RESAR: Reliable Storage at Exabyte Scale Reconsidered

Thomas Schwarz, SJ*, Ahmed Amer†, John Rose‡
*Marquette University, Milwaukee, WI, thomas.schwarz@marquette.edu
†Santa Clara University, Santa Clara, CA, aamer@scu.edu
‡Xavier Institute of Engineering, Mumbai, India, johnrose@xavierengg.com

***Abstract*—Stored data needs to be protected against device failure and irrecoverable sector read errors, yet doing so at exabyte scale can be challenging because of the large number of failures that must be handled. We have developed and presented RESAR (Robust, Efficient, Scalable, Autonomous, Reliable) storage, an approach to storage system redundancy that only uses XOR-based parity and employs a graph to lay out data and parity [5]. Here we add to our prior work by giving a proof of our analytic results for RESAR reliability. The results presented here are valid for any size RESAR layouts and allow us to make an exact comparison of spiral RESAR robustness with an equivalent RAID 6 disk array.**

I. Introduction

Very large storage systems contain millions of storage devices, typically hard disks for cost reasons. At this scale, device failure and discovery of latent sector errors becomes frequent events, against which the system protects its data by storing them redundantly. While occasionally higher levels of redundancy are desirable, the consensus in the storage community is that the system needs to tolerate two simultaneous failures without losing data. This small number is achieved by declustering where the space on a disk is divided into chunks, called *disklets*. A group of k disklets (typically $k = 8$, $k = 16$, or $k = 32$) forms a reliability stripe to which the standard declustered RAID Level 6 architecture adds two parity disklets. The contents of these parity disklets are calculated with an erasure correcting code based on Galois field calculations. With a disklet size of 1GB, the complete disklet can be read in about five seconds. If the system detects a failure, the contents of all disklets in the failed disk are reconstructed by reading k of the remaining $k+1$ disklets in the stripe (or reads $k/(k+1)$ of the remaining $k+1$ disklets for faster reads) and storing the result in a spare disklet. If a system discovers a disklet with a latent sector error, it is recovered in the same way.

Recently, we presented a new layout, called RESAR for Robust, Efficient, Scalable, Autonomous, and Reliable storage [5]. Unlike RAID 6 that places each data disklet in a reliability stripe with two additional parity disklets, RESAR places each data disklets in two reliability stripes. The parity in each reliability stripe is calculated using normal bitwise exclusive-or operations instead of the more involved Galois field calculations. Previously, we used simulation to compare the reliability of a RESAR layout with that of the typical declustered RAID 6 architecture described above. The previous work gave the probability of data loss after loss of a few disklets but did not provide a proof. Here, we give the proof and further use the analytic results to show that the robustness (probability of incurring dataloss after suffering f disklet failures) of RESAR is $\frac{1}{6}(k^2+3k+2)$ times better than that of an equivalent RAID 6 for very large systems.

In the remainder of this article, we explain the spiral RESAR layout (Section II) and the recovery operations from disklet failures (Section III). We then derive the dataloss probability given f failed disklets for $f \leq 5$ (Section IV). After determining analytically the probability of dataloss for a RAID 6 layout given f disklet failures for any number f (Section V), we use these exact results to compare the robustness of the two architectures as its size goes to infinity (Section VI).

II. Spiral RESAR Layout

If data disklets represents information symbols and parity disklets represent parity symbols, then a RESAR layout is an error correcting code, namely a flat XOR-code. Greenan *et al.* defined them as codes where parity symbols are calculated from certain subsets of data symbols with an exclusive-or operation [2]. We call these subsets *reliability stripes*.

Layouts based on flat XOR-codes that tolerate two simultaneous failures need to place each data disk in at least two different reliability stripes. The intersection of two reliability stripes cannot contain more than a single disklet. We can *label* each data disk by the numbers of the two reliability stripes to which it belongs. Similarly, we can label each parity disk with the number of the reliability stripe to which it belongs. This defines a graph structure derived from the disk layout where each parity disk corresponds to a vertex and each data disk to an edge between vertices.

We use a graph to visualize the layout of disklets into reliability stripes and later manipulate them. Fig. 1 gives an example. In the graph layout, parity disklets corresponds to vertices and data disklets to edges. For example, disklet 9 on the left belongs to a horizontal reliability stripe with parity disklet P2 and a vertical reliability stripe with parity disklet D1. In the graph visualization (Fig. 1), disklet 9 is represented by an edge (labelled with its number). This edge is adjacent to two vertices, that represent the parity disklets, P2 and D1, respectively.

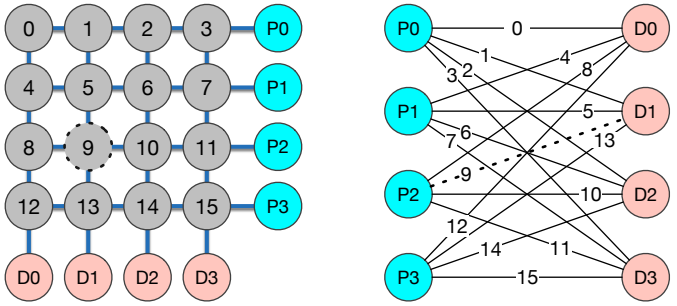


Fig. 1. Left: The two-dimensional layout for a disk array. 0, 1, 2, . . ., 15 are data disks and $P0$, . . ., $P3$ and $D0$, . . ., $D3$ the two set of parity disks. Right: The corresponding graph visualization, where edges correspond to data disks. For example, data disk 9 located in the stripes with parities $P2$ and $D1$ corresponds to the edge between $D1$ and $P2$ on the right.

To our knowledge, the graph visualization was first exploited by Xu *et al.* in the definition of B-codes [9]. B-codes are a precursor to Row-Diagonal parity codes [1] that only use exclusive-OR operations for parity calculation and guarantees two failure tolerance. An observation by Zhou *et al.* characterizes minimal failure sets of disks in a 2-failure tolerant flat XOR-code as those containing either a cycle of edges or a path where the end vertices have also failed [10]. The graph visualization is a good way to determine the failure resilience of these type of layouts [4].

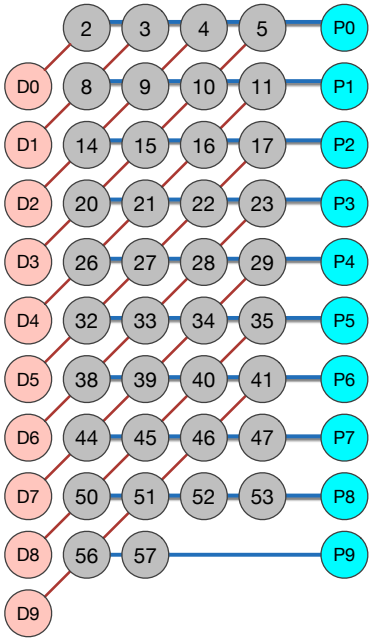


Fig. 2. A small bipartite RESAR layout with $k = 4$.

We base our RESAR layouts on a bipartite graph. RESAR layouts are scalable and can incorporate an arbitrarily large number of data disklets (or complete storage devices). Each data disklet belongs to a horizontal and a diagonal parity stripe with k data disklets and a single parity disklet.

We give an example of the graph representation of a RESAR

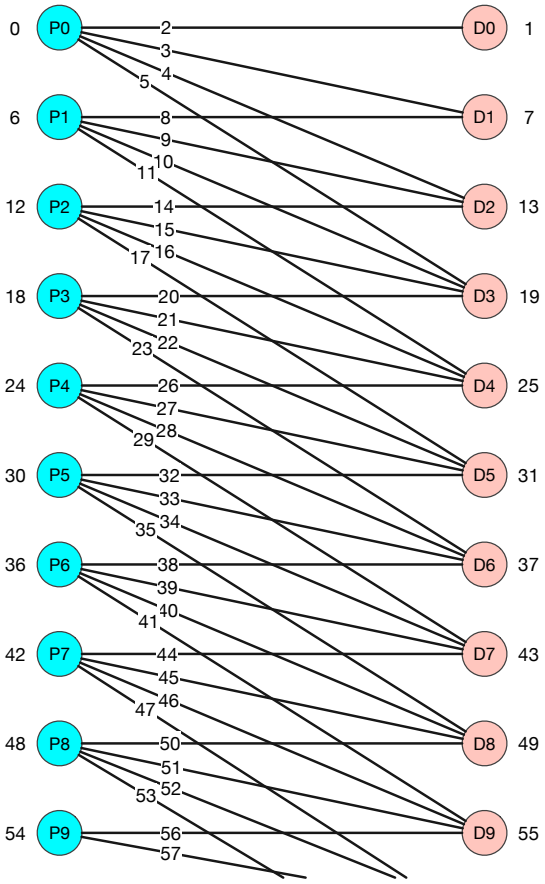


Fig. 3. The graph representation of the RESAR layout in Fig. 2.

layout, Fig. 2, in Fig. 3. The graph layout consists of two columns of parity disklets, the P- (Parity) on the left and the D- (Diagonal) parities on the right. Data disklets are represented by edges between the left and right column of vertices. Each vertex (parity disklet) has edge-rank k, meaning that there are k data disklets in each reliability stripe. Fig. 2 does not show that the RESAR layout is circular, that is we connect the upper and the lower edge of the graph in Fig. 2. Given the size of the disk arrays, this move is in reality not important, as having a few reliability stripes with less than k data disklets is not likely to be noticed. Our analysis however assumes the circular layout.

We use an enumeration of disklets not only to indicate how the disk array would grow if more disks are added, but also to control the placement of disklets in disks such that a single rack failure cannot cause data loss. If the elements in Fig. 2 denote disklets and we place disklets 33, 35, 38, and 40 in the same disk, then the failure of this disk causes data loss, since all four disklets share their two reliability stripes with another one of the four. Similarly, a rack failure can cause data loss if we place these disklets in different disks, but in the same rack.

Only for an archival workload would one use the static layout where a disk is either a parity or a data disk. Normally, we use declustering (originally called clustering by Muntz and

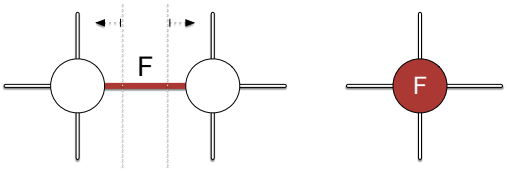


Fig. 4. Left: Failed data disklets with its neighbors. Right: Failed parity disklet with its neighbors.

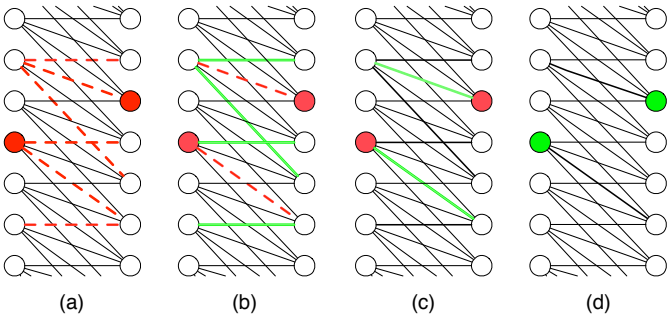


Fig. 5. Cascading recovery of multiple failures in RESAR.

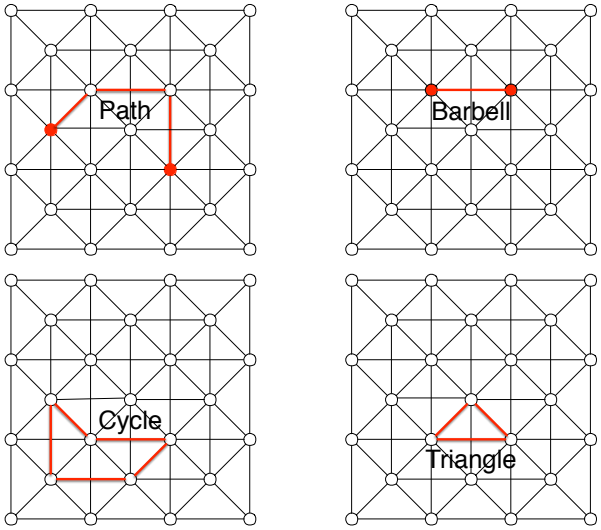


Fig. 6. Irreducible failure patterns. The barbell and the triangle on the right are irreducible failure patterns with the minimum number of failed disklets, namely three.

Liu [3]), where we create disklets, contiguous sets of blocks on a disk, which then are assigned by the layout. The Ceph system [6] uses a metadata server that deals with assigning actual disklets to the virtual disklets in the graph layout. The reason is the dynamic nature of large storage systems, where disks and other components failure is a more than daily occurrence [8] and batches of new components are introduced. Algorithms such as CRUSH have been developed to deal with this problem [7].

III. Recovery from Disk Failures in RESAR

To recover the data from a failed drive we must recover the data in each of the disklets in the drive. Since each data disklet is in two reliability stripes (see Fig. 4 for a depiction in the graph visualization), we can recover using either reliability stripe if all the other data disklets and the parity disklet in the corresponding stripe are still available. This flexibility is useful to avoid using a heavily loaded disk for recovery and is also a reason for the greater robustness of this layout. A parity disklet on a failed disk can be reconstructed in another disk if all its data disklets can be recovered. RESAR can use both stripes to halve the reconstruction time.

We use the graph visualization to discuss recovery. The graph represents disklets, not disks, and a single disk failure results in multiple disklet failures. As we mentioned, a good disklet-to-disks mapping ensures that these disklet failures resulting from the failure of a single disk are widely spread over the (rather large) graph. We can recover the data from a disklet (and then place the recovered disklet on another disk drive) if it is represented by a vertex (and is therefore a parity disklet) if all the edges (data disklets) adjacent to it are available. We can recover the data from a disklet represented by an edge (*i.e.* a data disklet) if one of the adjacent vertices and all the edges adjacent to it are available. Cascading recovery in a RESAR layout happens if a disklet can only be recovered after some of its neighbors have been recovered. Fig. 5 gives in column (a) a case with several failures in our graph layout. Failed elements are marked in red. In the upper failure cluster, there is one vertex on the left, which has lost three adjacent edges. For this reason, this vertex (or to be more precise, the reliability stripe represented by this vertex) cannot be used for recovery, but each data element in this stripe is also located in another reliability stripe. Two of the failed data elements can be recovered directly in the following step (column (b)), and the remaining failed element's data can be recovered in the third step (column (c)). The failed vertex in this cluster can only be reconstructed if all its adjacent edges are available. The lower failure cluster contains a failed vertex from which a path of failed edges emanates. This pattern resolves itself only in the fourth step.

The most frequent failure mode is that of a single block or of a disk. In both RESAR and a RAID Level 6 layout, we deal with the single block failure with the same efficiency. In the first case, we select one of the stripes and read the other blocks in the stripe, whereas in the second case, we select n out of the $n+1$ remaining blocks in the reliability stripe. If we have a non-declustered layout, RESAR has an advantage. We need to recover all the blocks on the failed disk (and write them elsewhere in the array). For this, we read half the blocks in the remaining disks of one of the two reliability stripes and half the blocks in the remaining disks of the other reliability stripe. The RAID Level 6 has more difficulties, since there is only one stripe. Each of the $n+1$ disks in it has to contribute $n/(n+1)$ of the read load. This means that a non-declustered RAID Level 6 layout takes longer in order to recover. The same is not true for a declustered array. The previous paper [5] determines entanglement, the number of blocks on two disks

that are in the same reliability stripe. It turns out that with high levels of declustering, RAID Level 6 has less entanglement than RESAR. Then, RESAR reconstructs most of the blocks on the failed disk faster, but has to wait for reading from disks highly entangled with the failed disk.

Arguments about failure tolerance are made much easier in the graph visualization than in the layout itself, as was previously observed [10]. Disk and sector failure induce a *failure pattern* in the graph. We are especially interested in patterns that represent data loss and that are minimal in the sense that removing one element of the pattern yields a pattern of failure from which we can recover. Any failure pattern that implies data loss is or contains at least one minimal failure pattern. A key observation is that an edge, that is part of a minimal failure pattern, either has end-vertices that also have failed or an end-vertex where one other adjoining edge has also failed. This allows us to classify all minimal failure patterns. They form either a cycle consisting of failed edges, or a path that starts and ends at a failed vertex and otherwise consists of failed edges in between. The smallest minimal failure patterns are the *barbell* and the *triangle*. Fig. 6 illustrates these concepts.

IV. Counting Failure Patterns in RESAR

We assume a spiral RESAR layout with an edge degree of k. We recall that this means that each reliability stripe contains k data disklets. We assume that there are N P-parities for a layout with an equal number N of D-parities, $N \times k$ edges and a total of $N \times (k+2)$ elements. If we number P-parities from 0 to $N-1$ and the D-parities also from 0 to $N-1$, then the k edges adjacent to P-parity i are adjacent to the D-parities i, $i+1 \pmod N$, $\ldots$, $i+k-1 \pmod N$.

A. Three Failures

There are two minimal three-failure patterns. The first one is the triangle. However, there is no triangles in the spiral RESAR layout. The graph is bipartite as edges connect a P-parity with a D-parity. If there would be a triangle, then one of the vertices of this triangle has to be a P-parity. Two edges of the triangle connect this P-parity to two D-parities, but these cannot be connected to form the third edge of the triangle.

The second one is the barbell, consisting of two vertices (necessarily a P- and a D-parity) and an edge, see Fig. 7. There are as many of these as there are edges, namely

$$p_{\text{Resar}}(n,k,3) = k \times N.$$

B. Four Failures

There are two minimal four failure patterns, the square and the two-path. The square consists of four edges arranged in a cycle. The two-path consists of two P- or two D-parities connected by two edges, Fig. 7 second column. We can count them easily by counting the middle vertex (that does not form part of the failure pattern). Once this one is selected, the pattern is uniquely determined by selecting the two adjacent vertices to it, which do form part of the failure pattern. Since there are $2N$ parity total, we have

$$\text{f2p}(k,N) = 2N\binom{k}{2}$$

of these patterns.

To determine the number of squares, we first observe that two of the vertices on it are horizontal and two are vertical parities, connected by the four edges that form the square. While the spiral layout wraps around, it is large, and it makes sense to speak of the upper left vertex. Edges always start at the left and either go horizontally or horizontally and down. Fig. 7 in its third column shows a square layout. Taking the upper P-vertex as a reference point, we call the relative offsets of the diagonal parities with respect to it i and j, $i < j$, respectively. In Fig. 7, $i = 1$ and $j = 3$. Let l stand for the relative offset of the other horizontal parity. Clearly $0 < l \leq i < j < k$. Any set of integers fulfilling this chain of inequalities characterizes a unique square with this upper left vertex. For $i = 0$, no square can exist. For $i = 1$, l has to be also 1, and j can take any value in $\{2, 3, \ldots, k-1\}$, giving $k-2$ possibilities. For general i, there are i possibilities for the selection of l and $k-i-1$ possibilities for selecting j. In total, there are

$$\sum_{i=1}^{k-2} i(k-i-1) = \frac{1}{6}(k-2)(k-1)k$$

possibilities for a square with given upper corner. The total number of squares is therefore

$$\text{s}(k,N) = \frac{1}{12}N \cdot \left(6 - 4k - 3k^2 + k^3\right).$$

Finally, four failure patterns can contain a minimum three failure pattern. There are

$$\text{mfp3}(k,N)(N(k+2)-3) = kN(N(k+2)-3).$$

Since the intersection between the classes (three failure pattern plus additional disk), squares, and two-paths, is empty, the total number is

$$p_{\text{Resar}}(n,k,4) = \frac{k^3N}{6} + k^2N^2 + \frac{k^2N}{2} + 2kN^2 - \frac{11kN}{3}$$

C. Five Failures

Because the graph is bipartite, it cannot include a pentagon, i.e. a cycle consisting of five edges. The only minimal 5-failure pattern is therefore the three-path, consisting of two failed vertices, connected by three failed contiguous edges. The pattern determines the middle edge uniquely. Once we have selected a middle edge, we can select the two adjacent edges, which then in turn determine the endpoints. Since there are Nk edges (the potential middle edges), which can be connected to $k-1$ edges on each side, we have a total of

$$\text{f3p}(k,N) = N \cdot k \cdot (k-1)^2$$

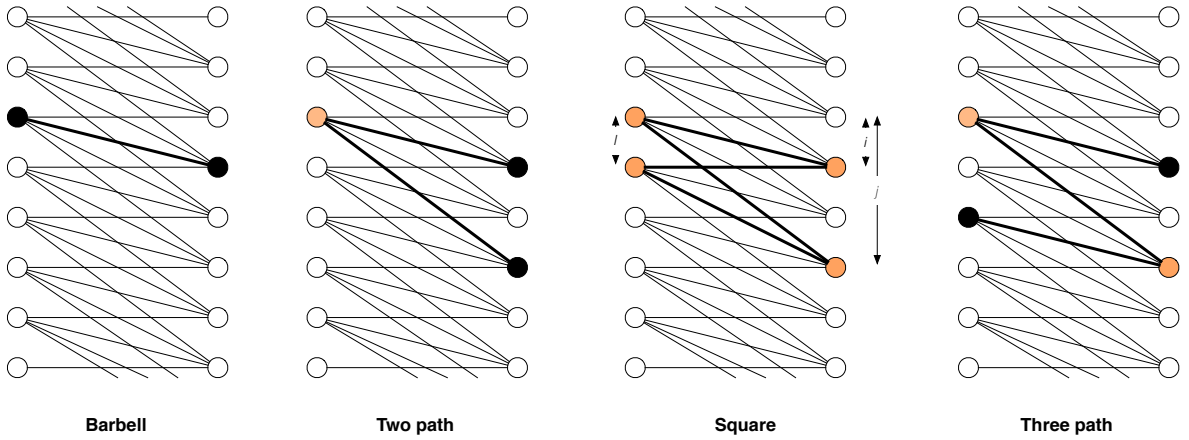
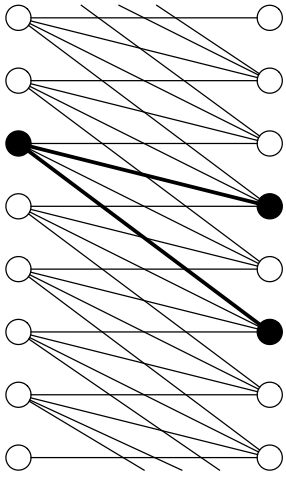


Fig. 7. Minimal failure patterns with up to five failures in a spiral RESAR layout (with $k = 4$).



Twopath cum two onepath

Fig. 8. Special five-failure patterns in a spiral RESAR layout (with $k = 4$).

minimal five failure patterns. Five failure patterns can contain also three failure patterns and four failure patterns.There are

$$N \cdot k \cdot \binom{N \cdot (k+2) - 3}{2}$$

patterns containing a barbell (we accommodate for over-counting later), and

$$\frac{1}{12} N \cdot \left(6 - 4k - 3k^2 + k^3\right) \left(N \cdot (k+2) - 4\right)$$

patterns containing a square. However, the resulting classes are however no longer disjoint. The Two-path-cum-two-barbells in Fig. 8 contains a twopaths and two barbells. Its number is exactly that of the two-paths, namely $\mathrm{f2p}(k, N) = 2N\binom{k}{2}$. In total, we have

$$p_{\text{Resar}}(n, k, 5) = \frac{35}{3}kN - 6k^2N + \frac{1}{3}k^3N - \frac{25}{3}kN^2 - \frac{19}{6}k^2N^2$$
$$+\frac{5}{6}k^3N^2 + \frac{1}{6}k^4N^2 + 2kN^3 + 2k^2N^3 + \frac{1}{2}k^3N^3.$$

V. Robustness of RAID 6

We compare our scheme with that of a RAID 6 layout. A RAID 6 layout consists of n reliability stripes with k data disks each and an additional 2 parity disks. By distributing parity, we can avoid write bottlenecks. A stripe's data survives if k of the $2 + k$ disks in it are still available. There is a total of $n(k + 2)$ disks and for f failures $\binom{n(k+2)}{f}$ ways to select them. There is no data loss, if the stripes have no failure, one failed disks, or two failed disks in them. We count the number of failure patterns that do not lead to dataloss. Let i be the number of stripes with a single failure and j be the number of stripes with a double failure. Therefore $i+2j = f$. Given i and j, we select first the stripes, giving us $\binom{n}{i} \cdot \binom{n-i}{j}$ possibilities, and within the stripes, one or two failed disks correspondingly. This gives us a total of $\binom{n}{i} \cdot \binom{n-i}{j} \cdot \binom{k+2}{1}^i \cdot \binom{k+2}{2}^j$ possibilities. The probability of dataloss with f failed disks is therefore

$$p_{\text{RAID}}(n, f, k) =$$
$$1 - \left(\sum_{i \geq 0, j \geq 0, 2j+i=f} \binom{n}{i}\binom{n-i}{j}\binom{k+2}{1}^i \binom{k+2}{2}^j \right) / \binom{n(k+2)}{f}.$$

VI. Comparison

We compare the robustness of the RESAR and the RAID 6 layout by calculating the limit of the quotient of the dataloss probabilities as its size n goes to infinity, that is, we calculate

$$\lim_{n \to \infty} p_{\text{Resar}}(n, k, f) / p_{\text{RAID}}(n, k, f).$$

It turns out that the *robustness factors* are the same for three and four failed elements

$$\begin{aligned}\lim_{n\to\infty} p_{\text{Resar}}(n,k,3)/p_{\text{RAID}}(n,k,3) &= \\ \lim_{n\to\infty} p_{\text{Resar}}(n,k,4)/p_{\text{RAID}}(n,k,4) &= \\ \lim_{n\to\infty} p_{\text{Resar}}(n,k,5)/p_{\text{RAID}}(n,k,5) &= \frac{k^2+3k+2}{6}\end{aligned}$$

We evaluate the function in Fig. 9. As we can see, the RESAR layout is quite a bit more robust, especially as k increases.

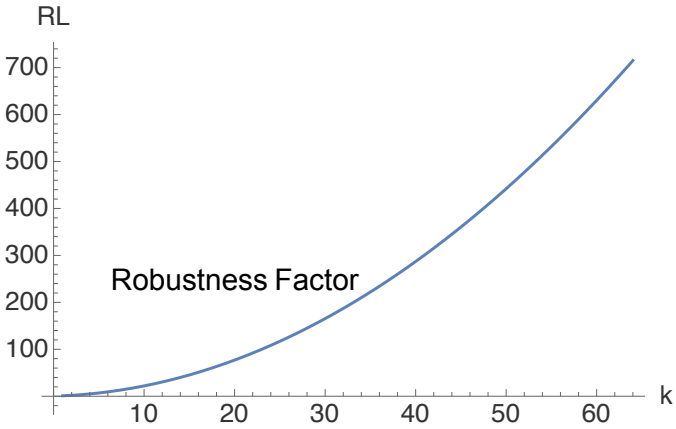


Fig. 9. Robustness factor for three, four, and five failed elements.

VII. Conclusions

We have presented detailed calculations of the probability of data loss in a novel storage layout called RESAR, presented previously [5]. As these calculations are combinatoric, we only succeeded in exact formulae for up to and including five failed storage elements. Our previous work [5] extended the determination of dataloss probability by using simulation. The results were then used in order to determine the robustness of RESAR and RAID 6 in the declustered case, where each disk contains several disklets and where we distribute the disklets according to the RESAR and the RAID 6 architecture. Declustering makes sense as it decreases the time to repair and therefore the window of vulnerability after a failure, where the system is susceptible to additional failures that can lead to dataloss.

Nevertheless, exact results are preferable to particular results obtained by simulation and therefore only valid for the simulated disk arrays. This paper thus fills an important lacuna in [5]. It shows in a mathematically exact manner that the RESAR layout has a much higher failure tolerance than the standard RAID 6 layout.

References

[1] P. Corbett, B. English, A. Goel, T. Grcanac, S. Kleiman, J. Leong, and S. Sankar, "Row-diagonal parity for double disk failure correction," in *Third Usenix Conference on File and Storage Technologies (FAST)*, 2004.

[2] K. M. Greenan, X. Li, and J. J. Wylie, "Flat XOR-based erasure codes in storage systems: Constructions, efficient recovery, and tradeoffs," in *IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, 2010.

[3] R. R. Muntz and J. C. Lui, "Performance analysis of disk arrays under failure," in *Proceedings of the* 16^{th} *VLDB Conference*, 1990.

[4] T. Schwarz, D. D. Long, and J. F. Pâris, "Reliability of disk arrays with double parity," in *IEEE 19th Pacific Rim International Symposium on Dependable Computing (PRDC)*, 2013.

[5] T. Schwarz, A. Amer, T. Kroeger, E. Miller, D. D. Long, and J. F. Pâris, "RESAR: Reliable storage at exabyte scale," in *Proceedings, IEEE International Symposium on Modeling, Analysis and Simulation of Computers and Telecommunication Systems (MASCOTS)*, 2016.

[6] S. A. Weil, S. A. Brandt, E. L. Miller, D. D. Long, and C. Maltzahn, "Ceph: A scalable, high-performance distributed file system," in *Proceedings of the 7th Symposium on Operating Systems Design and Implementation*. USENIX Association, 2006.

[7] S. A. Weil, S. A. Brandt, E. L. Miller, and C. Maltzahn, "Crush: Controlled, scalable, decentralized placement of replicated data," in *ACM/IEEE Conference on Supercomputing*, 2006.

[8] Q. Xin, E. L. Miller, T. Schwarz, D. D. Long, S. A. Brandt, and W. Litwin, "Reliability mechanisms for very large storage systems," in *Proceedings 20th IEEE/11th NASA Goddard Conference on Mass Storage Systems and Technologies (MSST)*, 2003.

[9] L. Xu, V. Bohossian, J. Bruck, and D. G. Wagner, "Low-density MDS codes and factors of complete graphs," *IEEE Transactions on Information Theory*, vol. 45, no. 6, pp. 1817–1826, 1999.

[10] J. Zhou, G. Wang, X. Liu, and J. Liu, "The study of graph decompositions and placement of parity and data to tolerate two failures in disk arrays: Conditions and existence," *Chinese Journal of Computers (chinese edition)*, vol. 26, no. 10, pp. 1379–1386, 2003.