Reliability of Disk Arrays with Double Parity

Thomas Schwarz, S.J.
Universidad Católica del Uruguay
Montevideo, Uruguay
tschwarz@calprov.org

Darrell D.E. Long
University of California
Santa Cruz, CA
darrell@cs.ucsc.edu

Jehan-François Pâris
University of Houston
Houston, TX
jfparis@uh.edu

***Abstract*—We present a general method for estimating the risk of data loss in arbitrary two-dimensional RAID arrays where each data disk belongs to exactly two single-parity stripes. We start by representing each array organization by a graph where each parity stripe, and its associated parity disk, is represented by a node and each data disk by an edge. We then use this representation to identify and enumerate minimal sets of disk failures, say, triple failures, quadruple failures and so forth, that will cause a data loss. The overall probabilities that a given number n of disk failures will cause a data loss is then given by the ratio of the total number of fatal disk failures involving n disks over the total number of possible failures of n disks.**

To illustrate the power of our method, we apply it to two distinct, archival two-dimensional array organizations. The first, "square" organization is a traditional square layout where data disks are formed into a square and the parity stripes are formed by the rows and columns in the square. Hence a square layout organization with n^2 data disks will have $2n$ parity disks. The second, "complete" organization corresponds to a closer weave, where all parity stripes intersect and each intersection contains a parity disk. This organization with n parity disks will have $n(n-1)/2$ data disks. Our results show that previous ad hoc estimates of the reliability of these arrays significantly underestimated their reliability by assuming that either all triple or all quadruple disk failures were fatal. We show that the two two-dimensional array organizations exhibit mean times to data loss and five-year survival rates that are very similar to those of a RAID Level 6 organization of much smaller capacity. Our complete organization is about 4.5 times and the square organization is about 8 times more reliable than a disk array with same storage capacity built from RAID level 6 stripes.

***Index Terms*—Disk array organization, archival storage system, Markov model, mean time to data loss, five year survival rate**

I. Introduction

As disks move towards a more archival role in the storage hierarchy, finding cost-effective solutions for the long-term storage of archival data becomes even more important. Archival data is rarely modified once written, but needs to be maintained safely for decades.

Data is threatened by a variety of failure modes, such as latent sector failures, hard drive failures, or central component failures (such as cooling and networking). Fortunately, most component failures do not destroy the data but only access to the data and latent sector failures affect only small amounts of data. By scrubbing (periodically verifying the capacity to read a sector) or by writing internal parity data, the incidence rate and the effects of these latent sector errors can be controlled [2], [3], [10], [15]. This leaves hard drive failure as the most critical type of failure. Even if it is known that a drive is about to fail, it takes hours to remove all of its data.

Instead of replicating data, we store it redundantly to make better use of storage capacity. We use erasure coding such as the well-known m-out-of-n codes. The best known examples of the use of these codes are the RAID 5 (with an m-out-of-$m+1$ code and RAID 6 organizations (with an m-out-of-$m+2$ code. The data is stored in *data blocks* and the code is used to calculate parity data over sets of data blocks and store it in parity blocks. In an archival storage system, longevity and capacity are more important than good load distribution among the disks, and it makes sense to use dedicated parity disks.

Two-dimensional RAID arrays with separate parity disks have been considered before [9], [14], [17]. They are especially attractive for archival storage because they protect their data against all double and most triple, quadruple, and even quintuple failures [14]. Unfortunately nearly all other studies assumed that either all triple failures [9] or all quadruple failures [12], [13] cause data loss. Yokota's work on DR-nets is an exception, but limits itself to a very small organization [16]. Depending on the approach, these organizations were too quickly dismissed as mere curiosities [9], [17] or had their mean time to data loss significantly underestimated [12], [13].

We extend here our previous work [1], [12] and give a general method to estimate the reliability of many disk arrays with protection against two simultaneous failures in a more accurate fashion. We then apply these principles to two different array organizations. The first, *square* organization is a traditional square layout where data disks are formed into a square. Parity stripes are constituted by the rows and columns of the square. To each parity stripe, we add a parity drive. Such an organization with n^2 data disks will have $2n$ parity disks. Our second, the *complete* organization corresponds to a closer weave, where all parity stripes intersect in exactly one parity disk. An organization with n parity disks will have $n(n-1)/2$ data disks. Both of our organizations are two-dimensional arrays where each reliability stripe contains a single parity disk that contains the ordinary eXclusive-OR (XOR) parity unlike a RAID level 6 stripe where calculating the contents of the second parity disk is more involved.

Arguing about the reliability of two-dimensional disk array organizations is difficult, but by moving to a graph-theoretic description of these arrays, arguments about reliability become simpler. In this manner, we calculate the reliability of these two organizations and compare them that of an organization consisting of several RAID level 6 stripes. Our method combines exact counting with extensive simulation. For small number of failures, we calculate exactly the probability of data loss. For larger number of failures, we use simulation to determine the data loss probability. We then determine Mean Time To Data Loss (MTTDL) figures and five year survival probabilities under various assumptions such as disk infant mortality. Our results show that these two two-dimensional organizations exhibit about the same five-year survival rate as does a single RAID level 6 stripe with the same number of data disks in its only reliability stripe, despite a much larger number of data disks in the organization. To store the same amount of data as in the square organization, one would have to use eight RAID level 6 and for the alternate two-dimensional organization, four and a half.

In Section 2, we review our generic method for designing two-failure resilient data layouts. Section 3 calculates the probability that data loss has occurred if there are a small number f of failed disks in the organization. Section 4 shows how to generate Markov models for the disk array organizations that we consider here. Section 5 discusses the reliability results for these organizations.

II. Graph Representation

Arguments about the reliability of disk arrays can be complicated since they are often abstract. We use a visualization that applies to many disk array organization with double failure protection that only uses simple (exclusive-or) parity for protection. In it, arguments about the effects of disk failures are translated into arguments about graphs, as was previously observed [11]. The visualization is mathematically exact and does not lose information. It functions like Feynman diagrams in theoretical physics by representing abstract properties in a more intuitive way.

We consider disk array organizations with dedicated *parity disks*. User data is stored on *data disks*. We only allow eXclusive OR (XOR) operations to calculate parity data. This means that every parity disk contains the exclusive-or of the contents of a group, the *reliability stripe*. A disk array with this type of organization is two-failure tolerant if and only if:

1) Each data disk belongs to exactly two different reliability stripes;
2) The intersection of two reliability stripes consists at most of one data disk.

If we require additionally that each reliability stripe has exactly the same number n of data disks, then the resulting mathematical structure is a called a *configuration* in Mathematical Design Theory [8]. The reliability stripes are the *blocks* of the configuration in the language of mathematical design theory and the disks are the elements. A typical move in design theory is to consider the *dual* of a design. The blocks of the design become the objects of the dual design, and the objects of the design become the blocks of the dual design. The *is-element-of* relation between objects and blocks in the design is inverted for the dual design. It turns out that the dual design of a configuration is an n-regular graph, which is a structure that is very familiar to computer scientists.

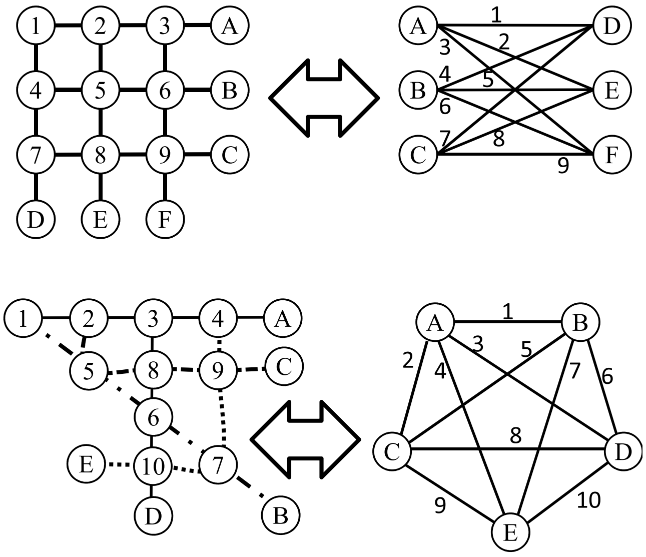

Fig. 1. Two small disk array organizations (square and complete layout) and their design theoretical dual graphs (right).

Figure 1 gives an example of the traditional square layout in the upper line. The data disks (1, …, 9) are arranged in the columns and rows of a square with 3 × 3 disks. Each row and column forms a reliability stripe with an additional parity disk (A, …, F). To the right, we give the design-theoretical dual. As each parity stripe has only one parity disk, we use the label of the parity disk to also label the stripe. For example, Disk 7 is situated in the reliability stripes of Disks C and D. In the dual, Disk 7 corresponds to the edge between vertices C and D. Below we give the complete graph with five vertices and ten edges. To the left, we draw the corresponding disk array organization, the *complete* organization. Edge 7 connects vertices B and E. Therefore, Disk 7 forms part of the reliability stripes with parity disks B and E respectively. The former is made up of disks 1, 5, 6, and 7, the latter of disks 4, 7, 9 and 10.

We can translate the properties of our two-failure tolerant design by substituting "edge" for data disk and "vertex" for reliability stripe. Our properties for two-failure tolerant arrays then become

1) Each edge is connected to exactly two different vertices;
2) Each pair of vertices are joined by at most one edge.

These are exactly the properties of a graph. If we require additionally that each reliability stripe contains n data disks, then we require that there are n edges emanating from each vertex. This is exactly the definition of n-regularity of a graph.

The dual design allows simple arguments about data loss, as has been previously observed [11]. If a parity disk has failed, then we can reconstruct its contents if and only if we can access the data of all the data disks belonging to the reliability stripe. Accessing data might include previous reconstruction

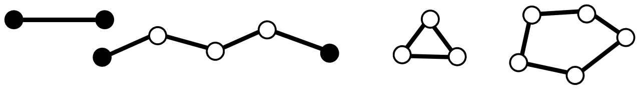

Fig. 2. Minimal irreducible failure patterns. (Failed parity disks (vertices) are represented by filled circles, good parity disks by non-filled circles, and failed data disks (edges) by lines).

steps. If a data disk has failed, we can reconstruct its content if the data on all the other data disks and the parity disks are accessible.

In order to reason about data loss, we look for set of disks whose failure implies data loss and call them *failure sets*. For the analysis, *minimal* failure sets are of interest. A minimal failure set does not properly contain any other failure set. We can show that there are only two types of minimal failure sets. The first type consists of two vertices (i.e. two failed parity disks) and all the edges in a path connecting them (Fig. 2, left). The second type consists of a circle of failed edges (Fig. 2, right). This simple observation allows us to count failure sets of a given small cardinality.

III. Combinatorial Determination of Minimal Failure Sets

We start with an exact model for the reliability of two-failure resilient disk arrays. Our method first calculates exactly the data loss probability given that a disk array organization has suffered f failures. We can do this with combinatorics for smaller values of f. For the remaining cases, we can use complete enumeration over all f-sets of failed disks if the number of disks in the disk array is small, or we can use Monte Carlo simulation to determine the failure probability within a narrow confidence interval. Our results justify this procedure because they show that the accuracy of reliability measure depends on the accuracy of data loss probabilities only for small f.

We calculate the probability that a random failure pattern with f failed elements has resulted in data loss by counting the number of failure patterns. We recall that in the graph description of the disk array, a failure pattern with data loss contains either an edge cycle, a path with two end vertices failed, or both. We will add up the failure patterns with certain minimal, data-loss inducing failure patterns and then adjust for overcounts.

A. Complete Organization

We first consider the complete graph with n vertices. This graph has vertex degree $n-1$, $\binom{n}{2}$ edges, and a total of $N = n + \binom{n}{2} = \binom{n+1}{2}$ *elements,* which encompasses both vertices and edges.

The *complete* organization is the dual of the complete graph and has $n-1$ data disks per reliability stripe for a total of $n(n-1)/2$ data disks and n parity disks. The storage overhead is $2/(n-1)$. Among all two-failure tolerant disk arrays representable by a graph, this layout has the smallest

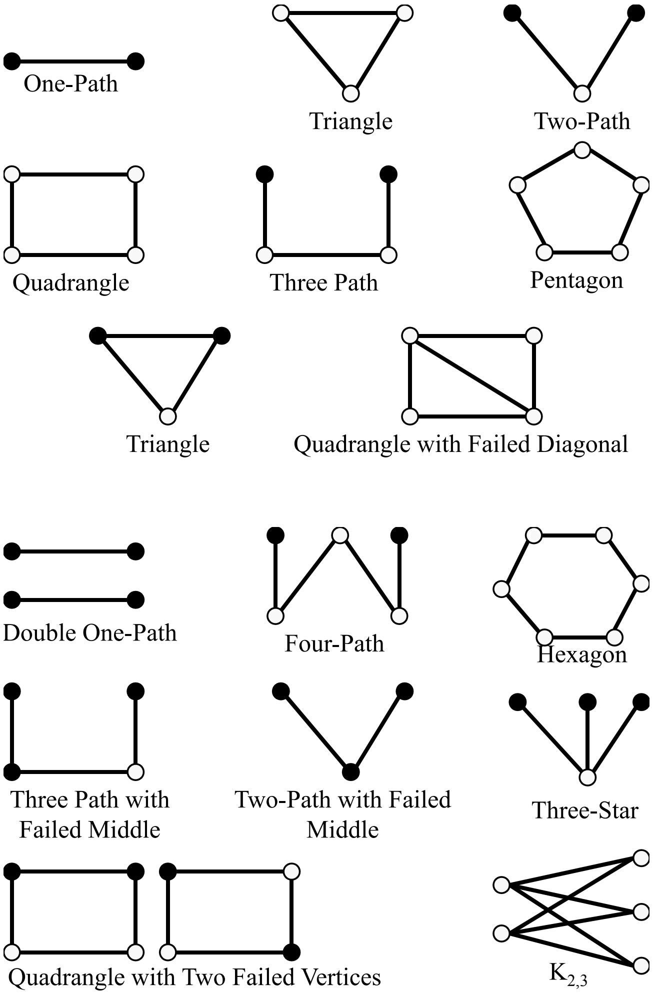
One-Path
Triangle
Two-Path
Quadrangle
Three Path
Pentagon
Triangle
Quadrangle with Failed Diagonal
Double One-Path
Four-Path
Hexagon
Three Path with Failed Middle
Two-Path with Failed Middle
Three-Star
Quadrangle with Two Failed Vertices
$K_{2,3}$

Fig. 3. Failure pattern definitions. Black nodes and edges represent failed elements.

number of parity disks possible for the size of the reliability stripe. It is also the smallest possible disk array for the size of the reliability stripe. As such, it seems to be an attractive layout for a modular storage system with a total of 36, 45, 55, or 66 disks for reliability stripes of length 8, 9, 10 and 11, respectively.

For $f = 3$, the data loss inducing 3-failure patterns are the triangle and the one-path. A triangle is uniquely determined by the three vertices that adjoin the three edges and their number is $\binom{n}{3}$. A one-path is made up of two vertices and the edge between them, so that their number is $\binom{n}{2}$. Thus,

$$f_{\text{pc3}}(n) = \binom{n}{2} + \binom{n}{3}.$$

For $f = 4$, we have two more patterns, the quadrangle and the two-path (see Figure 3. A quadrangle defines a four-set of vertices, but each four-set of vertices can be made into a quadrangle in three different ways. Therefore, the number of quadrangles is equal to $3\binom{n}{4}$. A two-path is given by the two-set of end-vertices and the vertex in the middle, their number is $(n-2)\binom{n}{2}$. The three-failure patterns with data loss give a four-failure pattern by choosing one additional element. Their number is $f_{\text{pc3}}(n)(N-3)$. In total, we have

$$f_{\text{pc4}}(n) = f_{\text{pc3}}(n)(N-3) + 3\tbinom{n}{4} + (n-2)\tbinom{n}{2}.$$

For $f = 5$, we have two more minimal failure patterns with data loss, the pentagon and the three-path. A pentagon determines a set of five vertices, and each such set gives rise to several pentagons. We now count these pentagons as follows: we label the vertices in some order with 1, 2, 3, 4, and 5. We first count pentagons with orientation. Each pentagon with orientation is given by a permutation of these vertex labels with 1 in the first place. This gives us 4! possibilities. Since a normal pentagon can be oriented in two directions, this gives us half as many pentagons per 5-set of vertices, or 12 ways. Since there are $\binom{n}{5}$ 5-sets of vertices, we have $12\binom{n}{5}$ possible pentagons.

The three-path is defined by two (failed) end-vertices, two (not failed) interior vertices, and the two ways in which to connect the four vertices to form the three-path. We therefore have $2\binom{n}{2}\binom{n-2}{2}$ three-paths.

5-failure patterns with data loss containing a 4-failure pattern as a minimal pattern are given by choosing a quadrangle or a two-path and one additional element. There are $(3\binom{n}{4} + (n-2)\binom{n}{2})(N-4)$ of these patterns. Those containing a 3-failure pattern as a minimal pattern are given by $(\binom{n}{2} + \binom{n}{3})\binom{N-3}{2}$, after we subtract the number of triangles with two failed vertices from it, since these contain a one-path, a triangle, and also a two-path. There are $3\binom{n}{3}$ of these triangles with two failed vertices. A quadrangle with diagonal contains a quadrangle and two triangles. We count them by choosing two vertices, which are on the quadrangle and the diagonal and by choosing then two more vertices, which are only on the quadrangle. Therefore, we have $\binom{n}{2}\binom{n-2}{2}$ quadrangles with diagonal. Finally, a two-path with failed middle contains two one-paths and a two-path. There are as many of them as there are two-paths, namely $(n-2)\binom{n}{2}$. We obtain the number of 5-failure patterns with data loss by adding the first set of numbers and then subtracting the overcounted patterns weighted according to the number of patterns they contain. Since all these patterns are counted thrice, but they should only be counted once, we subtract by twice their number. This gives us

$$\begin{gathered} f_{\text{pc5}}(n) = 12\tbinom{n}{5} + 2\tbinom{n}{2}\tbinom{n-2}{2} \\ + \left(3\tbinom{n}{4} + (n-2)\tbinom{n}{2}\right)(N-4) + \left(\tbinom{n}{2} + \tbinom{n}{3}\right)\tbinom{N-3}{2} \\ -6\tbinom{n}{3} - 2\tbinom{n}{2}\tbinom{n-2}{2} - 2(n-2)\tbinom{n}{2}. \end{gathered}$$

We supplemented these formulae with the results of a complete enumeration and evaluation of all failure patters for the cases of six and seven failed disks and then Monte Carlo simulation to obtain the data loss probabilities for the complete organization with 45 disks in Table I.

B. Square Organization

The design theoretical dual of the square disk array layout is a bipartite graph of degree n with $2n$ vertices, divided into a *left* and *right side* of n vertices each and n^2 edges connecting each vertex on the left side to a vertex on the right side. In a bipartite graph, all edge cycles have even lengths.

TABLE I
Number f_{PB} of failure patterns with data loss and data loss probability P_{dataloss} for a complete array organization with 9 parity and 36 data disks with f failed disks.

f	f_{pb}	P_{dataloss}
3	120	0.845666%
4	5670	3.8055%
5	129654	10.61211%
6	1887060	23.1682%
7	19279620	42.4852%
8		66.74879%
9		88.70095%
≥ 10		100.00%

The square disk array arranges the data disks into a $n \times n$ square of disks. Each column and row forms a reliability stripe with an additional parity disk. Thus, there are $2n$ parity disks for a n^2 data disks and there are n disks in a reliability stripe. This gives the same overhead as for the complete organization. The total number of disks in the square layout is quite a bit larger for the same reliability size, 80, 99, 120, and 143 for stripe sizes of 8, 9, 10 and 11, respectively.

The smallest failure pattern with data loss is the one-path, consisting of a failed edge and the two adjoining vertices. They correspond to the edges and therefore, there are $f_{\text{pb3}}(n) = n^2$ of them.

The minimum failure pattern with data loss and four elements are the quadrangle and the two-path. In a bipartite graph, a quadrangle corresponds one-to-one to the selection of two vertices on the left and two vertices on the right side, for a total of $\binom{n}{2}^2$ quadrangles. A two-path starts and ends on one side, where we chose the two end-vertices, and touches on the other side, where we chose one vertex. There are $2n\binom{n}{2}$ of them. The four failure pattern containing a one-path are counted by $n^2(N-3)$. Therefore:

$$f_{\text{pb4}}(n) = n^2(N-3) + \tbinom{n}{2}^2 + 2n\tbinom{n}{2}.$$

The only minimal 5-failure pattern with data loss is the three path. It corresponds one-to-one to the selection of one end-vertex on either side and one interior vertex on either side, so that there are $n^2(n-1)^2$ of them. Other 5-failure pattern with data loss result from choosing a minimal 3-failure pattern with data loss and two additional failed elements and a minimal 4-failure pattern with data loss and one additional failed element. However, the intersection between these groups is non-empty and we cannot simply add them up without adjusting for overcounting. The intersection consists of the two-path with middle vertex, which is counted twice as a one-path and once as a two-path. There are as many of them as there are two-paths, namely $2n\binom{n}{2}$. We therefore obtain

$$\begin{aligned} f_{\text{pb5}}(n) \quad = \quad & n^2\tbinom{N-3}{2} + (2n\tbinom{n}{2} + \tbinom{n}{2}^2)(N-4) \\ & + n^2(n-1)^2 - 4n\tbinom{n}{2}. \end{aligned}$$

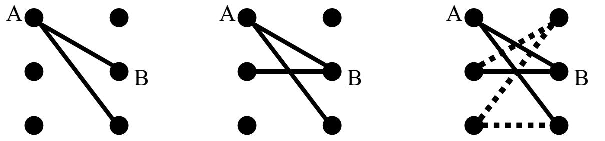

Fig. 4. Counting Hexagons

The minimal 6-failure patterns with data loss are the hexagon and the four-path. A hexagon is an edge-cycle and defines three vertices each on either side. However, if we are given two 3-sets of vertices on both sides, then there are various ways to select the edges between them to form a hexagon. An ad-hoc count argues as follows: We select the vertex with minimal ID and call it A. Vertex A needs to be connected by two edges in the hexagon to two of the vertices in the 3-set on the other side. This already gives us three choices. We pick the vertex with minimal ID among the neighbours of A in the hexagon and call it B. By choice, B is already connected by an edge in the hexagon to A, but it also needs to be connected to another vertex in the 3-set on the same side as A. This gives us two choices. A quick drawing will now convince the reader that with these choices there is only one way to complete a hexagon. Therefore, given two 3-sets on either side, there are 6 ways to generate a hexagon and there are $6\binom{n}{3}^2$ hexagons in total. Figure 4 illustrates our counting process.

The other minimal 6-failure pattern with data loss is the four-path. It starts out with one failed vertex, follows an edge to the other side, returns with a neighbouring edge to the starting side, and repeats this process once to incorporate the end vertex. It therefore determines two failed vertices on one side, another vertex on the same side, and finally two vertices on the other side. However, given such a configuration, there are two ways to create a four-path since we can connect the starting vertex to either of the two vertices on the other side. The number of four-paths is therefore $4(n-2)\binom{n}{2}^2$.

The number of 6-failure patterns with data loss that contain a minimal 3-failure pattern is the number of one-paths multiplied with the number of choices for three additional failed elements or $\binom{n}{2}\binom{N-3}{3}$, but for overcounting certain patterns for which we will account below.

Similarly, the raw number of patterns with data loss that contain a minimal 4-failure pattern is $\left(2n\binom{n}{2}+\binom{n}{2}^2\right)\binom{N-4}{2}$ and the raw number of those containing a minimal 5-failure pattern is $n^2(n-1)^2\binom{N-5}{1}$.

There is a number of 6-failure pattern that we count multiple times in this manner. We first have the double one-path configuration, consisting of two one-paths without an element in common. To count them, we select two failed end-vertices on either side. There are then two ways to connect these 2-sets, for a total of $2\binom{n}{2}^2$ patterns. A pair of two one-paths can have one end-vertex in common, which gives the 5-failure pattern we call a two-path with middle. We can then pick an additional failed element to count $2n\binom{n}{2}\binom{N-5}{1}$. A quadrangle can have two failed vertices on the quadrangle itself. Depending on their location, we either count this configuration as a one-path, a three-path, and a quadrangle, or as two two-paths and a quadrangle. There are $\binom{4}{2}\binom{n}{2}^2$ of them. Another overcounted 6-failure pattern with data loss is the three star, consisting of three edges sharing one end-vertex, which has not failed, and where the other end-vertex has failed. We count, we select sides, then one vertex on one side and three vertices on the other side, giving us $2n\binom{n}{3}$. We can also have a three-path, where one of the two middle vertices on the path also has failed. There are $2n^2(n-1)^2$ of them. Finally, we can obtain two quadrangles to obtain a pattern consisting of the edges of a complete bi-partite graph with two vertices on one side and three on the other side. This pattern contains three quadrangles. With the exception of the double one-path, all patterns are counted thrice. We add up the raw numbers, subtract patterns overcounted and obtain after some algebraic manipulation

$$f_{\text{pb6}} = \frac{7}{24}(n-1)n^2(n(n+3)(n(n(n+4)-1)-34)+144).$$

It is obviously possible to continue counting, but as n increases, the number of failure patterns also increases as does the complexity of the arguments. We used Monte Carlo simulation and for small organizations complete enumeration and evaluation of all failure patterns to determine the failure probabilities given in Table II for the square layout with 16 parity and 64 data disks.

TABLE II
NUMBER f_{PB} OF FAILURE PATTERNS WITH DATA LOSS AND DATA LOSS PROBABILITY P_{dataloss} FOR THE SQUARE ORGANIZATION WITH 16 PARITY AND 64 DATA DISKS WITH f FAILED DISKS

f	f_{pb}	P_{dataloss}
3	64	0.0778968%
4	6160	0.389484%
5	283136	1.17777%
6	8366848	2.78431%
7		5.6615%
8		10.3027%
9		17.2953%
10		27.0493%
11		39.58726%
12		54.27081%
13		69.66938%
14		83.44394%
15		93.39227%
16		98.55550%

C. Level 6 RAID

We compare both organizations considered before against a Level 6 RAID, consisting of n reliability stripes, each with two parity disks and k data disks. This organization is not representable as a graph and we have to count failure patterns directly.

If f disks among the $n(k+2)$ have failed, then the array still survives without data loss if there is at most two failures per reliability stripe. If there are i stripes with one failure and j stripes with two failures, then $i+2j=f$ and there are

TABLE III
NUMBER f_{PB} OF FAILURE PATTERNS WITH DATA LOSS AND DATA LOSS PROBABILITY P_{dataloss} FOR A RAID LEVEL 6 ORGANIZATION WITH 8 RELIABILITY STRIPES AND $8+2$ DISKS PER STRIPE

f	f_{pb}	P_{dataloss}
3	960	1.16845%
4	68880	4.35514%
5	2438016	10.1415%
6	56347200	18.7511%
7	951566400	29.9544%
8	12472493400	43.0271%
9	131768547200	56.8212%
10	1152082285120	69.9719%
11	8509194814400	81.2126%
12	54043627682800	89.7040%
13	300152603340800	95.2453%
14	1481912331702400	98.2601%
15	6605976260490560	99.5495%
16	26941406005117900	99.9376%

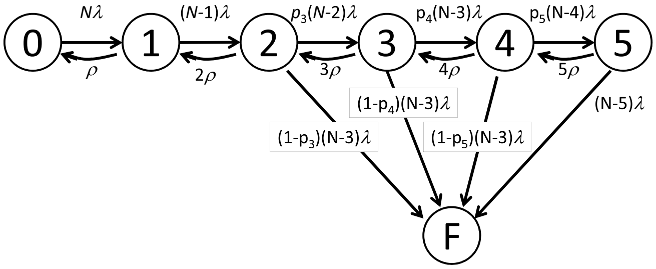

Fig. 5. Standard Markov model with 5 non-failure states.

$(k+2)$ ways to select the failed disk in a stripe with one failure and $\binom{k+2}{2}$ of selecting two failed disks in a stripe with two failures. Therefore the total number of arranging x failures without data loss is

$$f_{\mathrm{pb}f} = \sum_{i+2j=f} \binom{n}{i,j,n-i-j} (k+2)^i \binom{k+2}{2}^j.$$

Table III gives the results for a RAID Level 6 layout with 80 disks organized in 8 reliability stripes. This layout has the same number of disks and the same ratio of parity and data disk as the bipartite graph layout that we discussed in the previous section. A glance at the data loss probabilities reveals that the square organization is quite a bit more robust.

D. Failure Probability Results

We can combine our general formula with computer enumeration and simulation to obtain the probability of data loss given a certain number of disk failures. For small numbers of failures, our formulae derived above apply. For small arrays, it is possible to obtain an exact number by enumerating all possible cases. When this is no longer possible, we can use simulation in order to obtain approximate results. It is possible to narrow the 99% confidence interval to parts of a percent of the value. We give our results in Tables IV and V.

IV. MARKOV MODELS

To move from failure probabilities to disk array reliability measures, we use a Markov model. This gives us directly the mean time to data loss numbers and allows us to arm Monte Carlo simulations to determine their five year survival rate.

Our Markov model is one-dimensional, which makes it amenable for quick simulation, including when the distribution of times between transitions are not exponentially distributed. This allows us to assess the importance of more “realistic” disk failure behavior. Our Markov model consists of an absorbing Failure State F and States 0, 1, …, M. State i describes the disk array with i failed disks. M is the maximum number of failed disks in a specific organization that might not have lead to data loss. For example, the organization consisting of eight Level 6 RAID stripes might not have lost data with $M = 16$ failure, even though the probability is less than 0.1 per cent.

Figure 5 shows our Markov model with $M = 5$ non-failure states. This Markov model applies to a system that can withstand any two failures, but fails sometimes when three, four, or five disks have failed, and always when six disks have failed. From any state but the failure state, we can transition to the previous state with a repair transition. For the mean time to data loss calculations, we assume for ease of modeling that repairs are independently and exponentially distributed with a mean time to repair of ρ. If we are in State 0, we transition to State 1 with a transition rate of $N \cdot \lambda$ where $1/\lambda$ is the mean time to failure of a disk. In State 2, we transition to State 2 with a transition rate of $(N-1)\lambda$, because now we have $N-1$ disks instead of N disks. We transition out of State 2 at a rate of $(N-2)\lambda$ because of failures. If the data loss probability for three failures is equal to $p_{\mathrm{dl}}(3)$, then we transition with probability $p_3 = 1 - p_{\mathrm{dl}}(3)$ to State 3 (yielding a rate of $p_3(N-2)\lambda$) and with probability $1 - p_3 = p_{\mathrm{dl}}(3)$ to the failure State F. As discussed, we also transition with rate 2ρ from State 3 to State 2 because of a repair.

Similarly, we transition from State 3 to State 4 at a rate of $p_4(N-3)\lambda$ and to State F at a rate of $(1-p_4)(N-3)\lambda$. The relationship between the probability $p_{\mathrm{dl}}(4)$ of data loss given four disk failures and the transition probability p_4 is slightly more involved. Data loss with four failures could result from two different scenarios. First, it could just be that the chronologically first three failures already caused data loss. This would have happened with probability $1 - p_3 = p_{\mathrm{dl}}(3)$. Second, it could have happened that the chronologically first three failures did not cause data loss, but that the fourth did. This would have happened with probability $p_3(1-p_4) = (1 - p_{\mathrm{dl}}(3))(1-p_4)$. As both scenarios are disjoint,

$$p_{\mathrm{dl}}(4) = (1-p_3) + p_3(1-p_4).$$

We solve this equation for p_4 to obtain

$$p_4 = 1 - \frac{p_{\mathrm{dl}}(4) - (1-p_3)}{p_3}.$$

In general, we have

$$p_{\mathrm{dl}}(i) = p_{\mathrm{dl}}(i-1) + (1 - p_{\mathrm{dl}}(i-1))(1-p_i).$$

TABLE IV
DATA LOSS PROBABILITY IN PER CENT FOR THE COMPLETE ARRAY ORGANIZATION

Nr. parity disks	7	8	9	10	11	12
Nr. data disks	21	28	36	45	55	66
Failures						
0	0.000	0.000	0.000	0.000	0.000	0.000
1	0.000	0.000	0.000	0.000	0.000	0.000
2	0.000	0.000	0.000	0.000	0.000	0.000
3	1.709	1.176	0.846	0.629	0.481	0.376
4	7.863	5.348	3.805	2.806	2.129	1.654
5	22.051	14.973	10.612	7.786	5.877	4.544
6	46.726	32.407	23.168	17.045	12.864	9.927
7	77.860	57.558	42.485	31.775	24.195	18.740
8	100.000	84.194	66.595	51.784	40.319	31.650
9	100.000	100.000	88.708	73.943	60.065	48.419
10	100.000	100.000	100.000	91.948	79.820	67.248
11	100.000	100.000	100.000	100.000	94.241	84.450
12	100.000	100.000	100.000	100.000	100.000	95.874
13	100.000	100.000	100.000	100.000	100.000	100.000

TABLE V
DATA LOSS PROBABILITY IN PER CENT FOR THE SQUARE ORGANIZATION

Nr. parity disks	4	6	8	10	12	14	16	18	20	22	24
Nr. data disks	4	9	16	25	36	49	64	81	100	121	144
Failures											
0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
2	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
3	7.143	1.978	0.791	0.382	0.208	0.123	0.078	0.052	0.036	0.025	0.019
4	35.714	9.890	3.953	1.910	1.041	0.617	0.389	0.258	0.178	0.127	0.093
5	100.000	29.670	11.971	5.791	3.154	1.868	1.178	0.780	0.537	0.382	0.280
6	100.000	64.196	27.710	13.628	7.454	4.418	2.784	1.838	1.270	0.906	0.658
7	100.000	100.000	52.326	27.021	15.024	8.954	5.661	3.747	2.569	1.829	1.334
8	100.000	100.000	80.880	46.477	26.837	16.222	10.303	6.839	4.707	3.339	2.436
9	100.000	100.000	100.000	69.604	43.110	26.876	17.295	11.548	7.962	5.670	4.111
10	100.000	100.000	100.000	89.931	62.492	41.009	27.049	18.262	12.676	9.035	6.601
11	100.000	100.000	100.000	100.000	81.329	57.705	39.587	27.283	19.130	13.725	10.034
12	100.000	100.000	100.000	100.000	94.726	74.699	54.270	38.611	27.559	19.934	14.681
13	100.000	100.000	100.000	100.000	100.000	88.811	69.670	51.773	37.915	27.862	20.670
14	100.000	100.000	100.000	100.000	100.000	97.233	83.444	65.741	49.888	37.431	28.141
15	100.000	100.000	100.000	100.000	100.000	100.000	93.392	78.879	62.636	48.386	37.077
16	100.000	100.000	100.000	100.000	100.000	100.000	98.556	89.386	75.030	60.169	47.220
17	100.000	100.000	100.000	100.000	100.000	100.000	100.000	96.152	85.609	71.810	58.140

An algebraic transformation yields

$$1 - p_i = \frac{p_{\mathrm{dl}}(i) - p_{\mathrm{dl}}(i-1)}{1 - p_{\mathrm{dl}}(i-1)}.$$

and

$$p_i = \frac{1 - p_{\mathrm{dl}}(i)}{1 - p_{\mathrm{dl}}(i-1)}.$$

Therefore, we can calculate the transition probabilities directly from the data loss probabilities $p_{\mathrm{dl}}(f)$ given a number f of disk failures.

V. RELIABILITY COMPARISONS

We can use the Markov models obtained to compare the reliability of various disk array organizations. Disk array organizations are complex and there are many more failure modes than just individual disk failure, but these additional failure modes do not depend on the organization of the data in the array.

We first calculated Mean Time To Data Loss (MTTDL) in years for an average disk life span of $\lambda = 100,000$ hours. Figure 6 gives the absolute number and Table VI compares the ratio of the MTTDL of a square organization with 64 data and 16 parity disks, a complete organization with 36 data and nine parity disks, and one, two, three, four, and eight stripes of a RAID Level 6 organization with 8 data disks and 2 parity disks per stripe. Raw MTTDL numbers have little meaning in isolation since corrosion and natural catastophies will do away with any disk array within a few millenia, but still allow comparisons among organizations.

Two results jump out: First, the complete design is almost as good as a single RAID Level 6 stripe, even though it contains 4.5 times the data. Second, the square layout is even more reliable, even though it contains 8 times the number of data. We can explain this in terms of our graph visualization by noting that the bipartite graph does not contain triangles consisting of edges.

Figure 6 allows us to see how we can trade a larger time between repairs for a more resilient organization of the data.

We now use our layout to evaluate the accuracy necessary in modeling disk array reliability. We first experimented with

TABLE VI

Relative MTTDL with regards to a single RAID Level 6 Stripe for variousmean time to repair (in days)

$MTTR$	Square (80)	Complete (45)	1×RL6 (10)	2×RL6 (20)	3×RL6 (30)	4×RL6 (40)	8×RL6 (80)
0.1	1.873	1.000	1.00	0.500	0.333	0.250	0.125
1.0	1.859	0.996	1.00	0.500	0.333	0.250	0.125
5.0	1.795	0.979	1.00	0.500	0.333	0.249	0.124
10.0	1.720	0.958	1.00	0.500	0.333	0.248	0.124

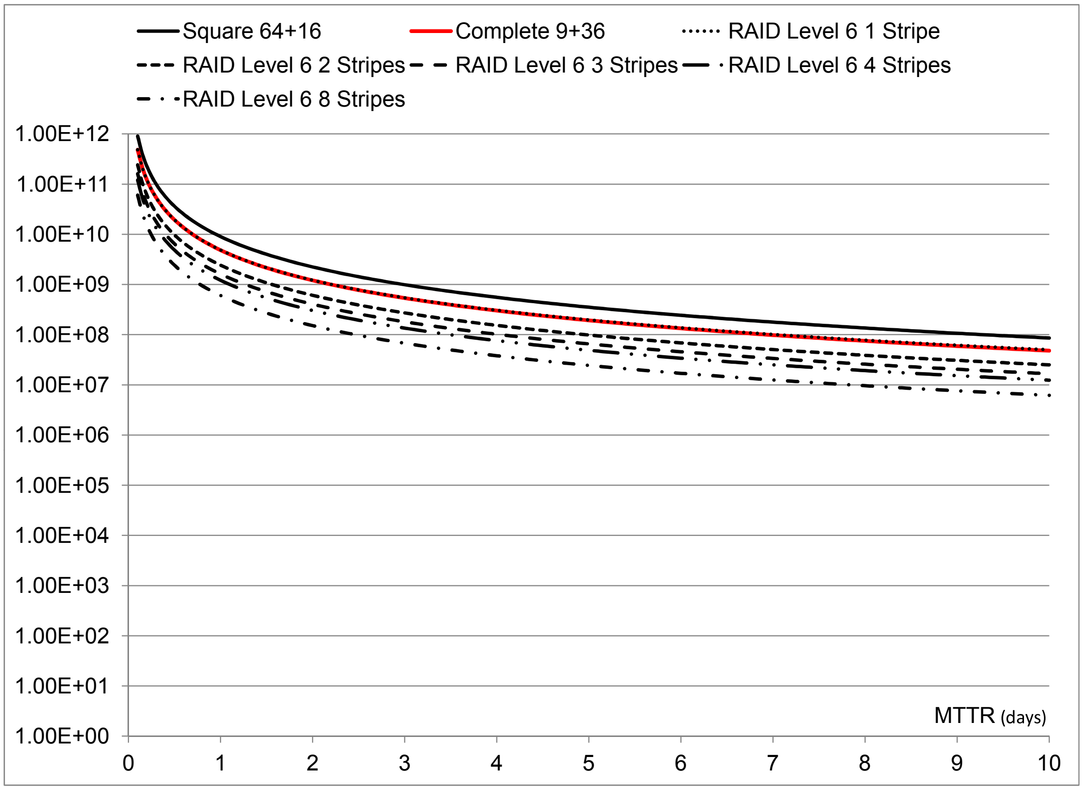

Fig. 6. MTTDL results in years for various disk organizations with 8 data disks per reliability stripe in dependence on the mean time to repair.

the complete organization with 9 parity and 36 data disks. We assumed that any triple, quadruple, quintuple, sextuple failure would already result in data loss and calculated the MTTDL accordingly. Our results represented in Figure 7 show that if we assume that quintuple failure lead to data loss, we already obtain MTTDL numbers barely distinguishable from the one using a completely accurate model.

Second, we calculated the five year survival rate of the various disk array organizations. We give the numbers in Table VII. We give the probability in nines. For example, a value of 5 means that the probability of data loss is 10^{-5}. It bears repeating that we only consider here data loss due to disk failure. Clearly, any disk array is exposed to a number of natural hazards such as fires, earthquakes, and even asteroid impact. Our survival rate do not measure these nor do they measure data loss due to faulty programming. What we can say is that the square and the complete disk array layout compare favorably with a RAID Level 6 organization.

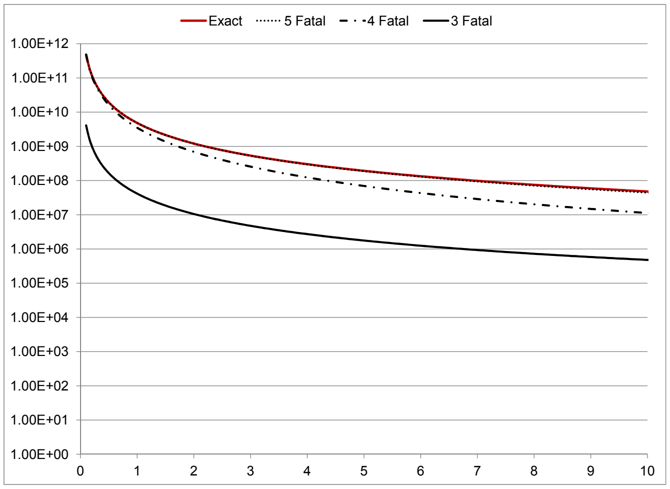

Fig. 7. MTTDL results in years for the complete organization using various simplifying assumptions.

TABLE VII

5 year survival rates in nines in dependence on the Mean Time To Repair (MTTR) in days.

MTTR	Square	Compl	1RL6	2RL6	3RL6	4RL6	5RL6
	(80)	(45)	(10)	(20)	(30)	(40)	(50)
0.5	5.914	5.643	5.645	5.344	5.167	5.043	4.946
1	5.310	5.040	5.043	4.742	4.566	4.441	4.344
1.5	4.955	4.687	4.692	4.391	4.215	4.090	3.993
2	4.703	4.436	4.443	4.142	3.966	3.841	3.744
2.5	4.507	4.241	4.250	3.949	3.772	3.648	3.551
3	4.346	4.082	4.092	3.791	3.615	3.490	3.393
3.5	4.209	3.947	3.959	3.658	3.482	3.357	3.260
4	4.091	3.831	3.844	3.543	3.366	3.242	3.145
4.5	3.986	3.727	3.742	3.441	3.265	3.140	3.043
5	3.892	3.635	3.651	3.350	3.174	3.049	2.952
5.5	3.807	3.551	3.569	3.268	3.092	2.967	2.870
6	3.729	3.475	3.494	3.193	3.017	2.892	2.796
6.5	3.657	3.405	3.426	3.125	2.949	2.824	2.727
7	3.591	3.339	3.362	3.061	2.885	2.760	2.663
8	3.470	3.222	3.247	2.947	2.771	2.646	2.549
9	3.363	3.118	3.147	2.846	2.670	2.545	2.448
10	3.267	3.025	3.057	2.756	2.580	2.455	2.358

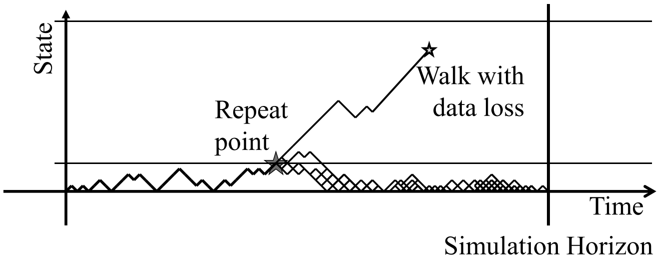

Fig. 8. Split importance sampling idea (after Görg et. al. [7]).

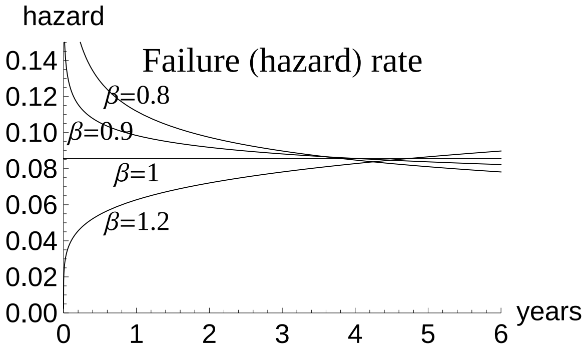

Fig. 9. Failure (hazard) rate of a Weibull distribution with mean 100000 hours and shape parameter $\beta \in \{0.8, 0.9, 1.0, 1.2\}$.

Since failures during a five-year economic lifespan are rare events, simple Monte-Carlo methods are unlikely to produce good variances for our estimates. In order to compare the impact of different failure distributions and repair time distribution, we used split importance sampling. The basic idea is simple and represented in Figure 8 [7]. We run many times a Monte-Carlo simulation simulating the state changes in the Markov model. Whenever we reach a certain *danger* state, we store the state of the simulation. From this *repeat point*, we then finish the simulation k times in order to count the number of times that we observe a data loss situation. Each dataloss counts as $1/k$ in the statistics. Therefore, our procedure is not biased. Because we deal with dangerous situations with greater intensity, we will observe many more data loss events compared to simple Monte-Carlo, which would just consider one trajectory out of the repeat point. Correspondingly, the variance is lower and our method yields better confidence intervals.

It is well known that disk arrays do not fail with constant failure rate, as was observed by Elerath and colleagues [4]–[6]. Sometimes, a drive model exhibits failure rates that are not easily described with any well-known distribution, but often, it is possible to fit a Weibull distribution with shape parameter between 0.8 and 1.2 to the failure number of the population. The *failure* or *hazard* rate decreases if the shape parameter is lower than 1 (infant mortality), is constant when it is 1 (in which case it is the exponential distribution), and increases if it is larger than 1 (aging), Figure 9.

We give the results of our simulations in Table VIII. We assumed constant repair time ($10h$, $25h$, $50h$ and $100h$) and compare the probabilities that a disk arrays suffers data loss during the first five years using a Weibull distribution with shapes 0.8, 0.9, 1.0 and 1.2. The first line in each block gives the probabilities. The second line gives the 99% confidence interval of our simulations. The third line gives the corresponding number of nines in reliability. We obtained this number generating 100 batches with 100000 simulation runs each. When the variation was exceptionally large, we added one hundred batches more for the RAID Level 6 variants. We can see that our variation reduction techniques is successfull where we can employ it, namely the complete and the square organization. The simulations for RAID Level 6 with 10 disks are faster, but show markedly higher variance. A comparison among the numbers reveals:

1) The complete and the square organization have survival rates that are very close to the rate of a much smaller array organized as a Level 6 RAID. This confirms our previous results even if we take disk infant mortality into account.
2) Disk infant mortality is important for the absolute numbers, but has no impact on the relative ranking. This confirms much research in the reliability of disk array organizations that simplifies by assuming exponential failure rate.

We should point out that our results do not include the pres-

TABLE VIII
FIVE YEAR FAILURE RATE OF RAID ORGANIZATIONS WITH DISKS WITH MTTF $100,000$ HOURS.

Shape	MTTR 100h	MTTR 50h	MTTR 25h	MTTR 10h
Raid Level 6: 8+2				
0.8	0.000466	0.000119	0.0000296	0.00000516
	±0.6 %	±2.3%	±4.9%	±12.0%
nines	3.332	3.925	4.528	5.287
0.9	0.000247	0.0000625	0.0000157	0.00000246
	±1.7%	±2.8%	±6.0%	±12.1%
nines	3.608	4.204	4.803	5.609
1.0	0.000151	0.0000377	0.00000979	0.00000165
	±1.5%	±4.3%	±7.4%	±20.9%
nines	3.820	4.424	5.009	5.783
1.2	0.0000718	0.0000171	0.00000424	0.00000066
	±1.3%	±6.2%	±12.6%	±32.4%
nines	4.144	4.768	5.373	6.180
Complete Organization: 36+9				
0.8	0.000484	0.000122	0.0000307	0.00000488
	±0.4%	±0.8%	±1.5%	±2.6%
nines	3.315	3.915	4.513	5.311
0.9	0.000256	0.0000641	0.0000161	0.00000257
	±0.5%	±1.0%	±2.3%	±5.0%
nines	3.592	4.193	4.793	5.591
1.0	0.000159	0.0000397	0.00000991	0.00000155
	±10.5%	±1.5%	±2.8%	±6.0%
nines	3.800	4.401	5.004	5.810
1.2	0.0000756	0.0000187	0.00000478	0.00000071
	±1.1%	±2.0%	±3.5%	±9.4%
nines	4.121	4.728	5.321	6.150
Square Organization: 64+16				
0.8	0.000423	0.0000994	0.0000245	0.00000399
	±1.8%	±2.7%	±0.9%	±1.8%
nines	3.374	4.002	4.611	5.399
0.9	0.000216	0.0000514	0.0000130	0.00000205
	±2.3%	±3.0%	±4.4%	±2.9%
nines	3.665	4.289	4.887	5.689
1.0	0.000127	0.0000315	0.00000784	0.00000128
	±2.2%	±0.8%	±1.3%	±3.4%
nines	3.895	4.501	5.106	5.893
1.2	0.0000612	0.0000151	0.00000374	0.000000585
	±0.7%	±1.3%	±2.2%	±5.4%
nines	4.214	4.821	5.428	6.233

ence of latent errors that might make a complete data recovery impossible. This is not a big flaw since the amount of data lost would be very small. As mentioned in the introduction, there are quite efficient methods to reduce the incidence of latent errors. Also, the probabilities are so low that other causes of data loss (human error, theft, fire, flooding, etc.) become important.

VI. CONCLUSIONS

We have presented a new disk organization, the complete organization, and compared its reliability to that of the square organization and that of one based on RAID Level 6 stripes. We have shown that the two two-dimensional organizations have a superior resilience against the effects of disk drive failures. A square organization with 64 data disks organized in reliability stripes of eight data disks has slightly better and a complete organization with 35 data disks organized with the same number of disks in a reliability stripe has slightly worse resilience then an organization based on RAID Level 6 with only 8 data disks, while offering the same storage overhead of 20%.

We have calculated mean time to failure values for the three disk organizations and shown that previous ad hoc estimates underestimated their reliability by assuming that all triple or all quadruple disk failures were fatal, but also shown that it is not necessary to take all failure combinations into account.

We have introduced an ad-hoc variance reducing method for the simulation of disk array reliability, which has proven itself effective.

REFERENCES

[1] I. Corderi, T. Schwarz, A. Amer, D. Long, and J. Pâris, “Self-adjusting two-failure tolerant disk arrays,” in *Petascale Data Storage Workshop (PDSW), 2010 5th*. IEEE, 2010, pp. 1–5.

[2] A. Dholakia, E. Eleftheriou, X. Hu, I. Iliadis, J. Menon, and K. Rao, “A new intra-disk redundancy scheme for high-reliability RAID storage systems in the presence of unrecoverable errors,” *ACM Transactions on Storage*, vol. 4, no. 1, pp. 1–42, May 2008.

[3] J. Elerath, “Hard-disk drives: The good, the bad, and the ugly,” *Communications of the ACM*, vol. 52, no. 6, pp. 38–45, 2009.

[4] ——, “Specifying reliability in the disk drive industry: No more MTBF’s,” in *Proceedings of the Annual Reliability and Maintainability Symposium*. IEEE, 2000, pp. 194–199.

[5] J. Elerath and M. Pecht, “Enhanced reliability modeling of raid storage systems,” in *37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, 2007, pp. 175–184.

[6] J. Elerath and S. Shah, “Server class disk drives: how reliable are they?” in *Annual Symposium on Reliability and Maintainability*. IEEE, 2005, pp. 151–156.

[7] C. Görg, E. Lamers, O. Fuß, and P. Heegaard, “Rare event simulation,” in *Modeling and Simulation Environment for Satellite and Terrestrial Communications Networks*, vol. 645. Kluwer International Series in Engineering and Computer Science, Springer, 2002, pp. 365–396.

[8] H. Gropp, “Configurations,” in *The CRC Handbook of Combinatorial Designs*, C. Cobourn and J. Dinitz, Eds. CRC Press, 1996.

[9] L. Hellerstein, G. Gibson, R. Karp, R. Katz, and D. Patterson, “Coding techniques for handling failures in large disk arrays,” *Algorithmica*, vol. 12, no. 2, pp. 182–208, 1994.

[10] I. Iliadis, R. Haas, X. Hu, and E. Eleftheriou, “Disk scrubbing versus intra-disk redundancy for high-reliability RAID storage systems,” in *Proceedings of the 2008 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*. ACM, 2008, pp. 241–252.

[11] Z. Jie, W. Gang, L. Xiaogugang, and L. Jing, “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*, vol. 26, no. 10, pp. 1379–1386, 2003.

[12] J. Pâris, A. Amer, and T. Schwarz, “Low-redundancy two-dimensional RAID arrays,” in *Computing, Networking and Communications (ICNC), 2012 International Conference on*. IEEE, 2012, pp. 507–511.

[13] J. Pâris, T. Schwarz, and D. Long, “Self-adaptive archival storage systems,” in *Proc. 26th International Performance of Computers and Communication Conference*, 2007, pp. 246–253.

[14] T. Schwarz, “Reliability and performance of disk arrays,” Ph.D. dissertation, University of California, San Diego, 1994.

[15] T. Schwarz, Q. Xin, E. Miller, D. Long, A. Hospodor, and S. Ng, “Disk scrubbing in large archival storage systems,” in *Proceedings of the IEEE 12th Annual International Symposium onn Modeling, Analysis, and Simulation of Computer and Telecommunications Systems*. IEEE, 2004, pp. 409–418.

[16] H. Yokota, “Dr-nets: data-reconstruction networks for highly reliable parallel-disk systems,” *ACM SIGARCH Computer Architecture News*, vol. 22, no. 4, pp. 41–46, 1994.

[17] H. Yokota and Y. Mimatsu, “A scalable disk system with data reconstruction functions,” in *Input/Output in Parallel and Distributed Computer Systems*. Springer, 1996, pp. 353–372.