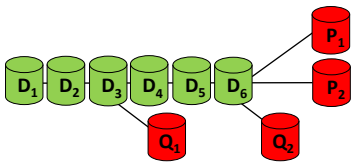

Fig. 1. A (6, 2, 2) Azure local reconstruction code.

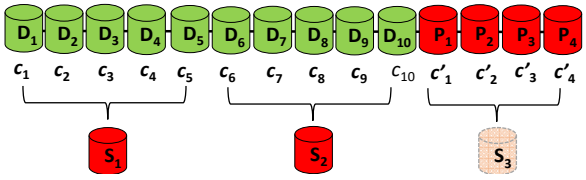

Fig. 2. A HDFS-XORBAS locally repairable code.

- Any reasonably sized bundle consisting of two or more single-parity RAID arrays will tolerate all triple disk failures and at least 96 percent of all quadruple failures.
- Any reasonably sized bundle consisting of two or more RAID level 6 arrays will tolerate all quintuple disk failures and at least 99.9 percent of all sextuple failures.

The remainder of the paper is structured as follows. Section II describes extant locally repairable codes. Section III introduces our proposal and Section IV discusses its main features. Finally, Section V has our conclusions.

II. Previous Work

In this section, we review the most significant previous work on locally repairable codes. Space considerations prevented us from being more complete.

A. Windows Azure Local Reconstruction Codes

Fig. 1 shows a (6, 2, 2) Azure Local Reconstruction Code (LRC) [6]. As we can see, the code partitions its six data blocks into two sets of three blocks with each set having its own local parity block, namely parity blocks Q_1 and Q_2. In addition, the code computes the two global parity blocks P_1 and P_2. More generally, a (k, l, r) code will partition its k data blocks into l sets of k/l blocks each and will have l local parity blocks and r global parity blocks. The code will be able to recover from the loss of either any single data block or any single local parity block using exactly k/l block reads. At the same time, the recovery performance of the code depends on the coefficients selected for the linear expressions defining the $l + r$ parity blocks. These coefficients must be specifically chosen to ensure that the code will be able to decode all the information theoretically decodable failure patterns. For a (6, 2, 2) LRC, this means all triple failures and 86 percent of all quadruple failures.

Overall, the (6, 2, 2) LRC ensures that all single data block failures can be resolved using three read operations and has a space overhead of 4/10 = 40 percent.

B. HDFS-XORBAS locally repairable codes

As we can see in Fig. 2, the HDFS-XORBAS locally repairable code [13] comprises 10 data blocks and 7 parity blocks. The four global parity blocks P_1,P_2, P_3, and $P_4$are built with a standard Reed-Solomon code and ensure that the code can tolerate the loss of four arbitrary blocks. Parity blocks S_1, S_2, and S_3 are local parity blocks aimed at reducing the cost of recovering from single block failures. Block S_1 is a linear combination $S = c_1D_1 \oplus c_2D_2 \oplus c_3D_3 \oplus c_4D_4 \oplus c_5D_5$ of the contents of data blocks D_1 to D_5 and block S_2 is similarly obtained from data blocks D_6 to D_{10}. Block S_3 is an *implied* parity block. It is not stored but can be created on demand as long as the coefficients of the linear expressions defining blocks S_1 and S_2 satisfy the relation $S_1 + S_2 + S_3 = 0$. In addition, the coefficients $c_i$are subjected to further optimization to maximize the fault tolerance of the code.

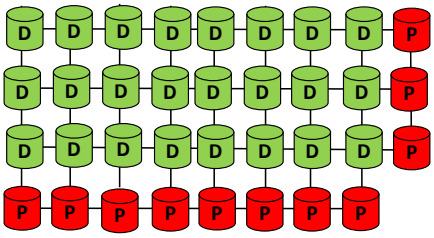

Fig. 3. A two-dimensional RAID array.

Overall, HDFS-XORBAS ensures that all single block failures can be resolved using five read operations and protects its contents against all quadruple disk failures. Its space overhead is 6/16 = 37.5 percent.

Comparing the (6, 2, 2) Azure LRC with HDFS XORBAS, we can see they make very different choices regarding the three-way tradeoff [8] among space efficiency, durability, and recovery efficiency. The (6, 2, 2) Azure LRC only requires three block reads to recover from a single data block failure but cannot tolerate all quadruple block failures. Conversely, HDFS XORBAS tolerates all quadruple block failures and has a somewhat smaller space overhead than the (6, 2, 2) Azure LRC (37.5 percent instead of 40 percent), but requires five read operations instead of three to recover from a single block failure.

C. Rotated Reed-Solomon codes

Khan et al. [7] investigated some of the most popular erasure codes and proposed a new class of codes that perform degraded reads more efficiently than all known codes, but otherwise keep the reliability and performance properties of extant Reed-Solomon codes. The emphasis of their work was on minimizing overall data transfers rather than minimizing the number of disks involved in the reconstruction.

D. Shingled Erasure Codes (SHEC)

Miyamae et al. [8] have proposed a disk array organization comprising k data disks and m parity disks. Each of these m parity disks contains the XOR of the contents of l data disks, which are said to form a *locality*. As $ml > k$, it is possible to assign each of the k data disks to exactly ml/k distinct localities in a way that ensures that the array will tolerate the simultaneous failure of up to ml/k disks. The organization is referred to as a shingled erasure code (SHEC) because localities overlap with each other like the tiles of a roof.

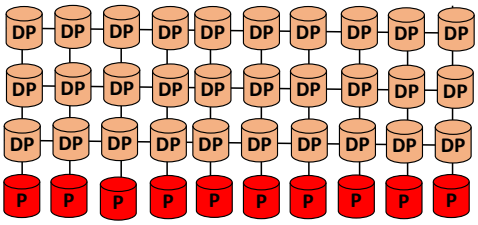

Fig. 8. A bundle consisting of three RAID level 6 arrays with ten disk each and ten additional column parity disks.

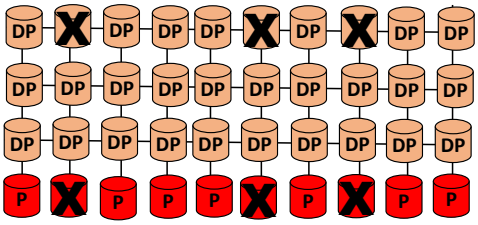

Fig. 9. A fatal sextuple failure.

Parity information will occupy the equivalent of one disk per RAID level 5 array plus $n + 1$ column parity disks. As a result, the parity space overhead of the bundle will be equal to:

$$\frac{m + n + 1}{(m + 1)(n + 1)}.$$

The sole possible fatal quadruple failures involve the failures of four disks that share the same two rows and the same two columns. Fig. 7 shows one of these fatal quadruple failures. As we can see, the four failed disks share the same two rows and the same two columns. Enumerating the set of fatal quadruple failures is thus equivalent to enumerating the number of rectangles that we can form by selecting two arbitrary rows out of $m + 1$ and two arbitrary columns out of $n + 1$, which is equal to $\binom{m+1}{2}\binom{n+1}{2}$. Observing there are $\binom{(m+1)(n+1)}{4}$ possible quadruple failures, the probability that an arbitrary quadruple disk failure will result in a data loss is:

$$\frac{\binom{m+1}{2}\binom{n+1}{2}}{\binom{(m+1)(n+1)}{4}}.$$

Table I displays some selected bundle configurations with their space overheads and the percentage of quadruple disk failures that will result in a data loss. We want to keep the number m of RAID arrays per bundle small in order to involve as few disks as possible in the recovery of single disk failures. At the same time, we want to use longer RAID stripes to keep the parity space overhead under 50 percent.

Comparing the performance of our solution with that of a (6, 2, 2) Azure locally repairable code is difficult because our solution tends to group together many more data disks. The closer we can get is comparing the (6, 2, 2) Azure code with a

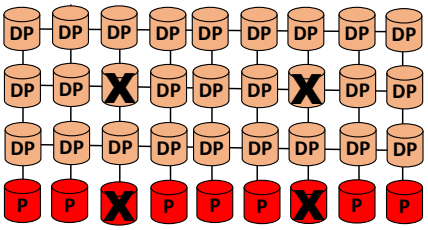

Fig.7. A fatal quadruple failure.

TABLE I. SELECTED RAID LEVEL 5 BUNDLE SIZES AND THEIR RESPECTIVE SPACE OVERHEADS.

Number of RAID stripes	Disks per RAID stripe	Storage capacity (disks)	Space overhead	Fatal quadruple failures
2	8	14	41.7%	0.791%
2	10	18	40.0%	0.493%
2	12	22	38.9%	0.336%
3	8	21	34.4%	0.467%
3	10	27	32.5%	0.295%
3	12	33	31.3%	0.204%
4	8	28	30.0%	0.306%
4	10	36	28.0%	0.195%
4	12	44	26.7%	0.135%
5	8	35	27.1%	0.216%
5	10	45	25.0%	0.138%
5	12	55	23.6%	0.096%

bundle of two RAID level 5 arrays with 4 disks each. Both organizations would comprise six data disks. Our organization would have six parity disks, bringing its space overhead to 50 percent instead of 40 percent for the (6, 2, 2) code. At the same time, our organization would be able to resolve all single disk failures by accessing two disks instead of three and would tolerate 96 percent of quadruple disk failures instead of 86 percent for the (6, 2, 2) code.

Another option would be to bundle together three RAID level 5 arrays with 5 disks each. The organization would have 12 data disks and 8 parity disks, giving it the same space overhead as the (6, 2, 2) code. It would require accessing three disks to resolve all single disk failures and would tolerate 98.8 percent of quadruple disk failures, which is better than the (6, 2, 2) code.

B. Bundles of RAID level 6 arrays

Whenever a higher level of data resiliency is required, we can replace the RAID level 5 arrays used in our scheme by RAID level 6 arrays [2] [14], that is, RAID arrays that tolerate two disk failures.

Fig. 8 represents one such bundle. It consists of three RAID level 6 arrays with ten disks each plus an additional row of ten disks that contain the column parities. Each of the disks in the two RAID level 6 arrays contains both data blocks and parity blocks with data blocks occupying 8/10 of the disk capacity and parity blocks the remaining 2/10. Similarly, the ten disks in the bottom row contain a mixture of parity and superparity blocks.

quadruple disk failures while bundles of RAID level 6 arrays could similarly recover from all quintuple and at least 99.9 percent of sextuple disk failures.

More work is still needed to evaluate the cost of repairing double and triple failures and estimating the impact of irrecoverable read errors on the repair process.

REFERENCES

[1] B. Beach, "BackBlaze open sources Reed-Solomon erasure coding source code," https://www.backblaze.com/blog/reed-solomon/, June 16, 2015, retrieved June 20, 2018.

[2] W. A. Burkhard and J. Menon, "Disk Array Storage System Reliability," Proc. 23rd Int. Symp. on Fault-Tolerant Computing (FTCS-23), pp. 432–441, June 1993.

[3] B. Calder, J. Wang, A. Ogus, N. Nilakantan, A. +, 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. Fahim ul Haq, M. Ikram ul 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," Proc. 23rd ACM Symposium on Operating Systems Principles (SOSP '11), Cascais, Portugal, Oct. 2011.

[4] S. Ghemawat , H. Gobioff ,and S.-T. Leung, "The Google file system," Proc.19th ACM Symp. on Operating Systems Principles (SOSP '03), Bolton Landing, NY, Oct. 2003.

[5] C. Huang and L. Xu, STAR: An efficient coding scheme for correcting triple storage node failures. IEEE Transactions on Computers, Vol. 57, No. 7, pp.889–901, July 2008.

[6] C. Huang, H. Simitci, Y. Xu, A. Ogus, B. Calder, P. Gopalan, J. Li, and S,Yekhanin, "Erasure coding in Windows Azure storage," Proc. 2012 USENIX Annual Technical Conf. (USENIX ATC 12), Boston, MA, pp. 15-26, 2012.

[7] O. Khan, R. Burns, J. Plank, W. Pierce, and C. Huang, "Rethinking erasure codes for Cloud file systems: minimizing I/O for recovery and degraded reads," Proc 10th USENIX Conf. on File and Storage Technologies (FAST '12), San Jose, CA, Feb. 2012.

[8] T. Miyamae, T. Nakao, and K. Shiozawa, "Erasure code with shingled local parity groups for efficient recovery from multiple disk failures," Proc. 10th USENIX Workshop on Hot Topics in System Dependability (HotDep '14) Broomfield, CO, Oct. 2014.

[9]. J.-F. Pâris, T. Schwarz, S. J., A. Amer and D. D. E. Long, "Highly Reliable Two-Dimensional RAID Arrays for Archival Storage," Proc, 31st Int. Performance of Computers and Communication Conf (IPCCC 2012*)*, Austin, TX, pp. 324–331, Dec. 2012.

[10] D. A. Patterson, G. Gibson, and R. H. Katz, "A case for redundant arrays of inexpensive disks (RAID)," Proc. 1988 ACM SIGMOD International Conf. on Management of Data, Chicago, IL, pp. 109–116, June 1988.

[11] K. V. Rashmi, N. B. Shah, D. Gu, H. Kuang, D. Borthakur, K. Ramchandran, "A Hitchhiker's Guide to Fast and Efficient Data Reconstruction in Erasure-coded Data Centers," Proc. 2014 ACM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM), Chicago, IL, Aug. 2014.

[12] I. S. Reed and G. Solomon, "Polynomial Codes over Certain Finite Fields", Journal of the Society for Industrial and Applied Mathematics (SIAM), 8 (2): 300–304, 1960.

[13] M. Sathiamoorthy, M. Asteris, D. Papailiopoulos, A. G. Dimakis, R. Vadali, S. Chen, and D. Borthakur, "XORing elephants: novel erasure codes for big data," Proc. of the VLDB Endowment, Vol. 6, No. 5, pp. 325-336, 2013.

[14] T. Schwarz, S. J., Reliability and Performance of Disk Arrays, PhD Dissertation, Department of Computer Science and Engineering, University of California, San Diego, 1994.

[15] T. Schwarz, S. J., A. Amer, J.-F. Pâris, "Combining low IO-operations during data recovery with low parity overhead in two-failure tolerant archival storage systems," Proc. 21st IEEE Pacific Rim International Symp. on Dependable Computing (PRDC '15), Zhangjiajie, China, Nov. 2015.

[16] A. Wildani, T. J. E. Schwarz, E. L. Miller and D. D. E. Long, "Protecting against rare event failures in archival systems," Proc. 17th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS '09), London, GB, pp. 246–256, Sep. 2009.