International Journal of Applied Engineering Research ISSN 0973-4562 Volume 10, Number 15 (2015) pp 35167-35173

A Novel Approach To Tornado Codes For Archival Storage System

N.Gopika rani
Assistant Professor(Sl.Grade),
CSE department
PSG College of Technology
Coimbatore
9994153301
gopika79@yahoo.com

Dr.G.Sudha Sadasivam
Professor, CSE department
PSG College of Technology
Coimbatore
919843112267
sudhasadhasivam@yahoo.com

G.vignesh
Student, CSE department
PSG College of Technology
Coimbatore
9003390795
vignesh.accetain@gmail.com

Abstract- Large amount of data is being stored in archival storage system for a long period of time. While retrieving the file from the storage, there is a chance of getting errors due to file corruption. This problem can be resolved by storing the file with encryption technique in archival storage. Existing system using LDPC codes for archival storage is unable to eliminate the problems. The proposed technology generates a tornado graph and stores it in RAID storage system. This RAID storage system is used for high fault tolerance. The encryption technology used here is tornado code encoding. Presently these codes are used in the network transmission lines. It improves fault tolerance when retrieve the data from storage system.

Keywords: LDPC codes, Tornado Codes, RAID, Encryption, Archival storage

Introduction (Section Header in 12pt. Bold)

This work examines a class of Low Density Parity Check (LDPC) erasure codes called tornado codes for archival storage systems. Tornado codes are generated with cascaded Tornado graphs. Tornado graphs are used to analyze the fault tolerance and it is possible to find the worst case failure scenarios in graphs and remove the critical nodes sets which can cause tornado codes to fail. The graph construction procedure is used to construct a tornado code storage combined with RAID storage system for increased throughput and fault tolerance.

Low-density parity-check (LDPC) codes are a class of linear block codes. The name comes from the characteristic of their parity-check matrix which contains only a few 1's in comparison to the amount of 0's. Two different possibilities are available to represent the LDPC codes. These linear block codes are described by matrices and graphical representation. A bipartite graph is otherwise called as digraph, which is a set of vertices composed of two disjoint sets such that no two vertices within the same set are adjacent vertices. A bipartite graph is a special case of a *k*-partite graph with k=2. Bipartite graphs are equivalent to two-colorable graphs. All acyclic graphs are bipartite. A graph is bipartite if all its cycles are of same length.

The basic unit of a Tornado code is a bipartite low density parity check (LDPC) graph. A series of data nodes represents the original data to be stored and check nodes are derived to provide redundancy. When calculating parity using XOR the graph edges describe which data blocks are used [1]. The codes are described as low density because they contain fewer edges than a fully connected graph. A tornado code generates a series of cascaded bipartite graphs which connects the data nodes to check nodes .The LDPC coding scheme is repeated at each level, with the left nodes being used to calculate right parity check nodes. Decoding technique is the reverse operation of encoding, if any right node has exactly one missing left node , the missing left node can be reconstructed. Evaluation Parameters considered in the construction of a tornado code includes the number of nodes, the number of levels .In addition to single-site mass storage systems, tornado codes may also be used to increase fault tolerance in archival storage where data is replicated. A redundant array of independent disks or RAID uses multiple disks that function as one large drive, and provide for data recovery. By using RAID storage system we can maintain the given large amounts of data with high availability storage. This can be easily done by either RAID software or hardware, such as RAID controller. The RAID mechanism acts as an intermediary between the multiple disk drives and the operation system. RAID mechanism allows simultaneous read/writes to all the drives in the array and sometimes uses memory buffering to I/O requests, an overall increase in the I/O performance for read/write operations is associated with RAID technology [9]. The RAID mechanism can be configured to provide data redundancy through mirroring, which is storing two copies of the original data, or through a scheme which uses "parity drives". Parity striping stores multiple copies of data across the multiple drives, in this way the failure of one drive will not result in any loss of data [7].

The rest of the paper is organized as follows: section II describes the related work about tornado codes, graph adjustment on tornado graphs, RAID 5 storage. Section III explains the system design .Section IV provides details about proposed methodology. Section V provides experimental result about tornado encoding and decoding followed by conclusion and references.

Related Work

In LDPC coding scheme, the input is got from the user as string ,and converted to equivalent binary stream. After that generator matrix(G= $[P|I_k]$) is multiplied with generated binary stream to identify the original code word. Error is generated in original code word by changing any one of the binary bits to convert a 0 to 1 or 1 to 0.Code word is checked to identify the error bits. If any error occurs, that data bits are sent to decoder to recover the original message bits. The hard-decision decoding[3] algorithm technique can be used for decoding. The matrix defined in equation (1) is a parity check matrix 8 x 4.

$$H = \begin{pmatrix} 0\,1\,0\,1\,1\,0\,0\,1 \\ 1\,1\,1\,0\,0\,1\,0\,0 \\ 0\,0\,1\,0\,0\,1\,1\,1 \\ 1\,0\,0\,1\,1\,0\,1\,0 \end{pmatrix} \quad \ldots\ldots\ldots (1)$$

Equation 1 Parity Check Matrix

Fig 1 shows that the tanner graph formed from above parity check matrix is a bipartite graph. The two types of nodes separated in tanner graph are called variable nodes (v-nodes) and check nodes(c-nodes). Check node f_i is connected to variable node c_j if the element h_{ij} of H is a 1.

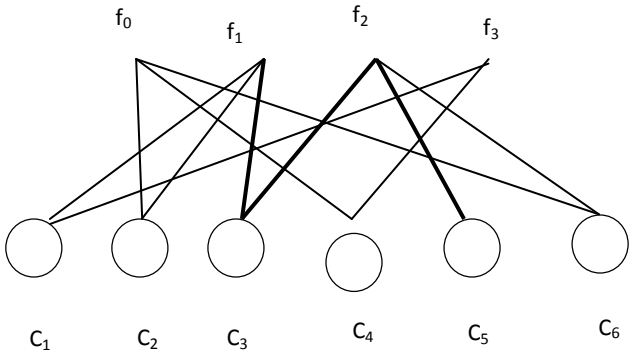

Fig 1 Tanner Graph

A LDPC code is regular if W_c is constant for every column and $W_r = W_c . (n/m)$ is also constant for every row. From the above equation 1, which is regular LDPC codes when $W_c = 2$ and $W_r = 4$. W_c is the number of 1's in the each column and W_r for each row. The regularity of these codes from the graphical representation are seen. The number of incoming edges for every variable nodes and also for all check nodes are the same. The irregular LDPC code contains number of 1's in each row or column aren't constant [6]. This is quite interesting since it denotes that constructing the well performed LDPC codes is not a hard problem. In fact , completely randomly chosen codes provide higher probability of good results. The problem is encoding complexity of such codes is usually difficult. Encoding LDPC codes done as follows: Initially select the certain number of variable nodes to place on the message bits. And in second step, the missing values of the other nodes are calculated. It involves the whole parity- check matrix and complexity would be quadratic in length. Decoding LPDC codes is done by two methods. one is hard decision problem and another one is soft decision problem. In Hard decision problem, the decoder finds the bit error by checking the parity it may be even or odd. The messages from v-nodes are transferred to c-nodes. The c-nodes checks the parity of the data stream received from v-nodes. If Check nodes receives the number of 1's which satisfies the required parity , then it sends the same data to message node if not it adjusts each received bit for satisfying the required parity and transmits the new data bits to the message node. In soft decoding problem, it enhances the performance in decoding procedure of LDPC codes. It is based on the idea of belief propagate. The sum – product algorithm is a soft decision message-passing algorithm. Apriori probability for received bits is the input probabilities which are known before running the LDPC decoder. The posterior probabilities are returned by the decoder [8,16,17].

OceanStore originally included an archival storage class using Reed-Solomon codes, but in 1999 the Typhoon project examined using Tornado Codes as a replacement coding method [11]. Recognizing that the original work on LDPC codes focused on the "collective and asymptotic" performance of the coding algorithms, Plank and Thomson performed one of the first analyses of realized codes for finite sized graphs [12]. Tornado Codes were originally described by Luby as a mechanism to perform reliable multicast [13]. Tornado codes produce an n packet encoding from a k packet source. For reconstructing the source data, it is necessary to recover £k of the n encoding packets, where £ >1 £ is the decoding inefficiency for decrease in encoding and decoding execution time[2]. Luby first presented Tornado Codes as a mechanism for reliable and efficient multicast file distribution on the Internet called digital fountain [2] .Tornado decoding algorithm can easily detect when it has received enough encoding packets to recover the original packets. Thus, a client can perform decoding in real-time as the encoding packets arrive, and reconstruct the original file as soon as it determines that sufficiently many packets have arrived. Tornado codes have encoding and decoding times proportional to $(k +£) \ln(1/(£ - 1))P$, where £ is the decoding inefficiency. Moreover, Tornado codes are simple to implement and use only exclusive-OR operation. A Tornado Code consists of a series of cascaded irregular bipartite graphs connecting data nodes to check nodes. In LDPC codes are repeated at each level , calculate the right parity check nodes from left variable nodes [4]. Reconstruction is the reverse operation of encoding technique. The original text is reconstructed from left check nodes. Evaluation Parameters involved in the construction of a Tornado Code include the number of nodes, the number of levels, and the left and right edge distributions between each level. The encoding is based on the check bits get constructed by computing the sum

(XOR) of all adjacent data bits. It takes linear time (O log n). Each check bits are considered here as Parity bits.

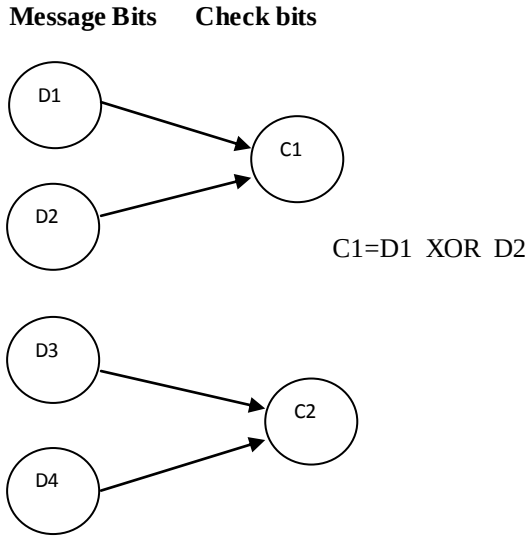

Fig 2 Tornado Encoder

In decoding**,** a missing data bit can be reconstructed if the value of the check bit and all but one adjacent data bits are known, by computing the sum (XOR) of the check bit with all known data bits.The whole decoding procedure is that repeated decoding steps eventually reconstruct all missing data bits. It takes linear time (O log n)Decoding results in linear complexity in the number of edges just as for the encoding.Nodes gaining a node degree of 0 are treated as removed as well. The following Fig.3 explained the decoding process of Tornado.

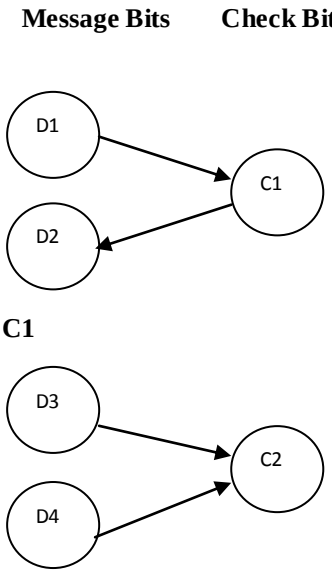

Fig 3 Tornado Decoder

Tornado coded archival storage system is an extent-based, log-structured, encoded-stripe file system(fig 4). Files are broken into extents and aggregated on fixed size stripes.when stripe is full, the entire stripe is encoded and the data and check nodes are distributed to storage devices. The foundation of the Tornado Coded stripe storage is provided by a series of Tornado Code graphs with known failure profiles [3]. This might be close to the LDPC codes, it allows several stripes to be read and written simultaneously. This tornado code archival storage system supports variable block sizes and block size may be dynamically changed at runtime[10].

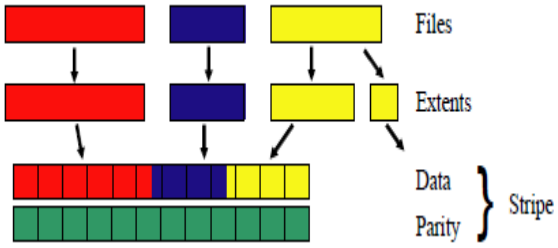

Fig 4 Tornado Coded Archival Storage system

RAID 5, data strips across multiple drives with distributed parity. The striping with distributed parity means it will split the parity information and stripe data over the multiple disks, which will have good data redundancy[15].It is able to predict the disk failures before they occur.Some implementations utilize this property to prevent data loss and reduce the Reconstruction time[13].RAID5 always has been selected for multi user and data processing environments which are characterized by high read access of data and a comparatively low need for write operations[14].

SYSTEM DESIGN

The Tornado Coded stripe storage is provided by a series of Tornado Code graphs with known failure profiles. Even though this is close to the region where LDPC codes have high overhead, it is an appropriate lower bound because it allows several stripes to be read or written simultaneously while being small enough to facilitate explicit device management[3]. The amount of data in each node (the block size) is a design choice – we chose to maintain a constant blocksize of 1 MB, so each stripe stores 48 MB of user data. To implement the tornado graph in RAID storage system, tornado codes encoding technique is used. Each data sets generates its own tornado graph which consists of data nodes which is presented in left hand side and check nodes which is presented in right hand side. It is identified that tornado codes will generate failure nodes sets because it selects the nodes in random manner. If it so there are many conflicts would happen. The chances are two checks nodes are formed from the same data nodes. If any one of data node is lost, it is difficult to reconstruct it. In this case of scenario, we have to adjust the graph to increase the fault tolerance.

To perform the adjustment, we first identify critical left nodes that were involved the most failure sets. Each failure set was influenced by a closed set of right nodes, so the target left node's right nodes are compared to the other right nodes involved in the failure set[3]. For the target left node, we find the right node with the highest failure rate and then change the connectivity of the target left node to include a different right node that was not involved in the failures. This opens the closed set that caused the failure and removes the failure set provided that the substitution did not tie one failure set to another

We are constructing RAID storage system to allocate the tornado codes by using proper encoding techniques. Every stripe consists of series of data nodes and it is stored with its own check nodes. Here the check nodes are considered as Parity bits. If any data is lost, then the system tries to reconstruct it. For the purpose of reconstruction it needs parity bit. The error correction will handled while executing the decoding process. Fig 5 shows the process of tornado encoding and decoding.

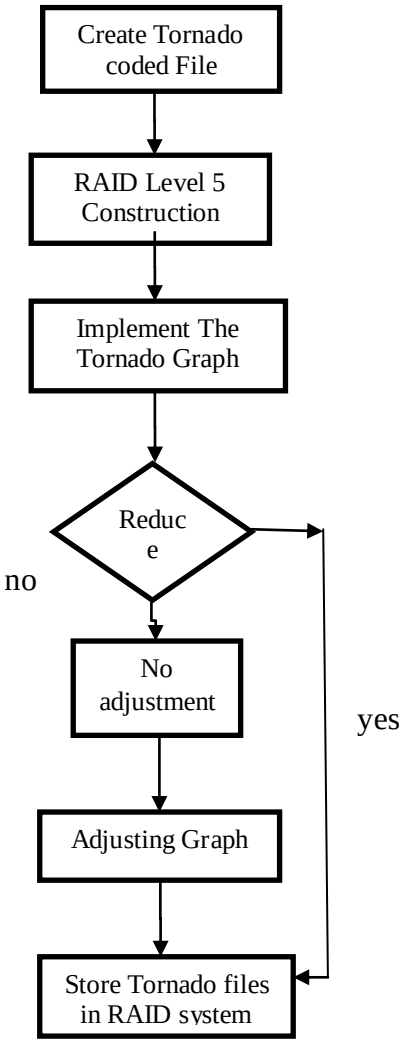

Fig 5 Simulation of tornado codes with RAID storage

$$P(fail|k\ drives) = \sum_{i=1}^{k/2} \binom{n}{k}\binom{n-1}{k-2i}\ 2^{k-2i} \Big/ \binom{2n}{k} \quad -(2)$$

Equation.2 Probability of disk Failures.

This above equation shows that probability of graph failures with corresponding k-drives. K is the number of drives. n is the number of mirrors. We can use this equation for calculating specific random graphs.

Simulated RAID system Architecture:

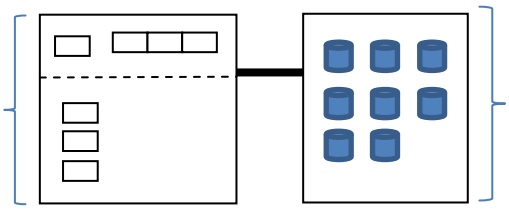

Fig 6 Organization of Tornado coded RAID simulator

The above Fig 6 shows how the data blocks are stored in the RAID Storage system. RAID Storage contains the data with parity bits. Parity bits are considered as check bits. The parity bits are distributed across all the stripes. For every data blocks generate with tornado graph with contains data bits and corresponding check bits.

PROPOSED METHODOLOGY

There is a more chance of getting error while retrieving the file from archival storage system. To reduce the error rate and to increase the efficiency without data loss, tornado codes are used. the tornado codes can be used to detect and correct multiple bit errors in an archival storage system. It is very efficient when compared to the other reliable codes. In existing system the files are encoded with tornado code graphs and stored. This tornado code graphs does not provide better fault tolerance. To recover this problem , we proposed a methodology which construct a tornado code graphs and store the tornado coded file to the RAID storage system. In this system encoding and decoding of tornado code graph is discussed. The main feature is RAID storage system would added as Archival storage system. This RAID storage system is used for high fault tolerance. We know that the RAID system is worked based on Parity bits. Result of combining both the Tornado code graphs and RAID storage provides improved performance compared with existing system. This code is executed with smaller amount of time. Based on time complexity analysis the efficiency is to be calculated during data stored on RAID storage system.

Flow chart for Tornado Encoder:

Initially, the input file is getting from the user. Convert the input text file into bit streams which is considered as data bits. From the data bits generate a check bits using XOR operation. In case of missing bits in tornado code graph, perform the adjustment for recovering the missed bits. Finally user ask to store the file in tornado coded file system. The Fig 7 tells about encoding process of tornado codes.

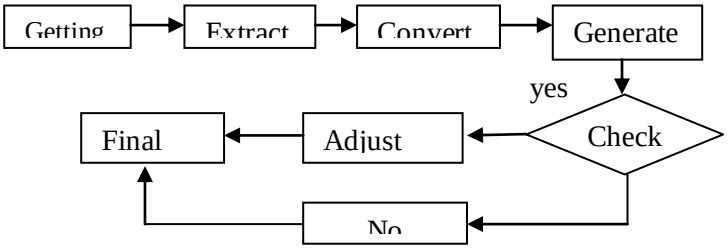

Fig 7 Tornado Encoder

Flow Chart for Tornado Decoder:

After conversion of tornado codes, the user searches for stored file in file system. Next is to generate a data bits from check bits using XOR operation as same as tornado encoder. If any error occurs in data bits , it recovers the error from check bits. Finally convert the bit streams into original file as displayed it with error free. Working of Tornado decoding is shown in below fig 8

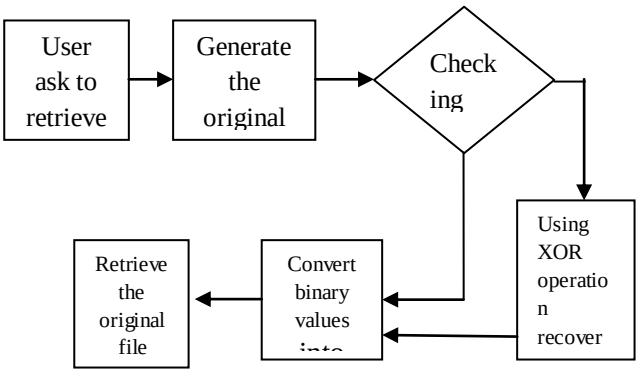

Fig 8 Tornado Decoder

Data bits with RAID:

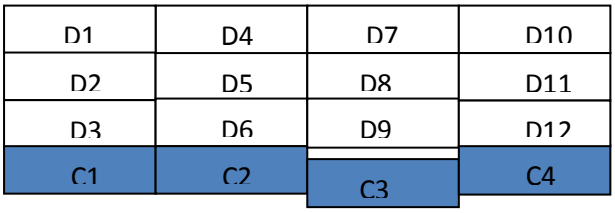

Fig 9 Simulation of Data bits and Check bits in RAID

The above figure Fig 9 shows storage of data bits in RAID with their corresponding check bits. Here the parity bits are considered as Check bits. The original bits are recovered from parity bits by using XOR operation.

EXPERIMENTAL RESULTS

The important goal for this storage system is retrieving the files without errors.Tornado codes provides the high flexibility in terms performance. Tornado codes has better error correction rate than standard convolution codes.

a) Experimental setup

Experiments are done on a cluster of about 7 systems with Pentium IV processor: Intel ® Core™2 Duo,2.54 GHz processor,1 GB RAM ,160 GB Hard Disk interconnected with Ethernet LAN cabable of transmitting at a maximum speed of 4 Mbps. Results are analyzed based on time and space complexity.

S.No	File Size	Tornado		LDPC	
		Encoding	Decoding	Encoding	Decoding
1	10KB	187msec	865msec	345msec	1658msec
2	100KB	1498msec	4239msec	9056msec	10543msec
3	1 MB	4725msec	7169msec	14954msec	16569msec
4	100 MB	7894msec	10567msec	17868msec	21995msec
5	500 MB	9984msec	12453msec	20956msec	24654msec
6	1 GB	14678msec	19567msec	28456msec	32675msec
7	3 GB	19675msec	25345msec	32465msec	38345msec

Table 1 Time Comparison of Tornado and LDPC

The above table shows that tornado encoding and decoding provides better results compared to the LDPC codes. These LDPC and Tornado codes also compared to the Reed Solomon which comes with better results. The Reed Solomon takes much time to implement but tornado is easy to implement. Compared to Reed Solomon , Tornado time complexity is 20 % faster in encoding and 26% faster in decoding

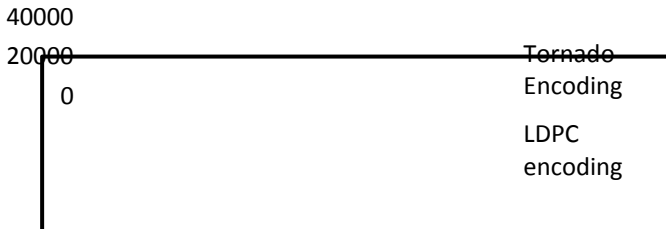

Fig 10 Time Comparison of Tornado and LDPC Encoding

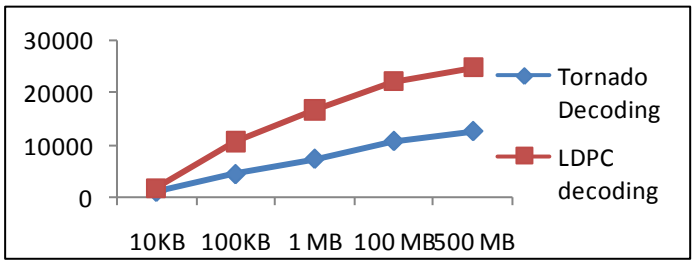

Fig 11 Time Comparison of Tornado and LDPC Decoding

The above figure shows that complexity analysis of tornado encoding and decoding process. From the above result we conclude that tornado codes are better than standard convolution codes.

Order of Complexity:

Complexity time	Tornado codes	LDPC codes
Encoding	O(n log n)	$O(n^2)$
Decoding	O(n log n)	$O(n^3)$

Table 2 Complexity Comparison of LDPC and Tornado

S.No	Error blocks	Tornado code	LDPC
1	X=2	58%	33%
2	X=4	50%	33%
3	X=6	47%	33%
4	X=8	40%	33%
5	X=10	33%	33%
6	X=12	31%	33%

Table 3 Space Savings

In case of LDPC, as the number of check nodes and message nodes increases the complexity of decoding and the computation reduces. Compared to LDPC Tornado decoding computation will be faster as values are retrieved from RAID Storage System. In case of Low Density Parity Check decoding computation will be slower as the number of updations between bit nodes and check nodes increases. Figure 12 shows the space savings of Tornado and LDPC.

1 Stripe=10blocks

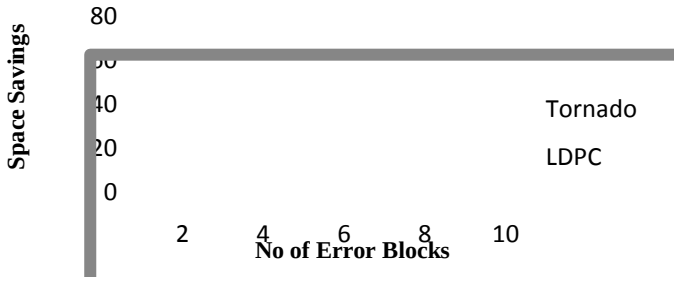

Fig 12 Space Savings of RS and LDPC

CONCLUSION

This work summarizes the important concepts regarding tornado codes and RAID storage system. It goes through the motivation of tornado codes can be encoded and decoded with RAID storage system. Tornado codes can detect and correct multiple bits of error while retrieving the original files from the storage system. As RAID Storage System uses tornado codes must avoid the data loss by using pre-compiled tested regular graphs. Compared to the default system, encoding is 26 % faster and decoding is 20 % faster.

ACKNOWLEDGEMENT

The authors convey their heartfelt thanks to Dr. R. Rudramoorthy, Principal, PSG College of Technology and Dr. R. Venkatesan, Professor and Head, Department of computer Science & Engineering, PSG College of Technology. This work is performed in the grid / cloud computing lab at PSG College of Technology.

REFERENCES

[1] Luby, M. G., Mitzenmacher, M., Shokrollahi, M.A., and Spielman, D.A., February 2001, "Efficient Erasure Correcting Codes", *IEEE T INFORM THEORY*, 47(2), pp. 569- 584,.

[2] Byers, J. W., Luby, M., and Mitzenmacher, M., 1999, "Accessing Multiple Mirror Sites in Parallel: Using Tornado Codes to Speed Up Downloads", Proceedings of IEEE INFOCOM , pp. 275-283.

[3] Woitaszek, M. and Tufo, H.M., "Fault Tolerance of Tornado Codes for Archival Storage", June 2006, Proceedings of the 15th IEEE International Symposium on High Performance Distributed Computing (HPDC 15), Paris, France.

[4] Cooley, J.A., Mineweaser, J.L., Servi, L.D. , and Tsung, E.T. , "Software-based Erasure Codes for Scalable Distributed Storage", , April 2003, Proc of the Twentieth IEEE/Eleventh NASA Goddard Conference on Mass Storage Systems &Technologies, San Diego, CA.

[5] Goodson, G. R. , Wylie, J. J. , Ganger G. R. and Reiter, M. K. , "Efficient Byzantine-Tolerant Erasure-Coded Storage", 2004, Int Conference on Dependable Systems and Networks, IEEE, pp.135-144.

[6] Bernhard M.J. Leiner, "LDPC Codes a brief Tutorial", April 8, 2005.

[7]. Chen, P.M.,. Lee, E.K., Gibson, G.A., Katz, R.H., and Patterson, D.A., "RAID High-Performance, Reliability Secondary Storage", June 1994, ACM Computing Surveys, pp. 145-185

[8] NastaranMobini, "New Iterative Decoding Algorithms for Low-Density Parity-Check (LDPC) Codes", August, 2011.

[9] Plank, J.S., "A tutorial on Reed-Solomon coding for fault-tolerance in RAID-like systems. Software – Practice& Experience", September 1997, 27(9), pp. 995–1012.

[10] Matthew Woitaszek and Henry M. Tufo ," Tornado Codes for MAID Archival Storage" ,Dec 2009

[11] Delco, M., Weatherspoon, and Zhuang, S., "Typhoon: An Archival System for Tolerating High Degrees of File Server Failure", 13 December 1999, University of California, Berkeley project report,. http://www.cs.berkeley.edu/~hweather/Typhoon/

[12] Plank, J.S. and Thomason, M.G., "On the Practical Use of LDPC Erasure Codes for Distributed Storage Applications", September 2003, Technical Report UT-CS-03-510, University of Tennessee.

[13] Manish Malhotra and Kishor S.Trivedi,"Reliability analysis of Redundant Arrays of inexpensive Disks", 1993, journal of parallel and distributed computing pp. 146-151.

[14] [online] http://www.recover-raid.com/RAID_graphical-look.html

[15] [online] http://www.tecmint.com/create-raid-5-in-linux/

[16] Namrata P Bhavsar, Prof.Brijesh Vala, Ankit Pancholi "FPGA based Implementation of Decoding Algorithms of LDPC", May 2014 , International Journal of Engineering Trends and Technology (IJETT) – Volume 11 Number 8.

[17] Namrata P. Bhavsar , Brijesh Vala, "Design of Hard and Soft Decision Decoding Algorithms of LDPC", March 2014, International Journal of Computer Applications (0975 – 8887) Volume 90 – No 16.