Optimizing Apportionment of Redundancies in Hierarchical RAID

Alexander Thomasian
Thomasian & Associates
Pleasantville, NY 10570 USA
alexthomasian@gmail.com

Abstract

Large disk arrays are organized into storage nodes - SNs or bricks with their own cashed RAID controller for multiple disks. Erasure coding at SN level is attained via parity or Reed-Solomon codes. Hierarchical RAID - HRAID provides an additional level of coding across SNs, e.g., check strips P, Q at intra-SN level and R at the inter-SN level. Spare disks and SNs are not provided and rebuild is accomplished by restriping, e.g., overwriting P and Q for disk failures and R for an SN failure. For a given total redundancy level we use an approximate reliability analysis method and Monte-Carlo simulation to explore the better apportionment of check blocks for intra- vs inter-SN redundancy. Our study indicates that a higher MTTDL - Mean-Time-to-Data-Loss is attained by associating higher reliability at intra-SN level rather than inter-SN level, which is contrary to that of an earlier study.

Keywords Redundant Array of Independent Disks - RAID, erasure coding, hierarchical RAID, storage nodes - SNs, intra-SN coding, inter-SN coding, approximate reliability analysis, rebuild via restriping, Monte Carlo simulation, Mean Time To Data Loss - MTTDL.

1 Introduction to Hierarchical RAID - HRAID

The five level classification of *Redundant Array of Independent Disks - RAID* introduced in [4] was extended by RAID0 and RAID6 in [1]. In this study we consider redundancy methods based on erasure coding, rather than mirroring, which double disk space requirements. RAID$(4+\ell), \ell \geq 1 >$ which is based on erasure coding utilizes the capacity equivalent of ℓ check disks to tolerate as many disk failures with *Maximum Distance Separable - MDS* codes [3].

Striping balances disk loads by partitioning data files into fixed size strips, which are placed round-robin across the disks. Strips in a row constitute a stripe. One strip per stripe is dedicated to parity in RAID5, two check strips in RAID6, Check strip update loads are balanced by placing them in right to left diagonals [1].

Node 1				Node 2				Node 3				Node 4			
$D^1_{1,1}$	$D^1_{1,2}$	$P^1_{1,3}$	$Q^1_{1,4}$	$D^2_{1,1}$	$P^2_{1,2}$	$Q^2_{1,3}$	$D^2_{1,4}$	$P^3_{1,1}$	$Q^3_{1,2}$	$D^3_{1,3}$	$D^3_{1,4}$	$Q^4_{1,1}$	$D^4_{1,2}$	$D^4_{1,3}$	$P^4_{1,4}$
$D^1_{2,1}$	$P^1_{2,2}$	$Q^1_{2,3}$	$D^1_{2,4}$	$P^2_{2,1}$	$Q^2_{2,2}$	$D^2_{2,3}$	$D^2_{2,4}$	$Q^3_{2,1}$	$D^3_{2,2}$	$D^3_{2,3}$	$P^3_{2,4}$	$D^4_{2,1}$	$D^4_{2,2}$	$P^4_{2,3}$	$Q^4_{2,4}$
$P^1_{3,1}$	$Q^1_{3,2}$	$D^1_{3,3}$	$D^1_{3,4}$	$Q^2_{3,1}$	$D^2_{3,2}$	$D^2_{3,3}$	$P^2_{3,4}$	$D^3_{3,1}$	$D^3_{3,2}$	$P^3_{3,3}$	$Q^3_{3,4}$	$D^4_{3,1}$	$P^4_{3,2}$	$Q^4_{3,3}$	$D^4_{3,4}$
$Q^1_{4,1}$	$D^1_{4,2}$	$D^1_{4,3}$	$P^1_{4,4}$	$D^2_{4,1}$	$D^2_{4,2}$	$P^2_{4,3}$	$Q^2_{4,4}$	$D^3_{4,1}$	$P^3_{4,2}$	$Q^3_{4,3}$	$D^3_{4,4}$	$P^4_{4,1}$	$Q^4_{4,2}$	$D^4_{4,3}$	$D^4_{4,4}$

Figure 1: A 4×4 HRAID1/1 with $N = 4$ nodes and $M = 4$ disks per node. Only the first $M = 4$ stripes are shown in the figure.

RAID performance is a primary issue in *OnLine Transaction Processing - OLTP*, which generates accesses to small randomly placed disk blocks. A high positioning overhead (seek time plus latency) is incurred, while the transfer time for small blocks is negligible. Given d^{new} we need to read d^{old}, unless it is cached, compute $d^{diff} = d^{old} \oplus d^{new}$ read p^{old} to compute $p^{new} = p^{old} \oplus d^{diff}$. Unless d^{old} and p_{old} are cached four disk accesses are required to update a single data block, which is known as small write penalty [1]. Given that the mean disk access for reads and writes is $\bar{x}_d$ and the fraction of read and (logical) write requests is f_r and $f_w = 1 - f_r$, then the average cost per disk access is $(f_r + 2f_w(\ell + 1))x_d$.

Hierarchical RAID - HRAID was a proposal to apply the RAID paradigm at two levels [7]. IBM's Icecube is a similar proposal [12], which led to a prototype at a startup, but not a product. We consider an HRAID with N *Storage Nodes - SNs* with M disks per SN. Each SN is a RAID0 or RAID$(4 + \ell), 1 \leq \ell 3$ storage system with ℓ check strips per stripe. Each SN is an ℓDFT $0 \leq \ell \leq 3$, which can mask the failure of ℓ out of M disks per SN. HRAID(k/ℓ) extends the RAID paradigm to mask the failures k out of N SNs. Up to k SN failures can be tolerated by providing $0 \leq k \leq 3$ strips per stripe at each SN. The average cost per disk access is then $x_{avg} = [f_r + 2f_w(k + 1)(\ell + 1)]x_d$, but there is the cost of transmitting d^{diff}.

2 Intra- & Internode Coding in HRAID

There are N SNs and M disks per SN. Internode (resp. intranode) data protection is achieved via internode (resp. intranode) check strips. Intranode check strips are computed over all strips at an SN, including internode check strips. Internode check strips are computed over data strips at the same position at the remaining SNs, but not intranode check strips. In HRAIDk/ℓ the coding across the N SNs is k*Node Failure Tolerant - NFT*, while the coding at each SN with M disks is ℓ*Disk Failure Tolerant - DFT*.

The HRAIDk/ℓ data layout with $N = M$ dedicates $k + \ell$ check strips per M strips on an SN. Check strips follow a left symmetric layouts as in RAID5 [1]. Strips are shifted from SN to SN and on a per row basis at each SN. The redundancy level is given as $(k + \ell)/M$, While MDS - Maximum Distance Separable codes are used at both levels [8], the overall code is not MDS. Given that HRAIDk/ℓ tolerates k SN failures and ℓ disk failures per SN, the maximum number of disk failures that can be tolerated is:

$$d_{max} = k \times M + (N - k) \times \ell = N(k + \ell) - k\ell, \text{ where } .M = N$$

Given that there are $N(k+\ell)$ check disks even in the best case fewer disk failures can be tolerated, so that the overall code is not MDS. It should be emphasized that MDS provides protection for all disk failure configurations, while this is a best case scenario.

Inter-SN check blocks allow the recovery of missing blocks if intra-SN recovery fails. This results in a significant improvement in *Mean Time To Data Loss - MTTDL* with respect to HRAID0/ℓ with no inter-SN check blocks.

When a disk fails performance improvements is attained via restriping, i.e., overwriting check strips with data strips or the data on a whole SN. Starting with RAID7 with check blocks P, Q, R restriping which overwrites the blocks in the order R, Q, P results in RAID7 $\rightarrow$ RAID6 $\rightarrow$ RAID5 $\rightarrow$ RAID0
An SN failure occurs due to its controller failures or when the number of its failed disks exceeds ℓ.

3 Shortcut Reliability Analysis of HRAID

Let $r = 1-\epsilon$ denote the reliability of each disk, where $\epsilon \ll 1$ is the disk unreliability. Let R_ℓ denote the reliability of RAID$(4+\ell)$ with R_0 for RAID0, e.g., in the case of RAID5 $\ell = 1$ which can tolerate one disk failure [6]:

$$R_1 = r^M + N(1-r)r^{M-1} = (1-\epsilon)^M + M\epsilon(1-\epsilon)^{M-1} \approx 1 - \binom{M}{2}\epsilon^2 + 2\binom{M}{3}\epsilon^3 - \dots$$

It is shown in [6] that the smallest power n in ϵ^n of the polynomial determines the minimum number of disk failures leading to data loss, which is two in this case. In other words RAID5 can tolerate a single disk failure according to the MDS criterion. The approximate reliability equation for RAID$(\ell+4), \ell \geq 1$ obtained by induction is:

$$R_\ell \approx 1 - \binom{N}{\ell+1}\epsilon^{\ell+1} + (\ell+1)\binom{N}{\ell+2}\epsilon^{\ell+2} - \dots$$

Given that $R_{k/\ell}(N,M)$ denote the reliability of HRAIDk/ℓ with N nodes and M disks as affected by disk failures only, the reliability of HRAID1/0(N,M) and HRAID0/1(N,M) can be expressed by first noting that $R_1 = r^M + M(1-r)r^{M-1}$ and $R_0 = r^M$:

$$R_{0/1}(N,M) = (R_1)^N \approx 1 - \frac{NM(M-1)}{2}\epsilon^2.$$

$$R_{1/0}(N,M) = (R_0)^N + N(1-R_0)(R_0)^{N-1} \approx 1 - \frac{N(N-1)M^2}{2}\epsilon^2.$$

It follows that $d_{min} = (k+1)(\ell+1) = 2$ disk failures may lead to data loss in both cases, but $R_{0/1} > R_{1/0}$.

It follows from the expressions for approximate reliability for HRAID1/2 and HRAID2/1 that both can fail with $(k+1)(\ell+1)=6$ disk failures.

$$R_{1/2}(N,M) = R_2^N + N(1-R_2)R_2^{N-1}$$
$$\approx 1 - \frac{N(N-1)M^2(M-1)^2(M-2)^2}{72}\epsilon^6 + \dots$$
$$R_{2/1}(N,M) = R_1^N + N(1-R_1)R_1^{N-1} + \binom{N}{2}(1-R_1)^2R_1^{N-2}$$
$$\approx 1 - \frac{N(N-1)(N-2)M^3(M-1)^3}{24}\epsilon^6 + \dots$$

and that HRAID1/2 is more reliable than HRAID2/1, since $R_{1/2}(N,M) > R_{2/1}(N,M)$ implies $N > 2 + (M-2)^2/(3M(M-1))$, which is true for reasonable values of N.

To determine the probability that HRAID1/2 encounters data loss with the sixth disk failure, consider the configuration with one failed node due to three failed disks and another node with two failed disks, The probability of a disk failure at this node is

$$p_{1/2} = (M-2)/D_S \text{ with } D_S = (N-2)M + M - 2.$$

In the case of HRAID2/1 array consider two nodes with two failed disks each, so that their data can only be reconstructed via the internode check code. Data loss occurs when a third node, which already has a failed disk, encounters a second disk failure. The probability of this event is $p_{2/1} = (M-1)/D_S$. The inequality $p_{1,2} < p_{2,1}$ leads to $N+M>4$, which is always true.

Monte Carlo simulation was used in [9] to determine the MTTDL. Based on the data provided in [2] the time to disk failure is assumed to be exponentially distributed [11] which can be specified by a single parameter: *Mean Time To Failure - MTTF* million hours or equivalently failure rate $\delta = 10^{-6}$ It follows that the total failure rate is the sum of the failure rates of surviving components . A more detailed discussion of reliability analysis appears in [10]. The assumption that controllers do not fail (failure rate $\gamma = 0$) leads to Table 4 in [9].

Table 1: HRAIDK/ℓ MTTDL in thousands of hours for $N = M = 12$ and disk MTTF=10^6 hours.

	$k=0$	$k=1$	$k=2$	$k=3$
$\ell=0$	6.9	14.6	23	32
$\ell=1$	36.9	58.9	78.4	97.7
$\ell=2$	118.9	118.8	148.7	176.8
$\ell=3$	139.6	191.5	231.8	268.1

Simulation results ptrovided in [9] for controller failure rates equalling or exceeding those of disks $\gamma = 1,2,3\times 10^{-6}$. show that higher redundancy at lower level is preferable, but to a lesser degree. For the specific target chosen for Icecube in [5] HRAID2/1 or HRAID3/0 meet the reliability requirement. The hierarchical reliability analysis method developed in [5] is approximate and require validation, via simulation similar to the one used in [9].

References

[1] P. M. Chen, E. K. Lee, G. A. Gibson, R. H. Katz, D. A. Patterson: RAID: High-performance, reliable secondary storage. ACM Computing Surveys 26(2): 145-185 (1994).

[2] G. A. Gibson: Redundant Disk Arrays: Reliable, Parallel Secondary Storage. MIT Press, 1992

[3] F. J. MacWilliams, N.J. Sloane. The Theory of Error-Correcting Codes. North-Holland 1977.

[4] D. A. Patterson, G. A. Gibson, R. H. Katz: A case for redundant arrays of inexpensive disks (RAID). Proc. ACM SIGMOD Conf. on Management of Data 1988: 109-116

[5] K. K. Rao, J. L. Hafner, R. A. Golding: Reliability for networked storage nodes. IEEE Trans. Dependable Secure Computing 8(3): 404-418 (2011)

[6] A. Thomasian: Shortcut method for reliability comparisons in RAID. *J. Systems Software 79*(11): 1599-1605 (2006).

[7] A. Thomasian: Multi-level RAID for very large disk arrays. ACM SIGMETRICS Performance Evaluation Review 33(4): 17-22 (2006).

[8] A. Thomasian, M. Blaum: Higher reliability redundant disk arrays: Organization, operation, and coding. ACM Trans. Storage 5(3): 7:1-7:59 (2009)

[9] A. Thomasian, Y. Tang, Y. Hu: Hierarchical RAID: Design, performance, reliability, and recovery. J. Parallel Distributed Computing 72(12): 1753-1769 (2012).

[10] A. Thomasian: Storage Systems: Organization, Performance, Coding, Reliability, and Their Data Processing, Morgan Kaufmann / Elsevier 2021,

[11] K. S. Trivedi. Probability and Statistics with Reliability, Queueing, and Computer Science Applications, 2nd Ed. Wiley 2001.

[12] Winfried W. Wilcke et al. IBM Intelligent Bricks project - Petabytes and beyond. IBM J. Research & Development 50(2-3): 181-198 (2006)

Alexander Thomasian (IEEE Life Fellow) has a Ph.D. degree in Computer Science from University of California, Los Angeles. He served as faculty at Case Western University, University of Southern California, University of Connecticut, and New Jersey Institute of Technology. He was a research staff member at Burroughs Corp. and IBM T.J.Watson Research Center. He held the visiting position at Shenzhen Institute of Advanced Technology and he was a Fulbright Fellow at AUA.am. He is the president of Thomasian & Associates Consulting. He takes the lead in performance analysis of database concurrency control, and indexing in high dimensional spaces storage research at IBM and NJIT with funding from NSF, Hitachi, and AT&T. He is the area editor of IEEE Transactions on Parallel and Distributed Systems and he is the PC member of numerous conferences. He has published over 150 papers, and four patents. He published Storage Systems: Organization, Performance, Coding, Reliability, and Their Data Processing by Elsevier in 2021.