and $k = 2$
- Access any 2 storage nodes to recover data
- How to repair failed storage node?



Information Flow for Repair: (4,2) Code

- Access more than $k = 2$ nodes to save repair bandwidth
- X^5 is connected to $d = 3$ active storage nodes
- β (data) blocks communicated from each active storage node
- The min-cut $\geq B = 4$ blocks
- The min-cut value: $\alpha + 2\beta$
- The total repair bandwidth: $\gamma = d\beta = 3$ blocks



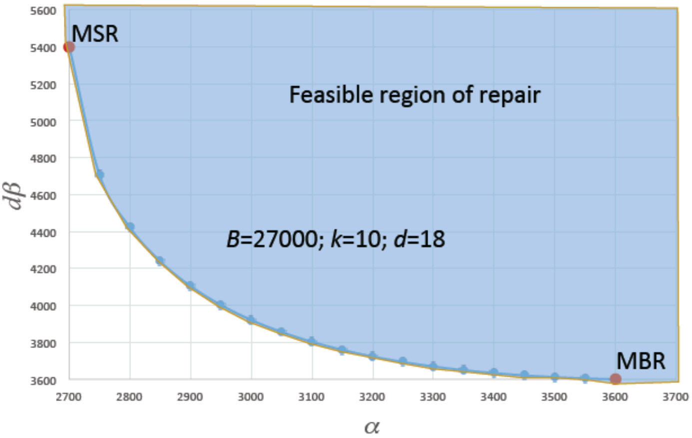
$[n, k, d]$ Regenerating Codes

- To reconstruct the original data symbols and regenerate coded data
- k and d surviving nodes to ensure successful data reconstruction and regeneration $(k \leq d \leq n - 1)$

Cut-Set Bound on the Repair Bandwidth

- $$B \leq \sum_{i=0}^{k-1} \min\{\alpha, (d-i)\beta\} \quad (1)$$
- Minimizing α (β) with minimum storage (repair bandwidth) requirement
- The two extreme points in (1): the minimum storage regeneration (MSR) and minimum bandwidth regeneration (MBR) points

An Example of Storage-Bandwidth Tradeoff Curve



Minimum Storage Regeneration (MSR)

- First minimizing α and then minimizing β
-

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

- Set $\beta = 1$

$$\begin{aligned} \alpha &= d-k+1 \\ B &= k(d-k+1) = k\alpha \end{aligned} \tag{3}$$

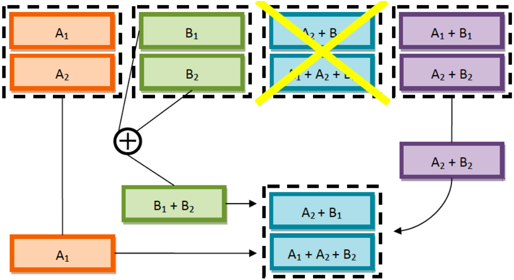
Minimum Bandwidth Regeneration (MBR)

- First minimizing β and then minimizing α
- $$\begin{aligned}\beta &= \frac{2B}{k(2d-k+1)} \\ \alpha &= \frac{2dB}{k(2d-k+1)}\end{aligned} \quad (4)$$
- Set $\beta = 1$

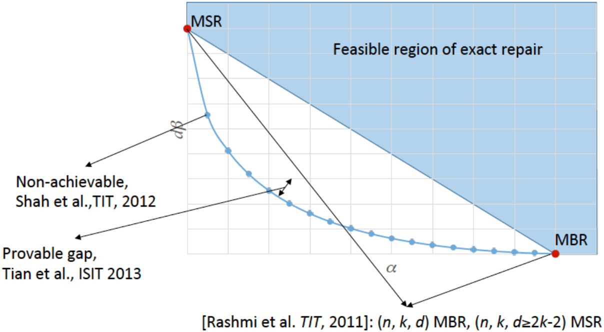
$$\begin{aligned}\alpha &= d \\ B &= kd - k(k-1)/2\end{aligned} \quad (5)$$

Exact Repair

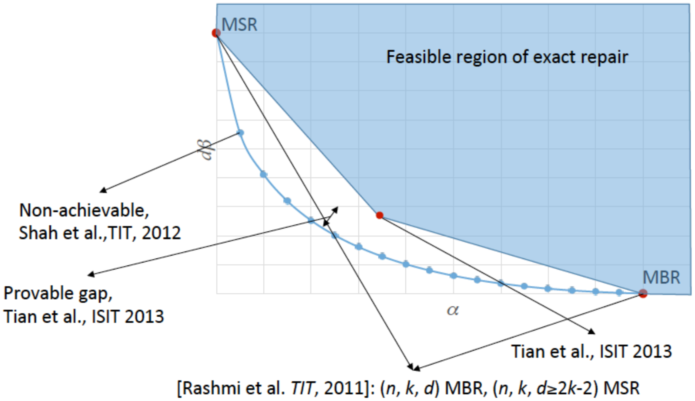
- Storage nodes to compute functions of their stored data before communicating



Feasible Region for Exact-Repair



Open Problem for Exact-Repair



Open Problem:

- Exact-repair codes between MSR and MBR points
- Exact-MSR codes for $d = 2k - 3$

Security on Regenerating Codes

- Fault Tolerance
- Passive Eavesdropper
- Active Adversary

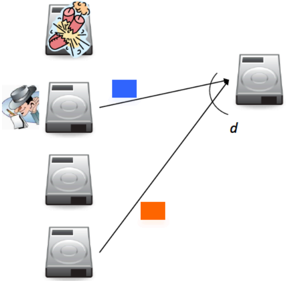
Related Existing Schemes

- Intruder model and its capacity bounds were given in (Pawar, et al, 2011)
- The exact-regenerating codes via product-matrix construction (Rashmi, et al, 2011)– fault-tolerance (erasures) and passive eavesdropper (Kumar, et al, 2014)
- The error-correcting exact-regenerating codes based on Reed-Solomon (RS) codes (Han, et al, INFOCOM2012)– active adversary
  - Efficient decoding for MBR codes (both data reconstruction and regeneration)
  - Efficient decoding for MSR codes (regeneration)
  - Decoding based on well-known RS code decoder
- The error-correcting exact-regenerating codes (Rashmi, et al, ISIT2012)– active adversary
  - Decoding capability of MBR and MSR codes are given
  - Decoding capability of MSR codes (for data reconstruction) is stated without efficient decoding scheme

Passive Eavesdropper -Intruder Model [Pawar et al., TIT 2011]

- Passive Eavesdropper: He can read the data on the observed $\ell < k$ nodes
- Capacity upper bound:

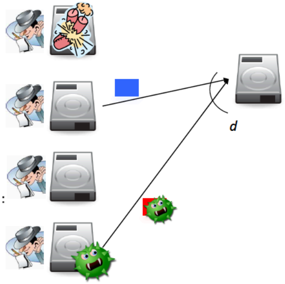
$$C_s(\alpha, d\beta) \leq \sum_{i=\ell+1}^{k} \min\{(d-i+1)\beta, \alpha\}$$



Active Omniscient Adversary -Intruder Model [Pawar et al., TIT 2011]

- Active Omniscient Adversary: He knows the file B and the data stored on all the nodes. Moreover, He can control b nodes in total, where $2b < k$
- Capacity upper bound:

$$C_s(\alpha, d\beta) \leq \sum_{i=2b+1}^{k} \min\{(d-i+1)\beta, \alpha\}$$



Attack Resilience [Rashmi et al., ISIT 2012]

- A regenerating code is (s, t)-resilient if it can correct up to s erasures and t errors during repair as well as reconstruction.
- A (s, t)-resilient regenerating code, connecting to Δ and κ nodes for repair and reconstruction respectively, must satisfy

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

where $d = \Delta - s - 2t$ and $k = \kappa - s - 2t$ [Rashmi et al., ISIT 2012]

- Practical decoding algorithms based on Reed-Solomon (RS) codes that meet the bound were provided in [Han et al., INFOCOM 2012] (except for data reconstruction for MSR codes)

Motivation

- What needs to be done in practice for MSR codes
  - Update efficient encoding based on RS codes
  - Efficient decoding for data reconstruction based on RS codes

Design Method

- Extending the code construction in [Rashmi, et al, 2011],[HAN12]
- MSR code for $d = 2k - 2$
- $\alpha = d - k + 1 = k - 1 = d/2$ and $B = k\alpha = \alpha(\alpha + 1)$
- Operating finite field: $GF(2^m)$
- The size of the data: mB bits

[Rashmi11] K. V. Rashmi, N. B. Shah, and P. V. Kumar, "Optimal exact-regenerating codes for distributed storage at the MSR and MBR points via a product- matrix construction," *IEEE Trans. Inf. Theory*, vol. 57, pp. 5227–5239, Aug. 2011.

[HAN12] Y. S. Han, R. Zheng, and W. H. Mow "Exact Regenerating Codes for Byzantine Fault Tolerance in Distributed Storage," *The IEEE INFOCOM 2012*, Orlando, March, 2012.

Encoding in [Rashmi, et al, 2011],[HAN12]

- Information sequence $\boldsymbol{m}$ arranged into an information vector $U = [A_1 A_2]$ with size $\alpha \times d$
- A_j: symmetric matrix with dimension $\alpha \times \alpha$
- Each row of U encoded into a codeword of length n by an $[n, d]$ RS code
- $U \cdot G = C$

$$G = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ a^0 & a^1 & \cdots & a^{n-1} \\ (a^0)^2 & (a^1)^2 & \cdots & (a^{n-1})^2 \\ & & \vdots & \\ (a^0)^{d-1} & (a^1)^{d-1} & \cdots & (a^{n-1})^{d-1} \end{bmatrix}$$

- a: the generator of $GF(2^m)$
- C: the codeword vector with dimension $(\alpha \times n)$
- The ith column of C distributed to storage node i

Encoding in [Rashmi, et al, 2011],[HAN12] (Cont'd)

- $$G = \begin{bmatrix} \bar{G} \\ \bar{G}\Delta \end{bmatrix}$$
- $\bar{G}$: the first α rows in G and a generator matrix for $[n, \alpha]$ RS code
- Δ: a diagonal matrix with $(a^0)^\alpha$, $(a^1)^\alpha$, $(a^2)^\alpha, \ldots,$ $(a^{n-1})^\alpha$ as diagonal elements

Update-Efficient Encoding

- Update complexity - the maximum number of encoded symbols that must be updated while a single data symbol is modified
- $G = \begin{bmatrix} \bar{G} \\ \bar{G}\Delta \end{bmatrix}$
- Least update complexity - make the number of zeros in each row of G as large as possible
- $\bar{G}$ is a generator matrix of the $[n, \alpha]$ RS code and G is a generator matrix of the $[n, d = 2\alpha]$ RS code
- The encoding scheme given in [Rashmi, et al, 2011],[HAN12] is with largest update complexity, n

Main Result 1

- $\bar{G}$ is a generator matrix of the $[n, \alpha]$ RS code with roots $a^1, a^2, \ldots, a^{n-\alpha}$
- The diagonal elements of Δ be $(a^0)^\alpha$, $(a^1)^\alpha$, …, $(a^{n-1})^\alpha$, where $m \geq \lceil \log_2 n \rceil$ and $\gcd(2^m - 1, \alpha) = 1$
- We prove that G is a generator matrix of $[n, d]$ RS code with roots $a^1, a^2, \ldots, a^{n-d}$

Main Result 2

- $\bar{G} = [D|I]$, I is identical matrix
- The update complexity is now $n - \alpha + 1$
- The largest number of zero elements in each row of G we can have
-

$$GF(2^3), n = 7, k = 4, d = 2\alpha = 6,$$

$$G = \begin{bmatrix} 5 & 7 & 7 & 4 & 1 & 0 & 0 \\ 2 & 4 & 6 & 1 & 0 & 1 & 0 \\ 5 & 5 & 3 & 2 & 0 & 0 & 1 \\ 7 & 1 & 3 & 3 & 6 & 0 & 0 \\ 1 & 6 & 4 & 2 & 0 & 1 & 0 \\ 7 & 2 & 2 & 4 & 0 & 0 & 3 \end{bmatrix}$$

Decoding with No Error

- Proposed in [Rashmi, et al, 2011]
- Received $Y_{\alpha\times k} = [Z_1 \bar{G}_k + Z_2 \bar{G}_k \Delta]$
- Instead of solving Z_1, Z_2, solve P and Q in $\bar{G}_k^T Y_{\alpha\times k} = P + Q\Delta$

Decoding with Multiple Errors

- Observing $P(Q)$ can be decoded based on $[n, \alpha = k-1]$ Reed-Solomon code generated by $\bar{G}$ that reaches decoding capability $\lfloor \frac{n-k+1}{2} \rfloor$
- It is not straightforward for decoding with multiple errors
- While calculating $P(Q)$, errors propagate from columns to corresponding rows.
-

$$GF(2^3), n = 7, k = 4, d = 2\alpha = 6$$

$$G = \begin{bmatrix} 5 & 7 & 7 & 4 & 1 & 0 & 0 \\ 2 & 4 & 6 & 1 & 0 & 1 & 0 \\ 5 & 5 & 3 & 2 & 0 & 0 & 1 \\ 7 & 1 & 3 & 3 & 6 & 0 & 0 \\ 1 & 6 & 4 & 2 & 0 & 1 & 0 \\ 7 & 2 & 2 & 4 & 0 & 0 & 3 \end{bmatrix}$$

Example 1- Undecodable

$$U = \begin{bmatrix} 5 & 4 & 7 & 6 & 7 & 6 \\ 4 & 3 & 2 & 7 & 5 & 0 \\ 7 & 2 & 3 & 6 & 0 & 6 \end{bmatrix} C = \begin{bmatrix} 5 & 3 & 6 & 2 & 7 & 3 & 6 \\ 3 & 3 & 6 & 2 & 0 & 6 & 2 \\ 6 & 5 & 7 & 1 & 5 & 2 & 2 \end{bmatrix}$$

$$P = \begin{bmatrix} \times & 4 & 7 & 1 & 2 & 5 & 6 \\ 4 & \times & 7 & 1 & 6 & 7 & 4 \\ 7 & 7 & \times & 0 & 1 & 6 & 1 \\ 1 & 1 & 0 & \times & 3 & 1 & 5 \\ 2 & 6 & 1 & 3 & \times & 4 & 7 \\ 5 & 7 & 6 & 1 & 7 & \times & 2 \\ 6 & 4 & 1 & 5 & 4 & 2 & \times \end{bmatrix}$$

Example 1- Undecodable (Cont'd)

$$E = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tilde{P} = \begin{bmatrix} \times & 7 & 5 & 3 & 2 & 1 & 1 \\ 7 & \times & 7 & 1 & 6 & 7 & 4 \\ 5 & 7 & \times & 0 & 1 & 6 & 1 \\ 3 & 1 & 0 & \times & 3 & 1 & 5 \\ 2 & 6 & 1 & 3 & \times & 4 & 7 \\ 1 & 7 & 6 & 1 & 4 & \times & 2 \\ 1 & 4 & 1 & 5 & 7 & 2 & \times \end{bmatrix}$$

$$\hat{P} = \begin{bmatrix} \times & 7 & 5 & 3 & 2 & 1 & 1 \\ 4 & 7 & 7 & 1 & 6 & 7 & 4 \\ 7 & 7 & 6 & 0 & 1 & 6 & 1 \\ 1 & 1 & 0 & 7 & 3 & 1 & 5 \\ 2 & 6 & 1 & 3 & 5 & 4 & 7 \\ 5 & 7 & 6 & 1 & 7 & 3 & 2 \\ 6 & 4 & 1 & 5 & 4 & 2 & 3 \end{bmatrix} P = \begin{bmatrix} \times & 4 & 7 & 1 & 2 & 5 & 6 \\ 4 & \times & 7 & 1 & 6 & 7 & 4 \\ 7 & 7 & \times & 0 & 1 & 6 & 1 \\ 1 & 1 & 0 & \times & 3 & 1 & 5 \\ 2 & 6 & 1 & 3 & \times & 4 & 7 \\ 5 & 7 & 6 & 1 & 7 & \times & 2 \\ 6 & 4 & 1 & 5 & 4 & 2 & \times \end{bmatrix}$$

Example 2- Incorrect Decoding

$$U = \begin{bmatrix} 0 & 2 & 3 & 0 & 2 & 2 \\ 2 & 0 & 2 & 2 & 6 & 4 \\ 3 & 2 & 6 & 2 & 4 & 5 \end{bmatrix} C = \begin{bmatrix} 7 & 4 & 5 & 3 & 0 & 0 & 5 \\ 2 & 7 & 3 & 0 & 5 & 6 & 5 \\ 4 & 4 & 5 & 5 & 4 & 6 & 2 \end{bmatrix}$$

$$P = \begin{bmatrix} \times & 4 & 5 & 6 & 0 & 0 & 3 \\ 4 & \times & 0 & 1 & 7 & 4 & 2 \\ 5 & 0 & \times & 3 & 2 & 3 & 4 \\ 6 & 1 & 3 & \times & 4 & 7 & 2 \\ 0 & 7 & 2 & 4 & \times & 2 & 3 \\ 0 & 4 & 3 & 7 & 2 & \times & 2 \\ 3 & 2 & 4 & 2 & 3 & 2 & \times \end{bmatrix}$$

Example 2- Incorrect Decoding (Cont'd)

$$E = \begin{bmatrix} 7 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tilde{P} = \begin{bmatrix} \times & 5 & 3 & 7 & 5 & 1 & 1 \\ 5 & \times & 0 & 1 & 7 & 4 & 2 \\ 3 & 0 & \times & 3 & 2 & 3 & 4 \\ 7 & 1 & 3 & \times & 4 & 7 & 2 \\ 5 & 7 & 2 & 4 & \times & 2 & 3 \\ 1 & 4 & 3 & 7 & 2 & \times & 2 \\ 1 & 2 & 4 & 2 & 3 & 2 & \times \end{bmatrix}$$

$$\hat{P} = \begin{bmatrix} 0 & \mathbf{7} & 3 & \mathbf{1} & 5 & 1 & 1 \\ 4 & 4 & 0 & 1 & 7 & 4 & 2 \\ 5 & 0 & 3 & 3 & 2 & 3 & 4 \\ 6 & 1 & 3 & 5 & 4 & 7 & 2 \\ 0 & 7 & 2 & 4 & 0 & 2 & 3 \\ 0 & 4 & 3 & 7 & 2 & 0 & 2 \\ 3 & 2 & 4 & 2 & 3 & 2 & 6 \end{bmatrix} P = \begin{bmatrix} \times & 4 & 5 & 6 & 0 & 0 & 3 \\ 4 & \times & 0 & 1 & 7 & 4 & 2 \\ 5 & 0 & \times & 3 & 2 & 3 & 4 \\ 6 & 1 & 3 & \times & 4 & 7 & 2 \\ 0 & 7 & 2 & 4 & \times & 2 & 3 \\ 0 & 4 & 3 & 7 & 2 & \times & 2 \\ 3 & 2 & 4 & 2 & 3 & 2 & \times \end{bmatrix}$$

Main Result 3

- Assume the storage nodes with errors correspond to the ℓ_0th, ℓ_1th, …, ℓ_{v-1}th columns in the received matrix $Y_{\alpha\times n}$.
- There are at least $n-k+2$ errors in each of the ℓ_0th, ℓ_1th, …, ℓ_{v-1}th columns of $\bar{G}^T Y_{\alpha\times n}$.

Main Result 4

- $\hat{P}$ be the corresponding portion of decoded codeword vector to $\tilde{P}$ and $E_P = \hat{P} \oplus \tilde{P}$ be the error pattern vector.
- Assume that the data collector accesses all storage nodes and there are v, $1 \leq v \leq \lfloor \frac{n-k+1}{2} \rfloor$, of them with errors.
- Then, there are at least $n-k+2-v$ nonzero elements in ℓ_jth column of E_P, $0 \leq j \leq v-1$, and at most v nonzero elements in the rest of columns of E_P.

$$Ep = \begin{bmatrix} \times & 2 & 0 & 6 & 0 & 0 & 0 \\ 1 & \times & 0 & 0 & 0 & 0 & 0 \\ 6 & 0 & \times & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & \times & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & \times & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & \times & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & \times \end{bmatrix}$$

Progressive Decoding

- It is not efficient to access all n storage nodes to correct errors
- Progressive decoding only accesses enough extra storage nodes to correct errors
- It needs an integrity check after the original data is reconstructed
- Integrity check: cyclic redundancy check (CRC) or cryptographic hash function
- Based on error-and-erasure decoding

Summary

- Propose the construction of least-update-complexity codes with a properly chosen systematic generator matrix
- Propose a new encoding scheme for the $[n, 2\alpha]$ error-correcting MSR codes from the generator matrix of any $[n, \alpha]$ RS codes
- The decoding scheme leads to an efficient decoding scheme that can tolerate more errors at the storage nodes
- Accesses additional storage nodes only when necessary
- More detailed results can be found in arXiv:1301.4620 [cs.IT]

Thanks

Q&A

National Science Council
行政院國家科學委員會

NSF