THOMSON REUTERS
ENDNOTE

[Rasal* 5(12), :December , 2016]
IC™ Value: 3.00

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

# CAUCHY CODING APPROACH WITH FILE AUDITING TECHNIQUE IN CLOUD

**Mr. Yuvraj P. Rasal**
Department of Computer Engineering, SND College of Engineering & Research Centre,
Yeola, Nashik-423401, Maharashtra, India

**DOI**: 10.5281/zenodo.221120

## ABSTRACT

Users of cloud storage usually assign different redundancy configurations (i.e., ðk; m;wÞ) of erasure codes, depending on the desired balance between performance and fault tolerance. Our study finds that with very low probability, one coding scheme chosen by rules of thumb, for a given redundancy configuration performs best. In this paper, we propose CaCo, an efficient Cauchy coding approach for data storage in the cloud. First, CaCo uses Cauchy matrix heuristics to produce a matrix set. Second, for each matrix in this set, CaCo uses XOR schedule heuristics to generate a series of schedules. Finally, CaCo selects the shortest one from all the produced schedules. In such a way, CaCo has the ability to identify an optimal coding scheme, within the capability of the current state of the art, for an arbitrary given redundancy configuration. By leverage of CaCo's nature of ease to parallelize, we boost significantly the performance of the selection process with abundant computational resources in the cloud.

**KEYWORDS**: Cloud storage, Cauchy matrix, fault tolerance, Reed-Solomon codes, XOR scheduling

## INTRODUCTION

CLOUD storage is built up of numerous inexpensive and unreliable components, which leads to a decrease in the overall mean time between failures (MTBF). As storage systems grow in scale and are deployed over wider networks,component failures have been more common, and requirements for fault tolerance have been further increased. So, the failure protection offered by the standard RAID levels has been no longer sufficient in many cases, and storage designers are considering how to tolerate larger numbers of failures,. For example, Google's cloud storage, Windows Azure Storage, Ocean Store, Disk Reduce, HAIL, and others all tolerate at least three failures. To tolerate more failures than RAID, many storage systems employ Reed-Solomon (RS) codes for fault-tolerance. Reed-Solomon coding has been around for decades, and has a sound theoretical basis. As an erasure code, Reed-Solomon code is widely used in the field of data storage. Given k data blocks and a positive integer m, Reed- Solomon codes can encode the content of data blocks into m coding blocks, so that the storage system is resilient to any m disk failures. Reed-Solomon codes operate on binary words of data, and each word is composed of w bits, where 2w _ k þ m. In the rest of this paper, we denote a combination of k, m, and w by a redundancy configuration ðk; m;wÞ, where k data blocks are encoded into m coding blocks, with the coding unit of w-bit words. Reed-Solomon codes treat each word as a number between 0 and 2w _ 1, and employ Galois Field arithmetic over GF(2w). In GF(2w), addition is performed by bitwise exclusive-or (XOR), while multiplication is more complicated, typically implemented with look-ups to logarithm tables. Thus, Reed-Solomon codes are considered expensive. Cauchy Reed-Solomon (CRS) codes improve Reed-Solomon codes by using neat projection to convert Galois Field multiplications into XOR operations. Currently, CRS codes represent the best performing general purpose erasure codes for storage systems. In addition, CRS coding operates on entire strips across multiple storage devices instead of operating on single words. In particular, strips are partitioned into w packets, and these packets may be large. Fig. 1 illustrates a typical architecture for a cloud storage system with data coding. The redundancy configuration of the system is k ¼ 4 and m ¼ 2. With CRS codes, k data blocks are encoded into m coding blocks. In such a way, the system can tolerate any m disk failures without data loss. Note that those k data blocks and m coding blocks should be stored on different data nodes. Otherwise, the failure of one node may lead to multiple faults in the same group of n ¼ k þ m blocks. A

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

matrix-vector product ,AX ¼ B, is central to CRS codes. All elements of A, X, and B are bits. Here, A is defined as a Cauchy matrix, X stands for data strips, and B stands for strips of coding information. Since the act of CRS coding involves only XOR operations, the number of XORs required

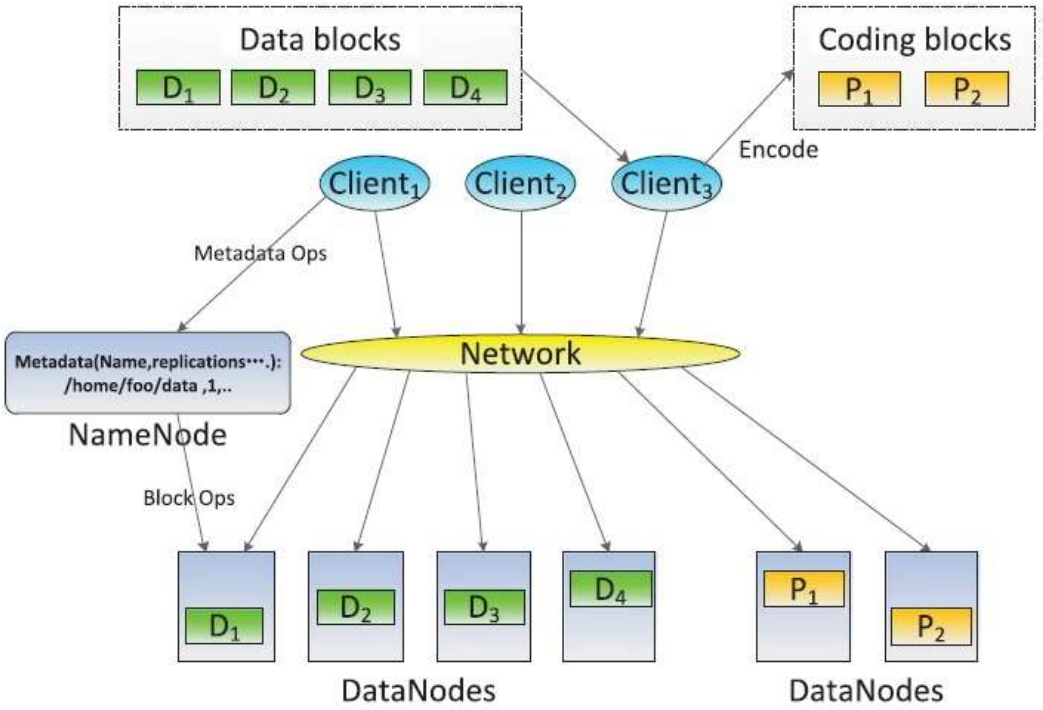


***Fig1. A distributed architecture for a cloud storage system***

A distributed architecture for a cloud storage system with data coding, where k ¼ 4, and m ¼ 2. by a CRS code impinges directly upon the performance of encoding or decoding. While there are other factors that impact performance, especially cache behavior and device latency, reducing XORs is a reliable and effective way to improve the performance of a code. For example, nearly all special-purpose erasure codes, from RAID-6 codes (e.g., EVENODD RDP [14], X [15], and P [16] codes) to codes for larger systems (e.g., STAR [17], T [18], and WEAVER [19] codes), aim at minimizing XOR operations at their core. The goal of this paper is to find a best-performing coding scheme that performs data coding with the fewest XOR operations Users of cloud storage usually assign different redundancy configurations (i.e., (k, m, w)) of erasure codes, depending on the desired balance between performance and fault tolerance. Our study finds that with very low probability, one coding scheme chosen by rules of thumb, for a given redundancy configuration, performs best. In this paper, we propose CaCo, an efficient Cauchy coding approach for data storage in the cloud.

## SYSTEM OVERVIEW

A graphical representation of the overall framework. Given a redundancy configuration ðk; m; wÞ, our goal is to find a Cauchy matrix, whose schedule is desired to be the shortest. In this paper, we propose CaCo, a coding approach that incorporates all existing matrix and schedule heuristics, and therefore is able to discover an optimal solution for data coding in a cloud storage system, within the capability of the current state of the art

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

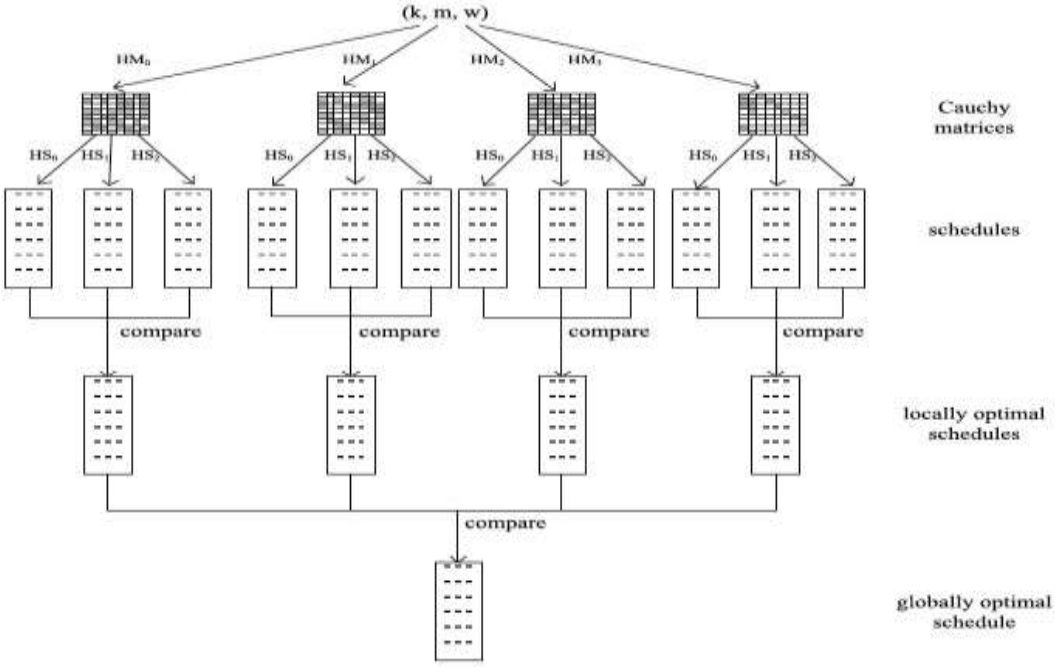


*Fig2. A graphical representation of the overall framework*

## SYSTEM ANALYSIS

Given a redundancy configuration (k, m, w) our goal is to find a Cauchy matrix, whose schedule is desired to be the shortest. In this paper, we propose CaCo, a coding approach that incorporates all existing matrix and scheduleheuristics, and therefore is able to discover an optimal solution for data coding in a cloud storage system, within the capability of the current state of the art.

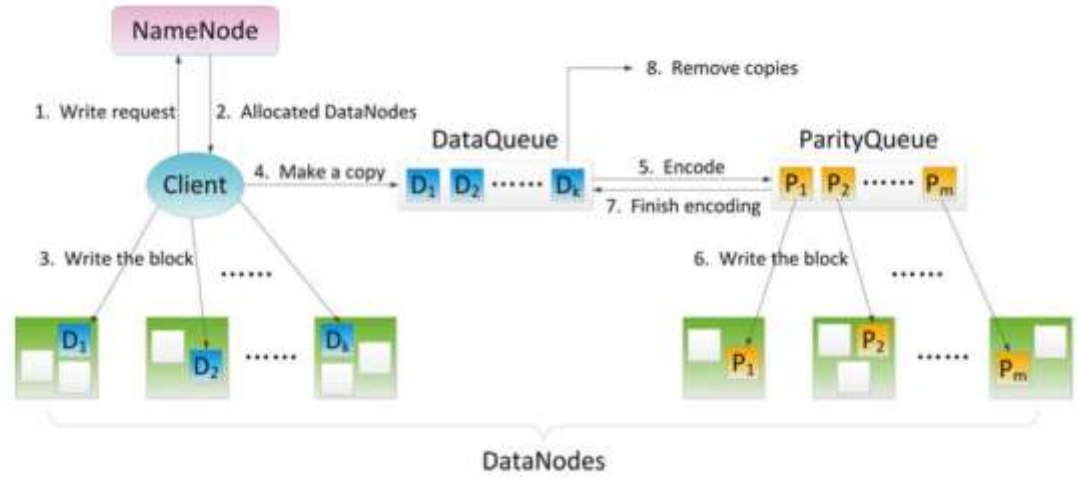


*Fig3. Block Diagram of Proposed System*

**Generating Cauchy matrices.**

Selecting the best one from Cauchy matrices using the enumeration method is a combinatorial problem. Given redundancy configuration (10; 6; 8), the magnitude of the matrices to be constructed can be up to 1029, and it is unrealistic to enumerate them. We cannot even determine which one of the matrices will produce better schedules. In theca Co approach, we choose only a certain number of them for scheduling.

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

***1. Constructing schedules for each matrix.***

For each matrix mi(0 <i< p) in the set Sm, we pass the parameters including k, m, w and pointer of the matrix tithe function do schedule (int k,int m,int w,int *matrix) to perform q heuristics in the function, such as Uber-CSHR,X-Sets, and so on. In this manner, we get a set of schedules, denoted as Ssi={s0i, s1i, . . . ,sq-1i}. If there appears good heuristic for scheduling at a later date, we can add into the function do schedule.

***2. Selecting the locally optimal schedule for each matrix.***

For each matrix mi(0 <i< p) in the set Sm, we select the shortest schedule from the set Ss;i, denoted as si, so that we get a set of matrices and their shortest schedules, denoted as S = {(m0, s0); (m1; s1); . . . ; (mp_1; sp_1)}.For mi in the set Sm, we can encode data in an order of XORs given by si. In this way, the times of XOR operations no longer have direct relationship with the density of the matrix. Therefore, scheduling excludes the influence of the lower limit of the number of ones in the matrix, so the performance improves significantly.

***3. Selecting the globally optimal solution***

From the collection of combinations of Cauchy matrix and schedule, namely {(m0; s0); (m1; s1); . . . ; (mp_1; sp_1)}, we choose the combinations with the shortest schedule. On this basis, for better performance, we tend to select the one containing the fewest ones in the matrix to be (m best; sbest).Once selected, sbest can be used for encoding data.

## RESULTS AND DISCUSSION

**Constructing Cauchy Matrices**

While using a binary Cauchy matrix for CRS coding, the number of XORs depends on the number of ones in the matrix. So, in order to get better performance, the number of ones in the binary Cauchy matrix should be as few as possible. To discover an optimal matrix from numerous Cauchy matrices, the simplest way is to enumerate them. Given a redundancy configuration ðk; m;wÞ, we can construct ( 2w kþm)(kþm k ) Cauchy matrices, each of which can be used for encoding. Therefore, the enumeration method is applicable only when the values of k, m, and w are small. Otherwise, the running time for enumerating all the matrices will be unacceptable, because the number of Cauchy matrices is a combinatorial problem. Some heuristics such as Original [11], Optimizing Cauchy [20] and Cauchy Good can generate a good matrix which contains fewer ones for larger w, but it may not be the optimal one.

To construct a Cauchy matrix, called GCðk; m;wÞ, the Optimizing Cauchy heuristic first constructs ð2w _ 2wÞ matrix, denoted as ONESðwÞ, whose element ði; jÞ represents the number of ones in the binary matrix Mð1=ði þ jÞÞ.Fig. 4a shows the matrix ONESð3Þ. Then we select two disjoint sets X ¼ fx1; . . . ; xmg and Y ¼ fy1; . . . ; ykg from f0; 1; . . . ; 2w _ 1g, as follows. _ When k ¼ m and k is a power of two, for k > 2, GCðk; k;wÞ contains the elements of GCðk=2; k=2; wÞ, and GCð2; 2; wÞ always contains the column set Y ¼ f1; 2g. For example, GCð4; 4; 3Þ is shown in Fig. 4b.

1.When k ¼ m and k is not a power of two, we define GCðk; k;wÞ by constructing GCðk0; k0; wÞ first, where k0 > k and k0 is a power of two. Then we delete redundant rows and columns alternately until we get a k _ k matrix. As Fig. 3c shows, GCð3; 3; 3Þ is defined by deleting one row and one column from GCð4; 4; 3Þ.

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

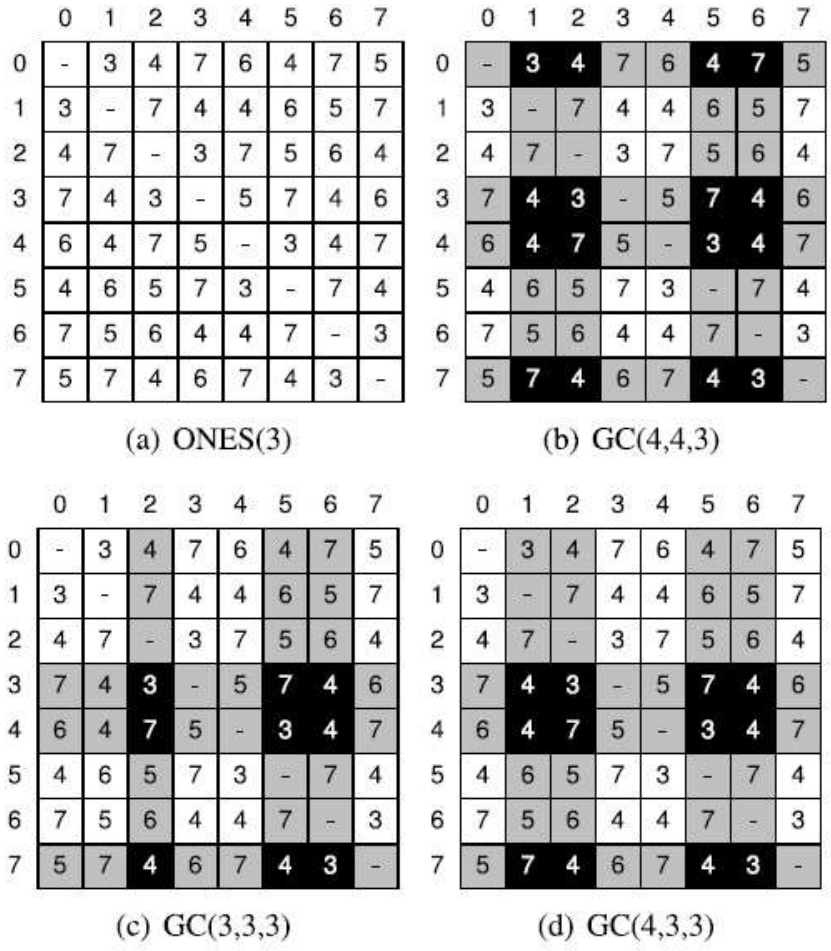


***Fig4. Block Diagram of Proposed System***

2.When k 6¼ m, we define GCðk; m;wÞ by constructing GCðminðk;mÞ; minðk;mÞ; wÞ first, and then add some rows or columns correspondingly. As Fig. 4d shows, we construct GCð4; 3; 3Þ by adding one column to GCð3; 3; 3Þ. With the divide (defined over Galois field) Cauchy Good heuristic, we first construct a Cauchy matrix called GM. Then every element of GM such as in column j by GM0;j, such that GM is updated and the elements of row 0 are all "1". In the rest of the rows, such as row i, we count the number of ones, recorded as N. Then we divide the elements of row i by GMi;j, and respectively count the number of ones, denoted as Nj (j 2 ½0; k _ 1_). Finally, select the minimum from fN;N0; . . .;Nk_1g and do the operations that generate it. Thuswe succeed in constructing a matrix using Cauchy Good heuristic. The two heuristics above can produce a binary matrix which contains fewer ones; however, it may not be the optimal one in the numerous Cauchy matrices. The research on how to reduce the number of XOR operations in the process of erasure coding has revealed that the number of ones in a Cauchy matrix has lower bounds. Therefore, only by reducing the density of the Cauchy matrix, it is difficult to improve the encoding performance greatly.

## CONCLUSION

In this paper, we propose CaCo, a new approach that incorporates all existing matrix and schedule heuristics, and thus is able to identify an optimal coding scheme within the capability of the current state of the art for given redundancy configuration. The selection process of has an acceptable complexity and can be accelerated by parallel computing. It should also be noticed that the selection process is once for all. In Future Other code properties, like the amount of data required for recovery and degraded reads may limit performance more than

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

the CPU overhead. We look forward to addressing these challenges in the future.

## ACKNOWLEDGEMENT

The authors are grateful to Xinkui Qu for providing helpful comments and assistance with our experimentation. This work was supported by the National Natural Science Foundation of China under Grants 61170008, 61272055, 61433008, and U1435216, the National Grand Fundamental Research 973 Program of China under Grant No. 2014CB340402, and the National High Technology Research and Development Program of China under Grant 2013AA01A210.

## REFERENCES

1. L. N. Bairavasundaram, A. C. Arpaci-Dusseau, R. H. Arpaci- Dusseau, G. R. Goodson, and B. Schroeder, “An analysis of data corruption in the storage stack,” Trans. Storage, vol. 4, pp. 8:1–8:28, Nov. 2008.
2. J. L. Hafner, V. Deenadhayalan, W. Belluomini, and K. Rao, “Undetected disk errors in raid arrays,” IBM J. Res. Develop., vol. 52, pp. 413–425, Jul. 2008
3. D. Ford, F. Labelle, F. I. Popovici, M. Stokely, V.-A. Truong, L. Barroso, C. Grimes, and S. Quinlan, “Availability in globally distributed storage systems,” in Proc. 9th USENIX Symp. Oper. Syst. Des. Implementation, 2010, pp. 61–74
4. B. Calder, J. Wang, A. Ogus, N. Nilakantan, A. Skjolsvold, S. McKelvie, Y. Xu, S. Srivastav, J. Wu, H. Simitci, J. Haridas, C. Uddaraju, H. Khatri, A. Edwards, V. Bedekar, S. Mainali, R. Abbasi, A. Agarwal, M. F. u. Haq, M. I. u. Haq, D. Bhardwaj, S. Dayanand, A. Adusumilli, M. McNett, S. Sankaran, K. Manivannan, and L. Rigas, “Windows Azure storage: A highly available cloud storage service with strong consistency,” in Proc. 23rd ACM Symp. Oper. Syst. Principles, New York, NY, USA, 2011, pp. 143–157.
5. J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao, “Oceanstore: An architecture for global-scale persistent storage,” SIGPLAN Notices, vol. 35, pp. 190–201, Nov. 2000
6. B. Fan, W. Tantisiriroj, L. Xiao, and G. Gibson, “Diskreduce: Raid for data-intensive scalable computing,” in Proc. 4th Annu. Workshop Petascale Data Storage, New York, NY, USA, 2009, pp. 6–10
7. K. D. Bowers, A. Juels, and A. Oprea, “Hail: A high-availability and integrity layer for cloud storage,” in Proc. 16th ACM Conf. Comput. Commun. Security, New York, NY, USA, 2009, pp. 187–198
8. G. Xu, F. Wang, H. Zhang, and J. Li, “Redundant data composition of peers in p2p streaming systems using Cauchy Reed-Solomon codes,” in Proc. 6th Int. Conf. Fuzzy Syst. Knowl. Discovery- Volume 2, Piscataway, NJ, USA, 2009, pp. 499–503
9. J. S. Plank, “A tutorial on Reed-Solomon coding for faulttolerance in raid-like systems,” Softw. Pract. Experience, vol. 27, pp. 995–1012, Sept. 1997
10. I. S. Reed and G. Solomon, “Polynomial codes over certain finite fields,” J. Soc. Ind. Appl. Math., vol. 8, no. 2, pp. 300–304, 1960.
11. J. Bl€omer, M. Kalfane, M. Karpinski, R. Karp, M. Luby, and D. Zuckerman, “An xor-based erasure-resilient coding scheme,” International Computer Science Institute, Berkeley, California, USA, 1995.
12. J. S. Plank, J. Luo, C. D. Schuman, L. Xu, and Z. Wilcox-O’Hearn, “A performance evaluation and examination of open-source erasure coding libraries for storage,” in Proc. 7th Conf. File Storage Technol., Berkeley, CA, USA, 2009, pp. 253–265.
13. M. Blaum, J. Brady, J. Bruck, and J. Menon, “Evenodd: An efficient scheme for tolerating double disk failures in raid architectures,” IEEE Trans. Comput., vol. 44, no. 2, pp. 192–202, Feb. 1995.
14. P. Corbett, B. English, A. Goel, T. Grcanac, S. Kleiman, J. Leong, and S. Sankar, “Row-diagonal parity for double disk failure correction,” in Proc. 3rd USENIX Conf. File Storage Technol., Berkeley, CA, USA, 2004, pp. 1–1.
15. L. Xu and J. Bruck, “X-code: MDS array codes with optimal encoding,” IEEE Trans. Inform. Theory, vol. 45, no. 1, pp. 272–276, Jan. 1999.

ISSN: 2277-9655
Impact Factor: 4.116
CODEN: IJESS7

16. C. Jin, H. Jiang, D. Feng, and L. Tian, "P-code: A new raid-6 code with optimal properties," in Proc. 23rd Int. Conf. Supercomput., New York, NY, USA, 2009, pp. 360–369. Fig. 12. Performance comparison of data decoding between CaCo and Hadoop-EC when three data disks fail. 446 IEEE TRANSACTIONS ON COMPUTERS, VOL. 65, NO. 2, FEBRUARY 201
17. C. Huang and L. Xu, "Star: An efficient coding scheme for correcting triple storage node failures," in Proc. 4th Conf. USENIX Conf. File Storage Technol.-Volume 4, Berkeley, CA, USA, 2005, pp. 15–15
18. S. Lin, G. Wang, D. Stones, J. Liu, and X. Liu, "T-code: 3-erasure longest lowest-density mds codes," IEEE J. Sel. Areas Commun., vol. 28, no. 2, pp. 289–296, Feb. 2010
19. J. L. Hafner, "Weaver codes: Highly fault tolerant erasure codes for storage systems," in Proc. 4th Conf. USENIX Conf. File Storage Technol.-Volume 4, Berkeley, CA, USA, 2005, pp. 16–16.
20. J. S. Plank and L. Xu, "Optimizing Cauchy Reed-Solomon codes for fault-tolerant network storage applications," in Proc. 5th IEEE Int. Symp. Netw. Comput. Appl., Washington, DC, USA, 2006, pp. 173–180.