An Improvement of Quasi-cyclic Minimum Storage Regenerating Codes for Distributed Storage

Chenhui LI*, Songtao LIANG*

* Shanghai Key Laboratory of Intelligent Information Processing, Fudan University, Shanghai, China

11210240016@fudan.edu.cn, 11110240013@fudan.edu.cn

Abstract— **In a distributed storage system, where failure is inclined to happen at one of the storage nodes linked via network, redundancy is often introduced to increase reliability. Erasure coding is universally adopted to optimize the additional storage size for redundancy while enhancing the fault tolerance. Regenerating codes, as a class of erasure codes, make significant contribution to bandwidth-saving when recovering a failed node. A new family of regenerating codes based on quasi-cyclic codes is presented by Bernat Gastón et al. Quasi-cyclic Minimum Storage Regenerating (MSR) codes are very interesting because of their simplicity and low computational requirements. In this paper, an improvement of Quasi-cyclic MSR codes is proposed and its correctness is proven. We not only improve the constraints of construction, but also minimize the number of nodes needed to connect in a repairing process.**

Keywords— **Distributed Storage, Network Coding, Maximum Distance Separable (MDS) Codes, Regenerating Codes, Quasi-cyclic MSR Codes**

I. INTRODUCTION

Nowadays, the big data storage has been an increasingly significant research focus. The distributed storage system is a cut-edge solution and commonly used in cloud computing applications. However, node failure which easily happens in a distributed storage system leads to the availability problem of storage nodes. Redundancy is introduced to increase reliability. Replication is the simplest and widely used way but consumes too much extra storage resource and bandwidth. In comparison [1], erasure coding can enormously increase efficiency (by optimizing the additional storage size for redundancy) while maintaining reliability.

A well-known class of erasure codes is maximum distance separable (MDS) codes. Let B be the total file size over finite field F_q of size q. Assume there are n nodes in the distributed storage system, using an $[n, k]$ MDS code, a data collector(DC) can recover the entire data by connecting any k nodes (where $n>k$). We call this process *data reconstruction*. The use of erasure codes brings about a noteworthy issue called *node regeneration* which can be described as follows: when the network environment is unstable, storage nodes may be damaged or unavailable; the newcomer (which is a newly generated node) connects a subset of the other active nodes, and downloads the required data to recover the failed one. The key problem of this issue is the amount of data necessary to download, and *Regeneration Codes* are designed by Dimakis in [2][1] to deal with this problem by minimizing the bandwidth overhead while maintaining the advantage of erasure codes. A new family of regenerating codes based on quasi-cyclic codes is introduced by Bernat Gastón et al. in [5] where $d = k + 1$ and $\alpha = 2$. It is termed as Quasi-cyclic Minimum Storage Regenerating (MSR) codes which are simply constructed and produce few computations in data reconstruction and node regeneration process.

This paper is organized as follows. In Section II, some notations are stated and existing properties are illustrated. In Section III, the Quasi-cyclic MSR codes are introduced in details including the code construction, data reconstruction and node regeneration. In Section IV, we propose a new construction of Quasi-cyclic MSR codes, named Lightweight QC-MSR Codes, which improves the constrains of construction and relaxes the connection condition when repairing. Finally, in Section V, the conclusion of our work is drawn.

II. PRELIMINARY

In an $[n, k, d]$ regenerating code, the message split into B data blocks is encoded into n nodes with α fragments. A DC can recover the entire message by any k nodes. Upon failure of an individual node, a newcomer node can reconstruct it by connecting $\forall d$ accessible nodes and download β fragments from each node. Then the bandwidth for reconstruction of a node is $\gamma = d\beta$. For convenience, we present a regenerating code as $[n, k, d]$ code instead of $[n, k, d, \alpha, \beta, B]$ code.

The cut-set bound of network coding proves that the parameters of a regenerating code must satisfy the following conditions [3]:

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

Obviously, minimization of α and β reaches the lower bound of resource consumption in a repairing process, since minimizing α results in a minimum storage solution while minimizing β (for a fixed d) results in a minimum bandwidth solution. A tradeoff exists between storage and repair bandwidth deriving two extreme points, one of which is Minimum Storage Regeneration (MSR) codes by minimizing α. MSR codes have the following properties:

$$\alpha = \frac{B}{k}, \quad (2)$$

$$\beta = \frac{B}{k(d-k+1)}. \quad (3)$$

There are three varieties of repair models: exact repair, functional repair and exact repair of systematic part. In exact repair model, the regenerating node is the replicas of the failed one. Functional repair need not to guarantee the newcomer as same as the lost one but maintain the whole system with MDS-Code property and regenerating property. For exact repair of systematic part model, the data controller only need to keep the systematic part exactly the same as the failed one and the non-systematic part follows a functional repair model.

Theorem 1. (Theorem 11[7]) Linear, exact-repair MSR codes achieving the cut-set bound on the repair bandwidth do not exist for $d < 2k - 3$, in the absence of symbol extension.

III. QUASI-CYCLIC MSR CODES

In this section, we introduce a new family of MSR codes, called Quasi-cyclic MSR Codes, which predefines a vector of coefficients for exact-repair of every storage node lowering computational requirements to a large degree. These codes also succeed to avoid extra overhead of the network with a new organization of data blocks and a special set of parameters. Moreover, Quasi-cyclic MSR Codes relax the constraint that the DC repairs one failed node with any other *d* nodes, and rule that the DC can repair one lost node after carefully selecting certain nodes.

Let $\beta = 1$, as proven in [4] that one can construct a *[n, k, d]* optimal regenerating code with $\beta \neq 1$ if one can construct a *[n, k, d]* optimal MSR code with $\beta = 1$, we can derive from (2) and (3) that

$$\alpha = d - k + 1 \quad (4)$$
$$B = k\,(d - k + 1) \quad (5)$$

Quasi-cyclic MSR Codes are constructed with the following two conditions:

- $\alpha = 2$. From (4), $d = k + 1$, which means the number of nodes necessary to regenerate a failed node is $k + 1$. From (5), $B = 2k$.
- $B = n$, where *n* is the number of nodes in the network.

It shows that exact repair of Quasi-cyclic MSR codes can be achieved for $d < 2k - 3$ when $k > 4$. This complements the inexistence of exact repair shown in Theorem 1.

A. Code Construction

Quasi-cyclic MSR codes partition a file into *n* blocks, the number of which is the same as that of the nodes in the system. Let $\boldsymbol{s} = (s_1, \ldots, s_n)$ over F_q denotes a vector of the *n* nodes in the system, $\boldsymbol{\upsilon} = (\upsilon_1, \ldots, \upsilon_n)$ denotes *n* blocks of source data, and $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_n)$ denotes the redundancy coordinate of each data block. The redundancy vector $\boldsymbol{\rho}$ is defined by the equation as follows:

$$\rho_i = \sum_{j=i+1}^{i+k} \phi_{j-i}\upsilon_j \quad , i = 1, \ldots, n \ , \quad (6)$$

where $\phi_m \in F_q \setminus \{0\}$ for $m = 1, \ldots, k$ and $j = i + 1, \ldots, i + k \bmod n$, and $\phi_m = 0$ elsewhere.

As shown in **Figure 1**, for $\forall s_i\,(i = 1, \ldots, n)$, it comprises 2 fragments (as $\alpha = 2$). The first fragment stores systematic codes and the second fragment stores redundancy coordinates which are generated by using a circulant matrix as we will see later.

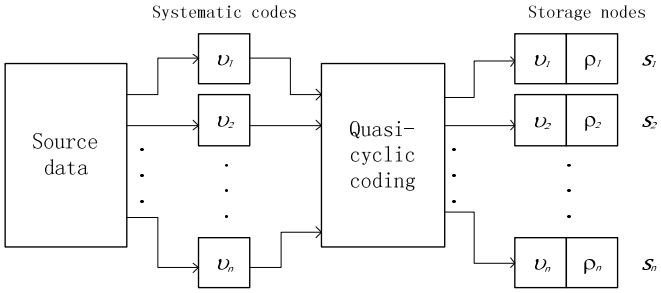


Figure 1. Constuction Process of Quasi-cyclic MSR Codes

The generator matrix of Quasi-cyclic MSR Codes is a $n \times 2n$ matrix $G = (I|\boldsymbol{\Phi})$, where I is the $n \times n$ identity matrix, and $\boldsymbol{\Phi}$ is a $n \times n$ circulant matrix given by the nonzero coefficients $\phi_1, \ldots, \phi_k$ as follows:

$$\boldsymbol{\Phi} = \begin{pmatrix} 0 & 0 & \cdots & 0 & \phi_k & \phi_{k-1} & \cdots & \phi_1 \\ \phi_1 & 0 & \cdots & 0 & 0 & \phi_k & \cdots & \phi_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \phi_k & \phi_{k-1} & \cdots & \phi_1 & 0 & 0 & \cdots & 0 \\ 0 & \phi_k & \cdots & \phi_2 & \phi_1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \phi_k & \phi_{k-1} & \phi_{k-2} & \cdots & 0 \end{pmatrix} \quad (7)$$

Each row of the matrix *G* is the encoding of one storage node, and each node is represented by two columns, the first one calculated from I and the other one from $\boldsymbol{\Phi}$. Actually, the node s_i, which stores (υ_i, ρ_i), is also defined by the following equation:

$$(\upsilon_i, \rho_i) = (\boldsymbol{\upsilon} I^{\{i\}}, \boldsymbol{\upsilon} \boldsymbol{\Phi}^{\{i\}}) \quad (8)$$

B. Data Reconstruction and Node Regeneration

With the above construction algorithm, Quasi-cyclic MSR Codes can satisfy the minimum bound of both the number of connection when reconstructing the source data and the necessary bandwidth when regenerating a lost node.

It has been proved in [5] that the DC can reconstruct a file by connecting arbitrary k nodes $\boldsymbol{s'} = \{ s_1, \ldots, s_k \} \in \boldsymbol{s}$ and downloading $\{(\upsilon_1, \rho_1), \ldots, (\upsilon_k, \rho_k)\}$. This conclusion is derived from the validation that submatrix $G[\boldsymbol{s}]$ has at least B different nonzero unknowns and that $\Pi_{s \in \boldsymbol{s}} \Delta(G[\boldsymbol{s}]) \neq 0$, $\Pi_{s \in \boldsymbol{s}} \Delta(\Phi_S^S) \neq 0$ over F_q.

As to node regeneration, Quasi-cyclic MSR codes choose not to comply with the convention but to select d nodes according to a predesigned algorithm. Assume node s_i is failed, and then the regeneration process can be achieved as follows:

- Firstly, download the first fragments from the next k blocks of node s_i in the sequence. Remember that due to the circulant scheme, the next node of node s_n is node s_1. Corresponding to the coefficient vector $\boldsymbol{\rho}$, the redundancy coordinate of the lost node can be calculated and will be as the second fragment of the new node.
- Secondly, download the second fragment from the previous node of node s_i in the sequence following the same circulant scheme. Solving some easy linear operations, the systematic codes of the lost node can be obtained and will be as the first fragment of the new node.

IV. LIGHTWEIGHT QC-MSR CODES

In this section, an improvement of Quasi-cyclic MSR Codes is presented and illustrated, in which we revise the constraints of construction deriving a more flexible construction algorithm and we further design a new node regeneration scheme minimizing the number of nodes necessarily connected in a repairing process. Specifically, in Subsection **IV.A**, we propose the new constraints and prove its validity, which brings about a new construction algorithm; in Subsection **IV. B**, according to the new property brought by the constraint improvement, we prove d can be reduced to a lower value at some circumstances and design a new node regeneration algorithm to implement the minimization, which means the bandwidth boundary may be less than $k+1$ in this encoding system; in Subsection **IV. C**, we compare the performance of Lightweight QC-MSR codes with that of Quasi-cyclic MSR codes, illustrate the results by a chart and make a discussion about the results.

A. The New Scheme of Construction

For an MSR code, the DC can repair a failed node by connecting arbitrary d nodes. In this subsection, we present one class of Quasi-cyclic MSR codes with $\alpha = 2$, $B = n = 2k$, $\beta = 1$. Let

$$\hat{\rho}_i = \sum_{l=1}^{k} \hat{\phi}_l \upsilon_{i+t}, i = 1, \dots, n, \quad (9)$$

where $\hat{\phi}_l \in F_q$ for $l = 1, \dots, k$ and $\hat{\phi}_l = 0$ elsewhere, and $t = \begin{cases} l + k - 1, & (i + l + k - 1) \le 2k \\ l - k - 1, & (i + l + k - 1) > 2k \end{cases}$.

Then the generator matrix can be defined as $\hat{G} = (I | \hat{\Phi})$. In this paper, we call these codes Lightweight Quasi-cyclic (QC) MSR Codes.

Notice that in the construction scheme shown in Section III, the coefficients of the redundancy vector $\boldsymbol{\rho}$ are all restricted to be nonzero. However, in this new construction algorithm, we relax this constraint and then attain a new property as shown and proven in Lemma 1.

Lemma 1. For a Lightweight QC-MSR Code with $\hat{\rho}_i$ defined in (9), $\hat{\phi}_1$ is always nonzero.

Proof: We can prove it by contradiction. Assume $\hat{\phi}_1 = 0$, the DC chooses node 1, □, k to recover the entire source file. Denote the linear space combined by the n column vectors W. Then we can calculate by (9) that $\text{span(W)} = [\upsilon_1, \dots, \upsilon_k, \sum_{l=1}^{k} \hat{\phi}_l \upsilon_{l+k}, \dots, \hat{\phi}_1 \upsilon_{2k} + \sum_{l=2}^{k} \hat{\phi}_l \upsilon_{l-1}]$. Since $\hat{\phi}_1 = 0$, $\upsilon_{1+k} \notin \text{span(W)}$, rank(W) < n, which violates the MDS-property of MSR codes (that any k out of n suffices to recover). In fact, the DC can always recover the entire source data if $\hat{\phi}_1 \neq 0$. From node s_i and s_j $(i \neq j)$, the DC attains that $\text{span}(s_i) = [\upsilon_i, \upsilon_{i+k} + \sum_{l=2}^{k} \hat{\phi}_l \upsilon_{i+t}]$ and $\text{span}(s_j) = [\upsilon_j, \upsilon_{j+k} + \sum_{l=2}^{k} \hat{\phi}_l \upsilon_{j+t}]$. It can be seen obviously that $\text{span}(s_i) \cap \text{span}(s_j) = \emptyset$. So, arbitrary k nodes can reconstruct the entire source file.

B. Node Regeneration with d = min{2wt($\hat{\rho}_i$) , k+1}

Theorem 2. The DC can repair a failed node of the Lightweight QC-MSR codes by connecting d nodes with $d = min\{2wt(\hat{\rho}_i), k+1\}$.

Proof: Assume node s_i is failed, and then we need recover its data fragment υ_i and redundancy coefficient vector $\hat{\Phi}^{\{i\}} = [0, \dots, \phi_1, \dots, \phi_k, 0, \dots]^T$. Let $w = 2\ wt(\hat{\rho}_i)$ (because $\hat{\boldsymbol{\rho}}$ has cyclic structure, $wt(\hat{\rho}_1) = wt(\hat{\rho}_2) = \cdots = wt(\hat{\rho}_n)$), which denotes the weight of $\hat{\rho}_i$. It also means that there are w nonzero terms in $\phi_1, \dots, \phi_k$, $w \le k$. To regenerate the second fragment of the failed node, instead of accessing k nodes in the system, we only need w nodes to calculate the linear sum. This can be achieved by connecting the first fragments of w nodes among node $s_{i+k}, \dots, s_{i+2k}$ (for s_m with $m = i + t > 2k$, let $m = i + t - 2k$) corresponding to the positions of nonzero coordinators in $\hat{\Phi}^{\{i\}}$. The process of recovering υ_i is a little more complicate, that firstly we search for one node s_j whose redundancy coordinator expression contains υ_i, and then access the relevant nodes of the other w-1 nonzero terms in its redundancy coordinator and download their first fragments. So, the total connection number equals $w + 1 + (w-1) = 2w = 2\ wt(\hat{\rho}_i)$. Notice that this chosen of w-1 nodes may have common with the former chosen of w nodes and the total number of nodes to be accessed are no more that k + 1. So the number of connecting nodes when repairing is $min\{2wt(\hat{\rho}_i), k+1\}$.

In order to implement the above bandwidth-saving exact-repair of a failed node (with a lower value of d), we design a node-regeneration algorithm as follows:

Algorithm 1. A node-regeneration Algorithm for Lightweight QC- MSR Codes

Input: the generator matrix $\hat{G}$ and the identifier of the failed node i.

Output: the storage content of the failed node s_i $(\upsilon_i, \hat{\rho}_i)$▷

1). Let $\widehat{\Phi}^{\{i\}T} = [0, \ldots, \phi_1, \ldots, \phi_k, 0, \ldots]$, $\boldsymbol{\upsilon} = [\upsilon_1, \ldots, \upsilon_n]$. Match $\boldsymbol{\upsilon}$ with $\widehat{\Phi}^{\{i\}T}$ column by column, and then connect each node s_j and download the first fragment if the value of column j in $\widehat{\Phi}^{\{i\}T}$ is nonzero. Thus, we obtain w data blocks to calculate the redundancy fragment of s_i.

2). Let $\boldsymbol{\phi} = [\phi_k, \ldots, \phi_1]$, $\boldsymbol{\upsilon}' = [\upsilon_{i+1}, \ldots, \upsilon_{i+k}]$. Match $\boldsymbol{\upsilon}'$ with $\boldsymbol{\phi}$ column by column, and then find the first nonzero term ϕ_l in $\boldsymbol{\phi}$ and connect corresponding node s_m $(m = i + k - l + 1)$, and then download its second fragment.

3). Let $\widehat{\Phi}^{\{m\}T} = [0, \ldots, \phi_1, \ldots, \phi_k, 0, \ldots]$, and match $\boldsymbol{\upsilon}$ with $\widehat{\Phi}^{\{m\}T}$ column by column. For each node s_r, if the value of column r in $\widehat{\Phi}^{\{m\}T}$ is nonzero and the node has not been connected, connect it and download its first fragment.

4). Compute υ_i with 1+(w-1) fragments downloaded at *step 2)&3),* and compute $\hat{\rho}_i$ w fragments downloaded at *step 1).* Finally, s_i is regenerated by $(\upsilon_i, \hat{\rho}_i)$ as defined in (8).

C. Performance Analysis

We make a set of n and compute the relevant d of both Lightweight QC-MSR codes and Quasi-cyclic MSR codes. As to Lightweight QC-MSR codes, the value of d shown in **Table 1** and **Figure 2** is the weight mean of d for different value of w. Since d equals the consumption of bandwidth (which is calculated by $d \times \beta$) in a repairing process in both schemes, the smaller the value of d is, the better the performance of codes is.

TABLE 1. REPARING BANDWIDTH OF TWO CODING SCHEMES

Code Name	**The Repairing Bandwidth: *d***				
The Number of Nodes: *n*	**10**	**20**	**30**	**40**	**50**
Quasi-cyclic MSR codes	6	11	16	21	26
Lightweight QC-MSR codes	4.80	8.50	12.27	16.00	19.76

The comparison of the results is illustrated in **Figure 2** as follows. It is obvious that our new scheme, named as Lightweight QC-MSR codes, consume less bandwidth than Quasi-cyclic MSR codes when regenerating a failed node. Thus, Lightweight QC-MSR codes improve the performance. Moreover, if we let r denotes the rate of bandwidth that Lightweight QC-MSR codes save compared with Quasi-cyclic MSR codes, it can be calculated according to **Table 1** that r={20%, 22.7%, 23.3%, 23.8%, 24%} when n={10, 20, 30, 40, 50}. We can deduce that the saving rate is considerable and increases with the increase of n.

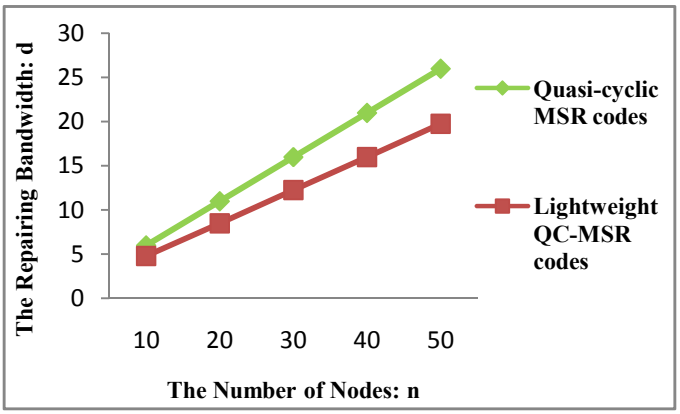


Figure 2. Comparison of the Repairing Bandwidth of Two Coding Schemes

V. CONCLUSIONS

In this paper, we focus on a family of Quasi-cyclic MSR codes with particular assignment of α and B. After deliberate consideration of their construction algorithm and properties, we propose an improvement of Quasi-cyclic MSR codes and name the new class Lightweight QC-MSR codes and then prove its correctness. With the new scheme of construction, Lightweight QC-MSR codes relax the connection constrains in a node regeneration process. We present a complete and detailed algorithm to achieve node regeneration in such conditions. Successively, we compare the performance of two coding schemes with quantitative analysis and discuss the results.

The further research involves the discussion about whether exact-repair of a failed node exists for connecting arbitrary (k+1) nodes in this Quasi-cyclic MSR coding system.

REFERENCES

[1] H. Weatherspoon and J. Kubiatowicz, "Erasure co ding vs. replication: A quantitative comparison," in *In Proceedings of the First International Workshop on Peer-to-Peer Systems (IPTPS) 2002*, 2002, pp. 328–338.

[2] Dimakis A G, Godfrey P B, Wu Y, et al., "Network coding for distributed storage systems," *Information Theory, IEEE Transactions* on, 2010, 56(9): 4539-4551.

[3] Y. Wu, A. G. Dimakis, and K. Ramchandran, "Deterministic Regenerating Codes for Distributed Storage," in *Proc. Allerton Conf.,Urbana-Champaign*, Sep. 2007.

[4] K. Rashmi, N. Shah, and P. Kumar, "Optimal exact-regenerating codes for distributed storage at the msr and mbr points via a product-matrix construction," *IEEE Trans. Inf. Theory*, in *eprint arXiv*: 1005.4178, 2010.

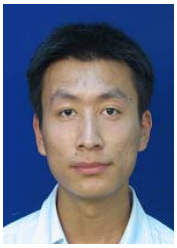
[5] Gastn B, Pujol J, Villanueva M. "Quasi-cyclic Minimum Storage Regenerating Codes for distributed data compression", *IEEE Data Compression Conference,* DOI 10.1109/ DCC.2011.11.

[6] C. Suh and K. Ramchandran, "Exact regeneration codes for distributed storage repair using interference alignment," in *Proc. IEEE Int. Symp. Inf. Theory*, Jun. 2010. [Online]. Available: http://arxiv.org/abs/1001.0107v2 .

[7] Shah N B, Rashmi K V, Kumar P V, et al. "Interference alignment in regenerating codes for distributed storage: Necessity and code constructions," *Information Theory, IEEE Transactions* on, 2012, 58(4): 2134-2158.

[8] Yuan Li and Haibin Kan, "Complex Orthogonal Designs with Forbidden 2×2 Submatrices," *IEEE Transactions on Information Theory*, Vol. 58, No. 7, July 2012.
[9] Haibin Kan and Hong Shen, "Lower bounds of the minimal delays of complex orthogonal designs with maximal rates," *IEEE Transactions on Communications*, Vol. 54, No. 3, March 2006.
[10] Haibin Kan and Hong Shen, "A relation between the characteristic generators of a linear code and its dual," *IEEE Transactions on Information Theory*, Vol. 51, No. 3, March 2005.
[11] Haibin Kan and Hong Shen, "A counterexample for the open problem on the minimal delay of orthogonal designs with maximal rates," *IEEE Transactions on Information Theory*, Vol. 51, No. 1, January 2005.
[12] Songtao Liang and Haibin Kan, "Practically Feasible Design for Convolutional Network Code," *IEICE Transactions on Fundamentals*, Vol.E96-A, No.9, Sep. 2013.
[13] C. Yuan and Haibin Kan, "A characterization of solvability for a kind of networks," *Science in China(F)*, Vol.55, No.4, 747-754, 2012
[14] Gastn B, Pujol J, Villanueva M., "Quasi-cyclic regenerating codes," *arXiv preprint,* arXiv: 1209.3977, 2012.

Chenhui Li (SM'13) received the BE degree in Information Security from Shanghai Jiao Tong University, China, in 2011. Since then, she has been a postgraduate student of Shanghai Key Laboratory of Intelligent Information Processing, Fudan University, Shanghai, China, and became a Student Member of IEEE in 2013. Her research interests include network coding, information security and cryptography.



Songtao Liang received his BS and MS degrees from the School of Computer Science, Harbin Institute of Technology and Fudan university, China, in 2008 and 2011. He is currently pursuing PhD degree at the School of Computer in Fudan university. His research interests include network coding and distributed storage.