RJ10358 (A0509-008) September 20, 2005
Computer Science

IBM Research Report

Reliability for Networked Storage Nodes

KK Rao, James L. Hafner, Richard A. Golding
IBM Research Division
Almaden Research Center
650 Harry Road
San Jose, CA 95120-6099

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

Reliability for Networked Storage Nodes

Abstract

High-end enterprise storage has traditionally consisted of monolithic systems with customized hardware, multiple redundant components and paths, and no single point of failure. Distributed storage systems realized through networked storage nodes offer several advantages over monolithic systems such as lower cost and increased scalability. In order to achieve reliability goals associated with enterprise-class storage systems, redundancy will have to be distributed across the collection of nodes to tolerate node and drive failures. In this paper, we present alternatives for distributing this redundancy, and models to determine the reliability of such systems. We specify a reliability target and determine the configurations that meet this target. Further, we perform sensitivity analyses where selected parameters are varied to observe their effect on reliability.

1 Introduction

High-end enterprise storage systems currently deployed in production environments have traditionally been monolithic systems – so-called 'big iron' with several symmetrical multiprocessors, multiple internal fabrics, large cache memories and no single point of failure. These systems are expensive – requiring customized hardware and multiple redundant components and paths to ensure that there is no single point of failure. In contrast, achieving scalability through distributed storage systems is becoming increasing popular in research and development, and, to some extent, in commercial deployments. A significant aspect of distributed systems is the ability to use common building blocks across a wide range of storage requirements: from a few terabytes to the scale of petabytes. This translates into several advantages: lower cost due to economies of scale, reduced number of inventory types, commonality of software across the product line, and so on.

The distributed storage system in this paper is modeled after the Collective Intelligent Bricks project in IBM Research [5]. The storage system consists of several bricks where each brick or node is a sealed unit consisting of a controller, power supply, networking interfaces and disk drives. Several components in the node represent single points of failure. In order to achieve reliability goals associated with high-end enterprise-class storage systems, redundancy has to be distributed across the collection of nodes to tolerate drive and node failure. In this paper, we will model the reliability of such a system and look at the alternatives for distributing redundancy between the nodes in order to meet reliability goals of large-scale enterprise systems.

The goal of this paper is to view redundancy requirements from a storage viewpoint. We assume that there is enough redundancy in switches and links so that reliability is limited by storage nodes and drives; that is, the interconnect fabric and topology is not a constraining factor in determining the overall reliability of the system. This is typically the case with [5].

We will describe the different configurations for achieving reliability in distributed storage systems, in Section 3. In Section 4, we will describe the models used to obtain reliability for these configurations. The implication of distributing data across such a system and its impact on reliability is presented in Section 5. Section 6 will present a baseline reliability analysis. We will analyze the sensitivity of the reliability of some of the configurations to several parameters, in Section 7.

2 Related Work

Trivedi [6] covers reliability analysis and in particular, the use of continuous-time Markov chains with absorbing states for crash failures. The modeling and analyses presented in this paper are based on this work. Xin *et al.* [7] present reliability for large distributed systems but do not consider node failures. Also, while uncorrectable sector errors are dealt with through a scheme of signatures, the reliability improvements through the use of this scheme are not characterized. Snappy Disk and Petal [4] represent shared-disk, shared-metadata systems and partitioned-disk, partitioned-metadata systems respectively. The availability analysis presented in this paper is intended only to gain insights into the factors affecting availability rather than to derive accurate predictions. Frølund *et al.* [2] describe an erasure coding algorithm in the context of a distributed storage system composed of inexpensive bricks, and Goodson *et al.* [3] describe erasure-coded storage that tolerates Byzantine failures. Both these

papers focus on the algorithms for erasure coding for distributed storage nodes and do not address the reliability analysis needed to ensure that erasure coded distributed storage will meet required reliability goals.

3 Redundancy Configurations

As mentioned earlier, a node consists of a controller card, network interfaces, a collection of disk drives and associated power supplies. Apart from the disk drives and the network interfaces, all other major components are not duplicated. Thus, the node is inherently unreliable as the failure of any one of these components will result in node failure. Therefore, in order to build a highly reliable storage system out of a collection of such nodes, redundancy will have to be distributed through the collection.

We will look at two dimensions to realizing redundancy in the collection of nodes: redundancy within nodes to tolerate internal drive failures, and redundancy across nodes to tolerate entire node failures. Within the nodes, we will employ three possible configurations: no internal RAID, RAID 5 and RAID 6, which will tolerate 0, 1 and 2 drive failures respectively. We will achieve redundancy across nodes by applying three types of erasure codes between them: codes that can tolerate 1, 2 and 3 node failures respectively. The three node configurations and three erasure code types between nodes yield a total of 9 combinations between them.

We assume that the nodes in this system are enclosed entities that are not amenable to service actions. This implies a fail-in-place philosophy where failed components within a node are not replaced. Specifically, in the case of failure of one or more disk drives within a node, the node will continue to operate with a reduced set of disks until either all disks fail or some other critical component fails rendering the node unusable. For the case of nodes with internal RAID (RAID 5 or RAID 6) we will assume that on a drive failure, data is re-striped removing the failed drive from the array, thereby restoring redundancy at the end of this operation. The resulting loss in capacity is adjusted against the spare capacity, as described below.

The fail-in-place service model implies that, initially, storage capacity is over-provisioned so that loss in capacity with subsequent failures can be tolerated. The over-provisioned storage capacity is either sufficient to deal with expected failures over the operational life of the installation, or spare nodes are added at appropriate times – e.g. when overall capacity utilization increases above predetermined thresholds.

4 Reliability Models

In this reliability analysis, we are primarily interested in preventing data loss. Consequently, to compare redundancy configurations, we use the expected number of data loss events per unit time as a measure of reliability. We believe the expected number of data loss events per unit time is a metric that is easier to comprehend and relate to than the more traditional Mean Time to Data Loss – MTTDL. We will use Markov models to determine MTTDL, and use it to obtain the expected number of data loss events per year.

We look at three types of failures that lead to data loss: an uncorrectable read error from a disk drive, a failure of a disk drive and a failure of a node. A data loss event occurs when the above failures occur in a combination that cannot be handled by the data protection scheme used in the system. For example, a controller with a RAID 5 array can tolerate a single failure (a disk failure or an uncorrectable read error). When a drive in the RAID 5 array fails and the array is rebuilding to a spare or a replacement drive, if either a second drive fails or an uncorrectable read error occurs on any of the remaining drives, this results in a data loss event. Clearly, the failure of the second drive results in data loss of a much larger scale than the uncorrectable read error, but either failure results in data loss of some magnitude.

With respect to an uncorrectable read error, we will assume it can result in a data loss event only if the array is in a critical state and cannot tolerate any further errors. We believe this is a reasonable assumption because as long as the array has not lost any drive, the recovery from an uncorrectable read error just requires reading from the remaining drives and regenerating the data item that encountered the read error. The conditions under which this recovery can fail are 1) if another element in the same stripe encounters an uncorrectable read error, or 2) if another drive fails during this recovery. We believe that both these conditions are extremely low probability occurrences and can be ignored.

To describe our modeling methodology, we illustrate the technique for a RAID 5 disk array. Figure 1 shows the Markov model for a RAID 5 array with mean time to failure of the disk drives $MTTF_d$ and mean time to repair (rebuild) a drive failure $MTTR_d$.

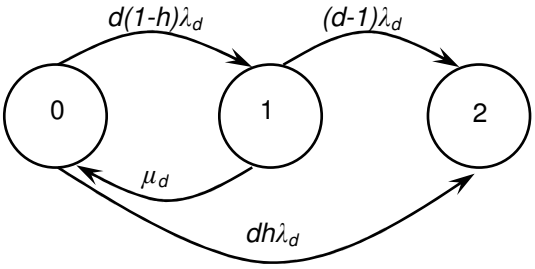

Figure 1: Markov Model for a RAID 5 array

State 0 is when the array is fully operational. State 1 corresponds to a drive failure that will not experience an uncorrectable error during the rebuild. State 2 represents a data loss state – either due to a second drive failure or due to an uncorrectable error during rebuild. The parameters are:

d =	number of drives in the array
λ_d =	drive failure rate = $1/MTTF_d$
μ_d =	drive rebuild rate = $1/MTTR_d$
h =	probability of an uncorrectable error during rebuild
=	$(d-1) \bullet C \bullet HER$
C =	drive capacity
HER =	disk hard error rate expressed in hard errors per number of bytes read

The methodology to solve a Markov model with absorbing states is described in [6].

Typically, $\mu_d >> \lambda_d$. Solving this model for *MTTDL* gives

$$MTTDL = \frac{(2d-1-dh)\lambda_d + \mu_d}{d(d-1)\lambda_d^2 + d\lambda_d\mu_d h}$$

$$\approx \frac{\mu_d}{d(d-1)\lambda_d^2 + d(d-1)\lambda_d\mu_d \bullet C \bullet HER}$$

4.1 Node Set and Redundancy Set

We introduce the concepts of node set and redundancy set for a storage system made up of networked storage nodes as shown in Figure 2. Data "objects" are stored across multiple nodes in such a system in order to meet requirements such as performance and reliability (the focus of this paper). Each data object constitutes exactly one 'stripe' of data – that is, the redundancy elements (parity) can be computed entirely from this data. For a given data object, the set of nodes that contain the data and its corresponding redundancy (parity) elements constitutes a redundancy set. The node set is the set of all the nodes in the storage system. We assume that data is evenly distributed across all the nodes in the storage system. Thus, each node has one or more redundancy set relationships with every other node in the node set. The total number of redundancy sets of size R in a node set of size N is given by: $\binom{N}{R}$

The even distribution of data implies that the failure domain is the entire node set and not just individual redundancy sets. For example, in a redundancy scheme that tolerates only a single failure, when such a failure has occurred and is being recovered from, a failure of any second node in the node set will result in data loss.

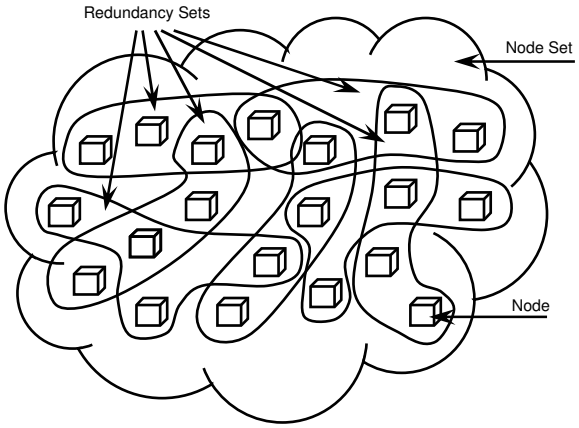

Figure 2: Node Sets and Redundancy Sets

We will present the modeling for systems where the nodes have internal RAID in section 4.2, and the modeling for nodes without internal RAID in section 4.3.

4.2 Nodes with Internal RAID

For a system in which the nodes have internal RAID, we use hierarchical Markov models to obtain *MTTDL*. We represent the RAID array internal to a node in a Markov model and obtain array failure rates from it. We then use these failure rates in a higher level Markov model representing the redundancy arrangement between nodes. It should be noted that, as we assume that the nodes are not amenable to service actions, and that on a drive failure, the array is re-striped removing the failed drive from the array, the μ_d term as depicted in Figure 1 is the array re-stripe rate and not the array rebuild rate. We already obtained the *MTTDL* for a RAID 5 array as:

$$MTTDL \approx \frac{\mu_d}{d(d-1)\lambda_d^2 + d(d-1)\lambda_d\mu_d \bullet C \bullet HER}$$

We define array failure as the failure of disk drives beyond the fault tolerance provided by the RAID scheme. From the above, we obtain λ_D, the rate of array failure and λ_S, the rate of a sector error during a re-stripe. These are:

$$\lambda_D(RAID5) \approx \frac{d(d-1)\lambda_d^2}{\mu_d}$$

$$\lambda_S(RAID5) \approx d(d-1)\lambda_d \bullet C \bullet HER$$

Figure 4 shows the Markov model for a RAID 6 array.

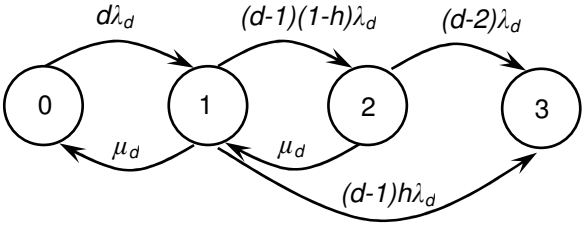

Figure 4: Markov Model for a RAID 6 array

State 0 is when the array is fully operational; state 1 is when a single drive has failed; state 2 corresponds to a second drive failure that will not experience an uncorrectable read error during rebuild; and state 3 represents a data loss state, either due to triple drive failure or an uncorrectable error when rebuilding with 2 drives failed. Solving this model for *MTTDL* gives

$$MTTDL \approx \frac{\mu_d^2}{d(d-1)(d-2)\lambda_d^3 + d(d-1)(d-2)\lambda_d^2\mu_d \bullet C \bullet HER}$$

Correspondingly, we obtain λ_D and λ_S as

$$\lambda_D(RAID6) \approx \frac{d(d-1)(d-2)\lambda_d^3}{\mu_d^2}$$

$$\lambda_S(RAID6) \approx \frac{d(d-1)(d-2)\lambda_d^2 \bullet C \bullet HER}{\mu_d}$$

We will use these rates in the higher level model for the erasure codes between nodes. Figure 5 shows the Markov model for nodes with internal RAID (either RAID 5 or RAID 6) and a redundancy arrangement with a fault tolerance of 1 between nodes.

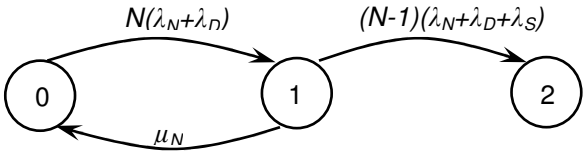

Figure 5: Markov Model for Fault Tolerance 1; Nodes with Internal RAID

Here N is the number of nodes in the node set, λ_N is the node failure rate and μ_N is the node rebuild rate. The array failure rate, λ_D, and the rate of sector error during a re-stripe, λ_S, correspond to the internal RAID in the nodes.

State 0 is when the storage system is fully operational. The system transitions to state 1 when either a node fails or a node experiences an array failure. In this state, the data of this node is rebuilt on the remaining nodes in the node set. (This is described in Section 5). State 2 represents the data loss state caused by a second node or array failure or a sector error during an internal RAID re-stripe while the node rebuild is in progress.

The *MTTDL* for this scheme (internal RAID, node fault tolerance 1) is given by:

$$MTTDL_{IR,NFT1} = \frac{\mu_N + (2N-1)(\lambda_N + \lambda_D) + (N-1)\lambda_S}{N(N-1)(\lambda_N + \lambda_D)(\lambda_N + \lambda_D + \lambda_S)} \approx \frac{\mu_N}{N(N-1)(\lambda_N + \lambda_D)(\lambda_N + \lambda_D + \lambda_S)}$$

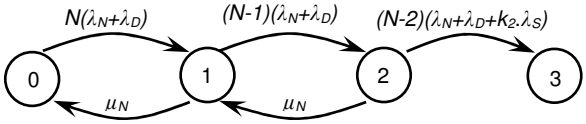

Figure 6: Markov Model for Fault Tolerance 2; Nodes with Internal RAID

Figure 6 shows the Markov model for nodes with internal RAID and an erasure code with a fault tolerance of 2 between nodes. As can be seen above, this scheme tolerates two failures; a third failure during the node rebuild operation results in a data loss event, State 3. We will explain the factor k_2 (and corresponding k_3 below) in section 5.2.1.

The *MTTDL* for this scheme (internal RAID, node fault tolerance 2) is:

$$MTTDL_{IR,NFT2} \approx \frac{\mu_N^2}{N(N-1)(N-2)(\lambda_N+\lambda_D)^2(\lambda_N+\lambda_D+k_2 \bullet \lambda_S)}$$

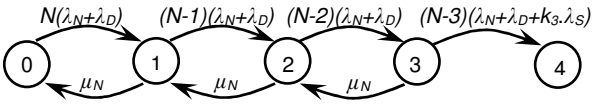

Figure 7: Markov Model for Fault Tolerance 3; Nodes with Internal RAID

Figure 7 shows the Markov model for nodes with internal RAID and an erasure code with a fault tolerance of 3 between nodes. This scheme tolerates three failures; a fourth failure during the node rebuild operation results in a data loss event, State 4.

The *MTTDL* for this scheme (internal RAID, node fault tolerance 3) is:

$$MTTDL_{IR,NFT3} \approx \frac{\mu_N^3}{N(N-1)(N-2)(N-3)(\lambda_N+\lambda_D)^3(\lambda_N+\lambda_D+k_3 \bullet \lambda_S)}$$

4.3 Nodes without Internal RAID

In configurations for nodes without internal RAID, individual drives within each node are used to realize the erasure code between nodes. We assume that no more than one drive per node is used in each redundancy set, that is, each block of a data stripe is on a different node; thus, each node failure causes only a single erasure on each redundancy set.

Figure 8 shows the Markov model for nodes without internal RAID and an erasure code of fault tolerance 1 between nodes.

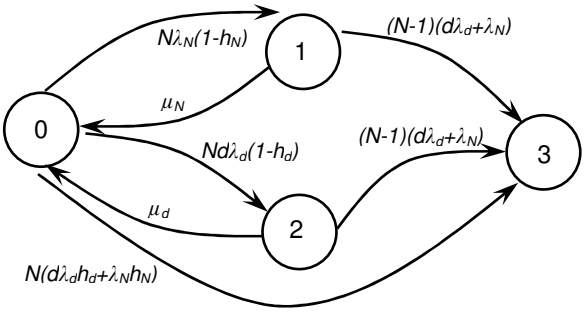

Figure 8: Markov Model for Fault Tolerance 1; Nodes without Internal RAID

Although there are only a few new parameters used in the above model, we list all the parameters:

N =	node set size
d =	drives per node
λ_N =	node failure rate
λ_d =	drive failure rate
μ_N =	node rebuild rate
μ_d =	drive rebuild rate
h_N =	probability of an uncorrectable error during node rebuild
=	$d \bullet (R-1) \bullet C \bullet HER = d \bullet h$
h_d =	probability of an uncorrectable error during drive rebuild
=	$(R-1) \bullet C \bullet HER = h$
R =	redundancy set size
C =	drive capacity
HER =	disk hard error rate expressed in hard errors per number of bytes read

State 0 is when the system is fully operational. State 1 corresponds to a node failure that will not experience an uncorrectable error during node rebuild. State 2 corresponds to a drive failure that will not experience an uncorrectable error during drive rebuild. State 3 represents a data loss state – either due to a second node or drive failure or due to an uncorrectable error during rebuild.

The *MTTDL* for this scheme (no internal RAID, node fault tolerance 1) is:

$$MTTDL_{NIR,NFT1} \approx \frac{\mu_d \mu_N}{N(N-1)(\lambda_N+d\lambda_d)(\mu_d\lambda_N+d\mu_N\lambda_D)+Ndh\mu_d\mu_N(\lambda_d+\lambda_N)}$$

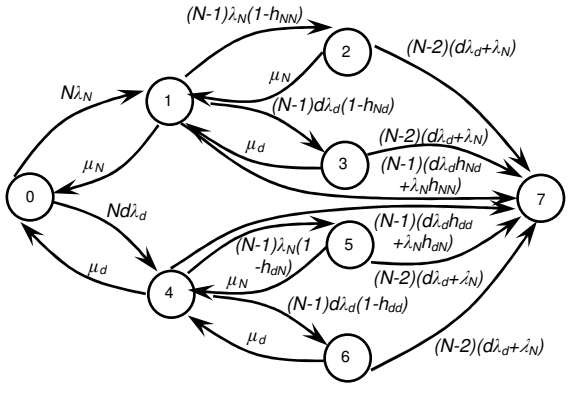

Figure 9: Markov Model for Fault Tolerance 2; Nodes without Internal RAID

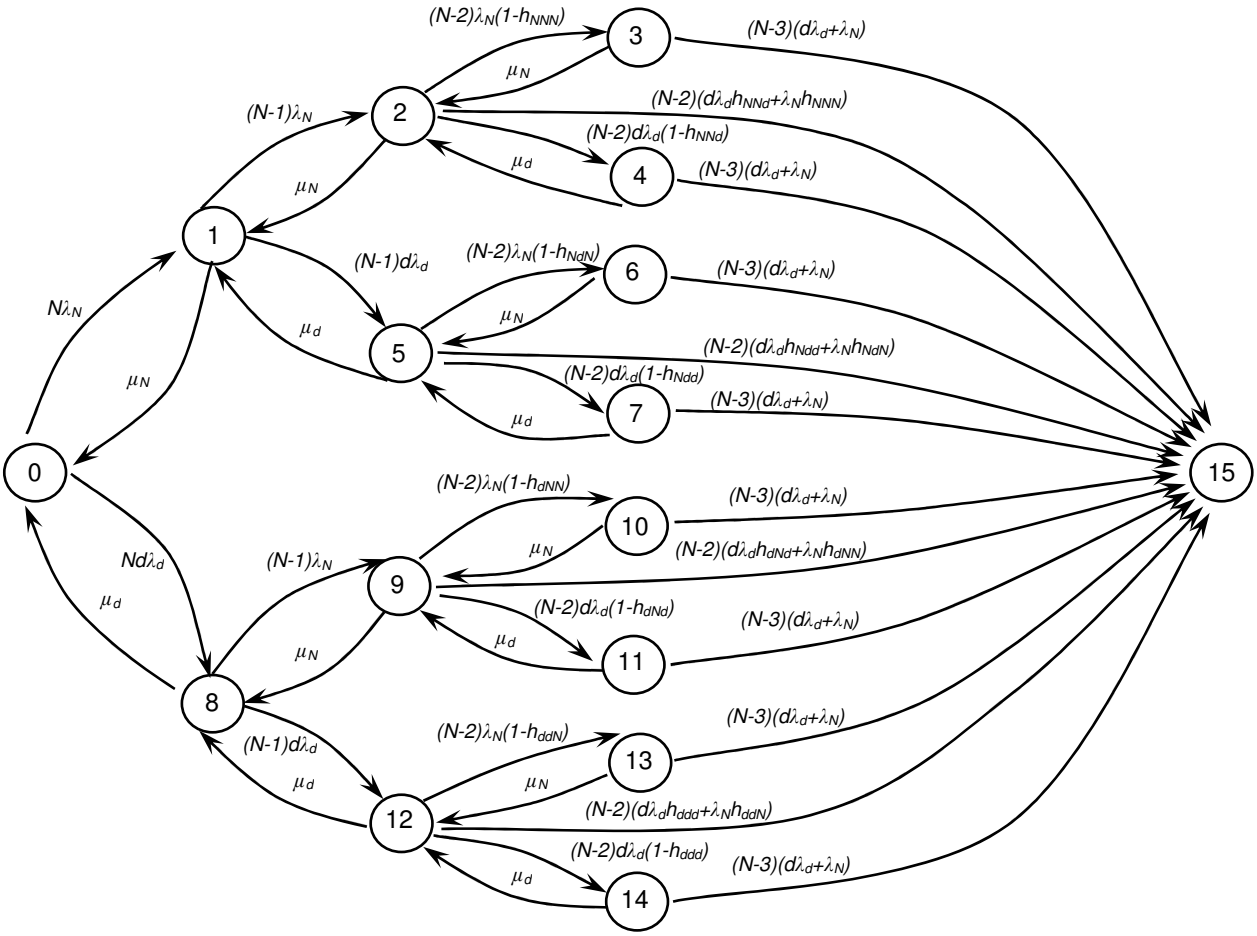

Figure 10: Markov Model for Fault Tolerance 3; Nodes without Internal RAID

Figure 9 shows the Markov model for nodes without internal RAID and an erasure code of fault tolerance 2 between nodes. The h_{xy} parameters are probabilities of encountering an uncorrectable error during a second node or drive rebuild ($y = N$ or d respectively), after an initial node or drive failure ($x = N$ or d respectively). We will show how these parameters are determined in Section 5.2.2.

Figure 10 shows the Markov model for nodes without internal RAID and an erasure code of fault tolerance 3 between nodes. As can be seen, the Markov models for nodes without internal RAID become increasingly complex as the fault tolerance increases. This is because without internal RAID, a drive failure state is distinct from a node failure state and these states multiply as the fault tolerance increases. Consequently, using conventional techniques to obtain a parameterized closed form solution for these higher levels of fault tolerance is not practical. However, by comparing Figures 8, 9 and 10, we observe similarities. For instance, the state transitions in Figure 8 are represented in two subsets in Figure 9 – states 1, 2, 3 and 7; and states 4, 5, 6 and 7. Similarly, Figure 9 itself is represented in two subsets in Figure 10. From these observations, it can be seen that a recursive method can be developed to solve these Markov models. In the appendix, we describe a recursive method to obtain a closed form solution for nodes without internal RAID with arbitrary fault tolerance across nodes.

The MTTDL for the last two schemes will be shown in Section 5.2 following the explanation of the h parameters.

5 Implications of Distributed Data

5.1 Node Rebuild Time

We mentioned earlier that the fail-in-place service model implies that the set of nodes is over-provisioned with spare capacity to deal with subsequent failures that will result in a loss of usable

capacity. This model, coupled with the even distribution of data, implies that spare capacity is also evenly distributed among the nodes. Thus, when a node fails, the data on the failed node is rebuilt by all the remaining nodes, utilizing their spare capacity. Similarly, in configurations without internal RAID, when a drive fails, the data on the failed drive is rebuilt on all the remaining drives. This is not the case for nodes with internal RAID: a drive failure results in a re-striping operation, removing the failed drive from the array and restoring redundancy.

Rebuild time, and hence the rebuild rate, is a key component in the expressions for MTTDL. We will describe a model to determine rebuild time accurately. Our model of rebuild time is based on the amount of data that is transferred during a rebuild.

We assume that in a rebuild, the destination node receives all the required redundancy data and performs the necessary exclusive-OR (or equivalent) operations to generate the data it will write on its drive(s).

For a node set size of *N*, a redundancy set size of *R* and a fault tolerance of *t*, we express the amounts of data below in units of a node's worth of data. Note that this means that *R* nodes are involved in the rebuild of one lost data object.

- Amount of data rebuilt by each node $= \frac{1}{N-1}$
- Amount of data received by each node from other nodes to rebuild the above $= \frac{R-t}{N-1}$
- Total data received by all the $N-1$ nodes $= (N-1)\frac{R-t}{N-1} =$ total data sourced by all $N-1$ nodes
- Total data sourced by each node $= \frac{1}{N-1}(N-1)\frac{R-t}{N-1} = \frac{R-t}{N-1}$

The effective rebuild time will be the maximum time required to move data in and out of nodes, to and from disks, and through the interconnecting network, depending on where the bottleneck lies.

- Hence, total data in and out of a node $= 2 * \frac{R-t}{N-1}$
- Total data to and from the disks in a node $= \frac{R-t}{N-1} + \frac{1}{N-1}$
- The total data flowing in the interconnecting network $= R-t$

5.2 Scope of Sector Error

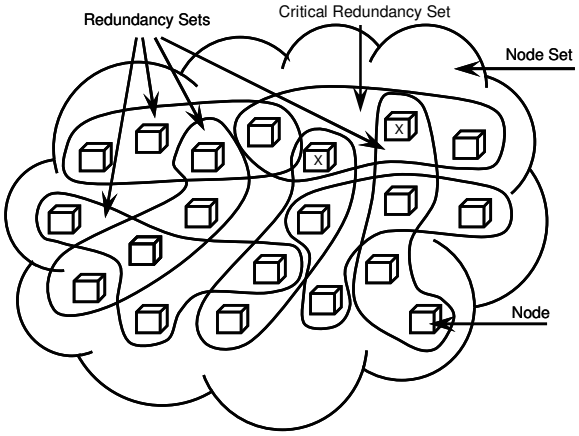

Figure 11: Critical Redundancy Sets

We stated earlier that we assume that an uncorrectable read error causes a data loss event only when the redundancy set is in a critical state. The even distribution of data across all the nodes implies that, for fault tolerance 2 or higher, when a redundancy set is critical, only a portion of a node's data (or drive's data in the case of no internal RAID) is critical.

This is illustrated in Figure 11. Let us assume that we have an erasure code of fault tolerance 2 between nodes and that the nodes have internal RAID. The X's indicated failed nodes. Each failed node is a part of two redundancy sets, one shared with the other failed node and one otherwise independent. However, only the shared set is critical; the other has lost one node but can tolerate a second loss.

5.2.1 Nodes with Internal RAID

The fraction of redundancy sets that are critical and hence, can contribute to a sector loss are represented in the k_2 and k_3 terms in the MTTDL expressions for internal RAID, fault tolerance 2 and 3 respectively. Each node is a part of $\binom{N-1}{R-1}$ redundancy sets. Thus,

$$k_2 = \frac{\binom{N-2}{R-2}}{\binom{N-1}{R-1}} = \frac{R-1}{N-1}, \text{ and}$$

$$MTTDL_{NIR,NFT\,2}$$

$$\approx \frac{\mu_d^2 \mu_N^2}{N(N-1)(N-2)(\lambda_N + d\lambda_d)(\mu_d \lambda_N + d\mu_N \lambda_D)^2 + N(R-1)(R-2) \bullet C \bullet HER \bullet d\mu_d \mu_N (\lambda_d + \lambda_N)(\mu_d \lambda_N + \mu_N \lambda_d)}$$

$$MTTDL_{NIR,NFT\,3}$$

$$\approx \frac{\mu_d^3 \mu_N^3}{N(N-1)(N-2)(N-3)(\lambda_N + d\lambda_d)(\mu_d \lambda_N + d\mu_N \lambda_D)^3 + N(R-1)(R-2)(R-3) \bullet C \bullet HER \bullet d\mu_d \mu_N (\lambda_d + \lambda_N)(\mu_d \lambda_N + \mu_N \lambda_d)^2}$$

Figure 12: MTTDL for No Internal RAID, Node Fault Tolerance 2 and 3

$$k_3 = \frac{\binom{N-3}{R-3}}{\binom{N-1}{R-1}} = \frac{(R-1)(R-2)}{(N-1)(N-2)}.$$

5.2.2 Nodes without Internal RAID

For nodes without internal RAID, we used *h*-with-subscript terms to represent probabilities of encountering uncorrectable sector errors during critical rebuilds. These probabilities depend on the amount of critical data that must be read for a rebuild operation, which in turn is derived from critical redundancy sets. Unlike nodes with internal RAID, redundancy sets may be critical because of combinations of node and drive failures.

The combinations and corresponding fractions of critical redundancy sets for fault tolerance 2 are:

- two nodes: $\frac{\binom{N-2}{R-2}}{\binom{N-1}{R-1}} = \frac{R-1}{N-1}$ of a node;
- two drives: $\frac{d^{R-2}\binom{N-2}{R-2}}{d^{R-1}\binom{N-1}{R-1}} = \frac{1}{d}\frac{R-1}{N-1}$ of a drive;
- and a drive and a node: $\frac{d^{R-1}\binom{N-2}{R-2}}{d^{R-1}\binom{N-1}{R-1}} = \frac{R-1}{N-1}$ of a drive.

The probability of encountering a hard error while rebuilding a drive if the entire drive is critical is $(R-2) \bullet C \bullet HER$.

Now, if $h = \frac{(R-1)(R-2)}{N-1} \bullet C \bullet HER$, then

$h_{NN} = dh$

$h_{Nd} = h_{dN} = h$ and

$h_{dd} = \frac{h}{d}$.

Similarly, the combinations and corresponding fractions of critical redundancy sets for fault tolerance 3 are:

- three nodes: $\frac{\binom{N-3}{R-3}}{\binom{N-1}{R-1}} = \frac{(R-1)(R-2)}{(N-1)(N-2)}$ of a node;
- two nodes and a drive: $\frac{d^{R-1}\binom{N-3}{R-3}}{d^{R-1}\binom{N-1}{R-1}} = \frac{(R-1)(R-2)}{(N-1)(N-2)}$ of a drive;
- two drives and a node: $\frac{d^{R-2}\binom{N-3}{R-3}}{d^{R-1}\binom{N-1}{R-1}} = \frac{1}{d}\frac{(R-1)(R-2)}{(N-1)(N-2)}$ of a drive;
- and three drives: $\frac{d^{R-3}\binom{N-3}{R-3}}{d^{R-1}\binom{N-1}{R-1}} = \frac{1}{d^2}\frac{(R-1)(R-2)}{(N-1)(N-2)}$ of a drive.

The probability of encountering a hard error while rebuilding a drive if the entire drive is critical is $(R-3) \bullet C \bullet HER$.

Now, if $h = \frac{(R-1)(R-2)(R-3)}{(N-1)(N-2)} \bullet C \bullet HER$, then

$h_{NNN} = dh$

$h_{NNd} = h_{NdN} = h_{dNN} = h$

$h_{Ndd} = h_{dNd} = h_{ddN} = \frac{h}{d}$ and

$h_{ddd} = \frac{h}{d^2}$.

We use these parameters to solve the Markov models and obtain the corresponding MTTDLs, which are shown in Figure 12. A general solution for arbitrary fault tolerance is described in the appendix.

6 Baseline Reliability

We use the closed form solutions for the MTTDL for the various configurations and determine baseline reliability using parameters defined below. We assume that desktop/ATA drives are used in the nodes.

$MTTF_N$ = node MTTF = 400,000 hours
$MTTF_d$ = drive MTTF = 300,000 hours
HER = drive hard error rate = 1 sector in 10^{14} bits read
C = drive capacity = 300 GB
Maximum drive throughput = 150 I/O operations/sec.
Drive sustained transfer rate (average) = 40 MB/sec.
N = node set size = 64
R = redundancy set size = 8
d = drives per node = 12
Re-stripe command size = 1 MB
Rebuild command size = 128 KB
Link speed = 10 Gbps (800 MB/sec. sustained)
Capacity utilization = 75%
Bandwidth utilization for rebuild, re-stripe = 10%

The link speed needs clarification. The rebuild performance depends on the total rate data can move in and out of the node over all links. We assume that nodes are physically sealed units shaped like cubes and are stacked together to build larger three-dimensional structures. Nodes communicate with adjacent nodes through links on each of their six surfaces. [1] has more information on effective bandwidth of such structures.

We specify the reliability target in terms of data loss events per PB-year. We view reliability from a manufacturer's perspective and choose a target that tracks the field population of such storage systems. We set a reliability target that a field population of 100 systems each with a petabyte of logical capacity will experience less than one data loss event in 5 years. This translates to less than 2 x 10^{-3} data loss events per PB-year.

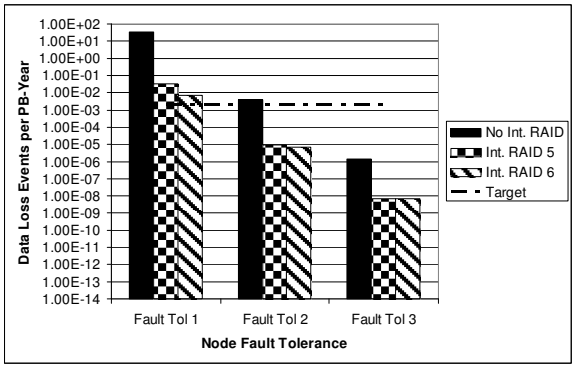

Figure 13: Baseline Comparison

Figure 13 shows a baseline comparison of the 9 configurations using the parameters defined above. We observe the following:

1. Configurations with node fault tolerance of 1 do not meet our reliability target.
2. There is no significant difference between internal RAID 5 and internal RAID 6 especially for fault tolerance 2 or higher. We will discuss why this is the case in Section 8.
3. At fault tolerance 3, the internal RAID configurations exceed the target by 5 orders of magnitude.

Based on the above observations, we will not consider configurations with fault tolerance of 1 between nodes. Also, for configurations with internal RAID, we will only use RAID 5 as RAID 6 does not provide any advantage. Further, we will not include the configuration at fault tolerance 3, internal RAID in the sensitivity analyses (item 3. above). This will leave us with three configurations for sensitivity analyses:

Fault Tolerance 2 without internal RAID,
Fault Tolerance 2 with internal RAID 5, and
Fault Tolerance 3 without internal RAID.

7 Sensitivity Analyses

We will perform sensitivity analyses of the reliability to the following parameters: drive MTTF, node

MTTF, rebuild block size, link speed, node set size, redundancy set size, and drives per node. As we vary these parameters one at a time, we will keep all the other parameters at their baseline level, except for drive and node MTTF. For the latter two, we will use two values, one at each end of a practical range as shown here:

Drive MTTF (hours): low 100,000; high 750,000;
Node MTTF (hours): low 100,000; high 1,000,000.

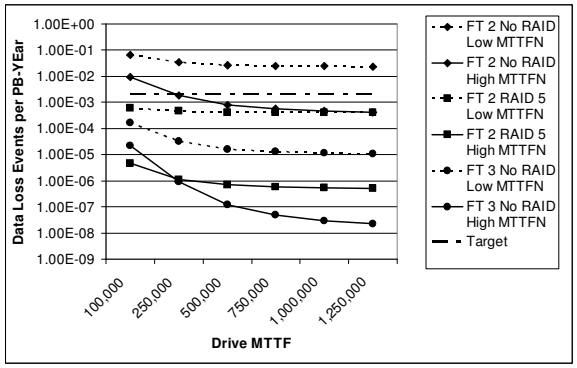

Figure 14: Sensitivity to Drive MTTF

Figure 14 shows the sensitivity to disk drive MTTF. We observe that the configuration at fault tolerance 2, no internal RAID does not meet the target at all for low node MTTF, and marginally meets it for high node MTTF. The other two configurations exceed the target – some more comfortably than the others – over the entire range. FT 2, Internal RAID 5 appears to be relatively insensitive to drive MTTF, especially for low node MTTF – clearly, it is limited by node MTTF and provides another view why RAID 6, which protects from a further drive failure, does not offer any advantage.

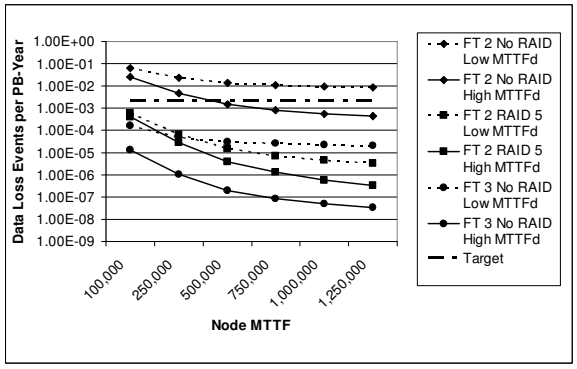

Figure 15: Sensitivity to Node MTTF

The sensitivity to node MTTF is shown in Figure 15. FT 2, Internal RAID 5 shows the most sensitivity to node MTTF and all three configurations show increased sensitivity with high drive MTTF. FT 2, No Internal RAID again does not meet the target for the most part.

The rebuild block size affects the node and the drive rebuild rate, μ_N and μ_d respectively. As we saw in Sections 4 and 5, these are key parameters for the MTTDL. From Figure 16, it can be seen that the rebuild block size affects the reliability significantly. FT2, No Internal RAID does not meet the target for low MTTF. The other two configurations meet the target if the rebuild block size is 64 KB or larger.

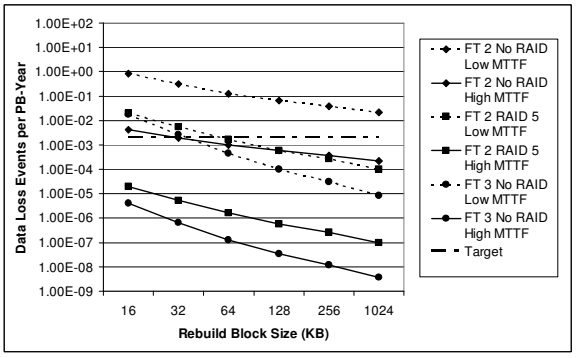

Figure 16: Sensitivity to Rebuild Block Size

The rebuild rate is determined by the slower of the data transfers – across the network between nodes or within a node to and from the disk drives. With the parameters as defined (12 drives per node, 150 I/O operations/second, and so on), the rebuild rate is constrained by the link speed up to around 3 Gb/s beyond which it is constrained by the disk drives.

This can be seen in Figure 17 which shows sensitivity to link speed at 3 points – 1, 5 and 10 Gb/s. There is no difference in reliability between the last two points.

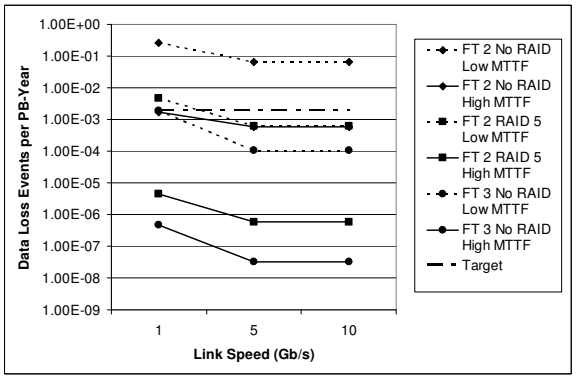

Figure 17: Sensitivity to Link Speed

We now look at sensitivity to the configurable parameters – node set size, redundancy set size and drives per node. Figure 18 shows the sensitivity to

node set size. As can be seen, FT 2, No Internal RAID shows some sensitivity to the node set size, but the other two configurations are relatively insensitive to it.

The sensitivity to redundancy set size is shown in Figure 19. It can be seen that all configurations appear to become less reliable as the redundancy set size increases, with about an order of magnitude difference between the extremes.

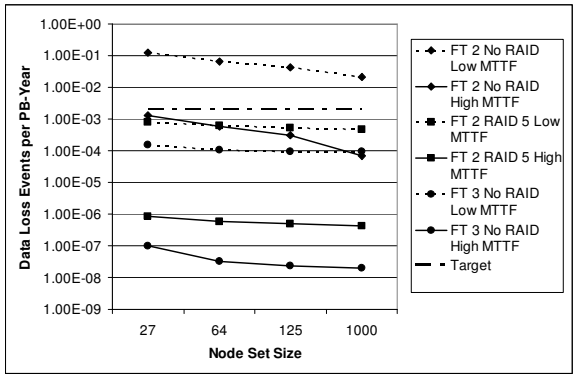

Figure 18: Sensitivity to Node Set Size

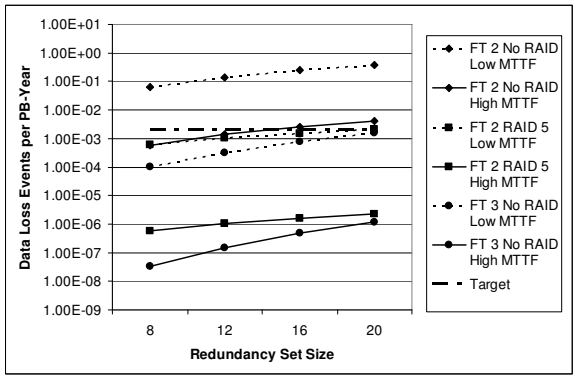

Figure 19: Sensitivity to Redundancy Set Size

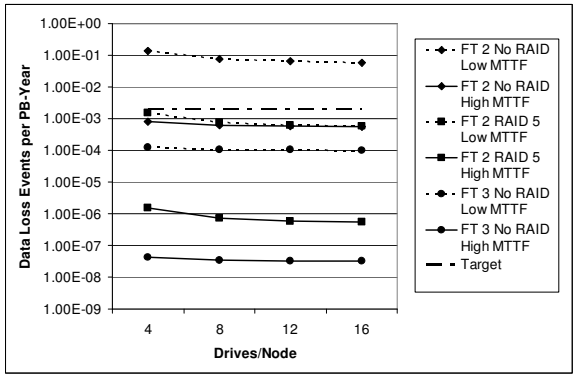

Figure 20: Sensitivity to Drives per Node

From Figure 20, it can be seen that there is very little sensitivity to the number of drives per node. It should be noted that we are measuring normalized reliability – data loss events per PB-Year. As a result, with some parameters such as drives per node, there is a cancellation effect. Increasing the number of drives in a node can result in decreased reliability per node – however, fewer such nodes will be required to yield a petabyte.

8 Discussion

The baseline reliability analysis in Section 6 showed that RAID 6 does not offer any advantage over RAID 5 when used internal to networked storage nodes. This is because the reliability of a networked storage system as a whole is affected by both drive and node failures. When RAID 5 is used internally, the effect of drive failures is considerably minimized such that the susceptibility to node failures becomes a dominant factor. Providing further tolerance to drive failures by using RAID 6 does not alleviate the susceptibility to node failures. It is interesting to note that we need to obtain a balance of protection against both drive and node failures – increasing the protection for one without correspondingly increasing it for the other does not result in an overall increase in reliability.

The sensitivity analyses in section 7 reveal interesting results. Firstly, we see that there is very little sensitivity to the configurable size parameters – node set size and drives per node and a little more pronounced sensitivity to redundancy set size. We alluded to the reason for the insensitivity to drives per node earlier. Similar arguments apply to the node set size. In the latter case, there is an additional factor. Even though increasing the node set size increases the size of the failure domain, the fraction of critical redundancy sets decreases.

We also see that the reliability is constrained by disk drive bandwidth rather than network bandwidth if the link speed is 3 Gb/s or higher, resulting in no change in reliability at higher link speeds. By using drive bandwidth more efficiently through the use of larger rebuild block sizes, we see significant improvements in reliability. In fact, the rebuild block size is a controllable parameter with the most significant impact on reliability.

In contrast, drive and node MTTF are not easily controllable. Industry experience has indicated that drive MTTF can vary significantly between batches of drives and the same can be expected of nodes.

The numbers we have used in the baseline analysis are conservatively realistic with the sensitivity analysis providing an insight into available headroom from a reliability perspective.

For the specific target we have chosen in this paper, it appears that either the [FT2, Internal RAID 5] or the [FT3, No Internal RAID] configurations meet the reliability requirement with the condition that the rebuild block size is at least 64 KB.

9 Conclusions

We have developed effective reliability models for networked storage nodes based on Markov chains. We deal with the complexity of solving large Markov models in two different ways – hierarchical models and recursive models. Using these methods, we are able to generate closed-form parametric solutions that have broad utility. We have chosen a specific reliability target in order to focus on a few redundancy configurations. However, the closed-form solutions we have presented may be used to determine redundancy configurations for a spectrum of reliability targets such as in systems that offer user-configurable goals.

We have also developed a model that utilizes basic parameters such as disk drive bandwidth and network link speed, to generate effective rebuild rates. System reliability, as we have seen, is impacted significantly by the rebuild rate; hence, obtaining a precise estimate using basic parameters ensures that the reliability results are accurate.

References

[1] C. Fleiner, D.R. Kenchammana Hosekote, R. Garner, and W. Wilcke. Quantitative Study of the Performance and Reliability of a Resilient 3-D Mesh-based Server. Technical Report RJ 10308, IBM Research, November 2003.

[2] S. Frølund, A. Merchant, Y. Saito, S. Spence, and A. Veitch. A Decentralized Algorithm for Erasure-Coded Virtual Disks. *Dependable Systems and Networks*, June 2004.

[3] G.R. Goodson, J.J. Wylie, G.R. Ganger, and M.K. Reiter. Efficient Byzantine-tolerant erasure-coded storage. *Dependable Systems and Networks*, June 2004.

[4] E.K. Lee, C.A. Thekkath, C. Whitaker, and J. Hogg. A Comparison of Two Distributed Disk Systems. Research Report 155, Digital Systems Research Center, April 1998.

[5] IBM Research. Collective Intelligent Bricks – Hardware. http://www.almaden.ibm.com/StorageSystems/autonomic_storage/CIB_Hardware/index.shtml

[6] K.S. Trivedi. Probability and Statistics with Reliability, Queuing, and Computer Science Applications. *Prentice-Hall*, 1982

[7] Q. Xin, E.L. Miller, T. Schwarz, D.D.E. Long, S.A. Brandt, and W. Litwin. Reliability Mechanisms for Very Large Storage Systems. *IEEE/ NASA Goddard Conference on Mass Storage Systems and Technologies*, April 2003.

Appendix: Recursive Solution to Reliability Models with No Internal RAID

In this section we outline the recursive methodology used to solve for the *MTTDL* in the case of no internal RAID with redundancy of arbitrary fault tolerance k across nodes. The results for k=1, 2, and 3 of Sections 4.3 and 5.2 are special cases. For a CTMC (see [6]) with state set S, absorbing states A, non-absorbing states $B = S - A$ and mean time spent in state $i \in B$ given by τ_i, the *MTTDL* is computed as

$$MTTDL = \sum_{i \in B} \tau_i \qquad \text{(A.1)}$$

The terms τ_i are can be computed as the solution to the system of equations

$$\tau_B Q_B = -\pi_B(0)$$

where $\tau_B = \langle ..., \tau_i, ... \rangle_{i \in B}$, $\pi_B(0)$ is the vector of initial probabilities for the states in B, and Q_B is the submatrix restricted to the non-absorbing states B of the infinitesimal generator matrix Q. The matrix Q is defined as follows: the off-diagonal entries are the transition rates for each pair of states in S (these are non-negative); the diagonal entries are defined so that the row sums of Q all equal zero (the diagonal entries are negative). In all our models, there is only one initial state (the first state in an enumeration of B) so that $\pi_B(0) = \langle 1,0,...,0 \rangle$. Consequently, we have $\tau_B = -\langle 1,0,...,0 \rangle Q_B^{-1}$ and

$$MTTDL = -\langle 1,0,...,0 \rangle Q_B^{-1} \langle 1,...,1 \rangle^t .$$

The vector on the right in this formula computes the sum in (A1). We let $R = -Q_B$ so that R has positive diagonal entries, non-positive off-diagonal entries and

$$MTTDL = \langle 1,0,...,0 \rangle R^{-1} \langle 1,...,1 \rangle^t \qquad \text{(A.2)}$$

We call R the absorption matrix for the model. Let $M(R)$ be the expression on the right hand side of (A.2). Recall the formula $R^{-1} = \text{adj}(R)/\det(R)$ where $\text{adj}(R)$ is the adjoint of R (the transpose of the matrix of determinants of all one-less dimension submatrices of R). Set

$$\text{Num}(R) = \langle 1,0,...,0\rangle \text{adj}(R) \langle 1,...,1\rangle^t$$

So that we have

$$M(R) = \text{Num}(R)/\det(R). \qquad \text{(A.3)}$$

("Num" is an abbreviation for numerator.) We also define the notation $\text{Sdet}(R) = $ upper left corner of $\text{adj}(R)$, that is, the determinant of the submatrix of R after removing the first row and first column. If $R = (r)$ is a scalar (1x1), then set $\text{Num}(R) = 1$, $\text{Sdet}(R) = 1$, and $\det(R) = r$, so that $M(R) = 1/r$. We use this notation and formulation later.

As we noted in Section 4.3, the CTMC for the no internal RAID model with fault tolerance k has a recursive structure. By a re-labeling, we can describe this recursion as follows.

First build the model as in Fig. 8 for fault tolerance $k = 1$. Re-label state "3" as "A" (the absorbing state), state "1" as "N" and state "2" as "d" (to indicate the type of failure that we model on the transition into these states). To create the model for general k from the model for $k-1$, do the following:

1. Make two copies of the model for fault tolerance $k-1$ (inductively). Each non-absorbing state has a label of length $k-1$ in the letters "0", "N", "d".
2. Merge the two absorbing states into one state "A".
3. Prefix each state label in the first copy with an "N" and in the second copy with a "d".
4. In the each copy, replace N by $N-1$ (and $N-1$ by $N-2$, etc.). In the first copy replace every subscript on each h with a new subscript prefixed by "N"; in the second copy prefix each h-subscript by "d".
5. Add a new root state with label all "0"s of length k. Set the rate from this new state to the root state of the first copy (labeled "$N0\ldots0$") to $N\lambda_N$, and back with μ_N; set the rate from this new state to the root state of the second copy (labeled "$d0\ldots0$") with rate $Nd\lambda_d$ and back with μ_d.

This completes the construction.

The general model is parameterized by N, $h^{(k)}$, μ_N, μ_d, λ_N, and $d\lambda_d$ where $h^{(k)} = \{h_\alpha : \alpha \in \{N,d\}^k\}$ (so the subscripts are all words of length k in the letters "N" and "d") and assume this is in reverse lexicographical order according to the subscripts. Generally, we will suppress the last four parameters as they are not dependent on what level we are in the recursive construction (only N and $h^{(k)}$ change as we see above). At times we suppress the dependence on N and $h^{(k)}$ as well for notational brevity.

When $k > 1$, there are no transitions from the root state to the absorbing state. When $k = 1$, there is a transition and it is determined by h_N and h_d (see Figure 8). When $k > 1$, the only transitions to the absorbing state come at the inner most level of the recursion. For every state with label containing only the letters "N" and "d", there is a transition to the absorbing state with rate $(N-k)(\lambda_N + d\lambda_d)$. For every state whose label is of the form $\alpha 0$ (where α contains only the letters "N" and "d"), there is a transition to the absorbing state with rate $(N-k+1)(\lambda_N h_{\alpha N} + d\lambda_d h_{\alpha d})$.

The construction (step 4) suggests the following notational operation for the sets $h^{(k)}$: for $x =$ "N" or "d", define the "dot" operation

$$h_x \circ h^{(k-1)} = \{h_{x\alpha} : \alpha \in \{N,d\}^{k-1}\}$$

so that

$$h^{(k)} = h_N \circ h^{(k-1)} \cup h_d \circ h^{(k-1)}.$$

Given this notation and construction, it is easy to see that the absorption matrix $R = R^{(k)}(N, h^{(k)})$ for the model of fault tolerance k has the form

$$R^{(k)} = \begin{pmatrix} r^{(k)} & -\mathbf{r}_N & -\mathbf{r}_d \\ -\boldsymbol{\mu}_N & R_N^{(k)} & 0 \\ -\boldsymbol{\mu}_d & 0 & R_d^{(k)} \end{pmatrix}$$

where $\boldsymbol{\mu}_N$ represents a vector of the form $\langle \mu_N, 0, .., 0\rangle$ (similarly for $\boldsymbol{\mu}_d$, $\mathbf{r}_N$ and $\mathbf{r}_d$), and for $k > 1$,

$$r_N = N\lambda_N$$
$$r_d = Nd\lambda_d$$
$$r^{(k)} = N(r_N + r_d)$$

since there is no transition from the root state (labeled with all zero word) to the absorbing state in this case.

If $k = 1$ then $r_N = N\lambda_N(1-h_N)$, $r_d = Nd\lambda_d(1-h_d)$ and $r^{(1)} = N(\lambda_N + d\lambda_d)$.

$$MTTDL(N,h^{(k)})) \approx \frac{(\mu_N\mu_d)^k}{N(N-1)\cdots(N-k+1)((N-k)(\lambda_N+d\lambda_d)L(\mu_d,\mu_N)^k+(\mu_N\mu_d)L_k(h^{(k)}))}$$

Figure A1: General form for MTTDL for *k* Fault Tolerance

The dimension of $R^{(k)}$ is $2^{k+1}-1$. The matrices $R_N^{(k)}$ and $R_d^{(k)}$ are of the same structural form as $R^{(k)}$. Let $U^{(k)}$ be the matrix of size 2^k-1 that is all zero except for a single one in the upper left corner. Then $R_N^{(k)}-\mu_N U^{(k)}$ is the absorption matrix for the $k-1$ level model with parameters N and $h^{(k-1)}$ replaced by $N-1$ and $h_N\circ h^{(k-1)}$, respectively. Similarly, $R_d^{(k)}-\mu_d U^{(k)}$ is the absorption matrix for the $k-1$ level model with parameters N and $h^{(k-1)}$ again replaced by $N-1$ and $h_d\circ h^{(k-1)}$, respectively. Symbolically, for $x=$ "N" or "d",

$$R_x^{(k)}(N,h^{(k)})-\mu_x U^{(k)} = R^{(k-1)}(N-1,h_x\circ h^{(k-1)}). \tag{A.4}$$

We now have a formal model of the recursive construction and the effect this recursive construction has on the absorption matrices and the parameters at each level.

From the definitions of adj and det and a straightforward calculation, it is not difficult to prove the following lemma:

Lemma. For $k\ge 1$,

$$\begin{aligned}\mathrm{Num}(R^{(k)}) &= \mathrm{Sdet}(R^{(k)}) + r_N\mathrm{Num}(R_N^{(k)})\det(R_d^{(k)}) \\ &\quad + r_d\det(R_N^{(k)})\mathrm{Num}(R_d^{(k)})\end{aligned}$$

and

$$\begin{aligned}\det(R^{(k)}) &= r^{(k)}\mathrm{Sdet}(R^{(k)}) \\ &\quad + r_N(\det(R_N^{(k)})-\mu_N\mathrm{Sdet}(R_N^{(k)}))\det(R_d^{(k)}) \\ &\quad + r_d\det(R_N^{(k)})(\det(R_d^{(k)})-\mu_d\mathrm{Sdet}(R_d^{(k)}))\end{aligned}$$

By (A.4), the term (with x = "N" or "d", and suppressing the N and $h^{(k)}$ on the left side)

$$\begin{aligned}&\det(R_x^{(k)})-\mu_x\mathrm{Sdet}(R_x^{(k)}) \\ &\quad = \det(R^{(k-1)}(N-1,h_x\circ h^{(k-1)})).\end{aligned} \tag{A.5}$$

and

$$\mathrm{Num}(R_x^{(k)}) = \mathrm{Num}(R^{(k-1)}(N-1,h_x\circ h^{(k-1)}))$$

as well. These formulas provide the basis for an inductive argument. We need some additional notation in order to state the result and assumptions (on relative size of parameters) in order to derive our approximation results.

Set $L(x,y)=x\lambda_N+yd\lambda_d$ so that $L(x,y)=(xr_N+yr_d)/N$ (on recalling that $r_N=N\lambda_N$ and $r_d=Nd\lambda_d$). Furthermore, for any ordered set $H^{(k)}$ of 2^k symbols, let $L_1(H^{(1)})=L(H_1,H_2)$ for $k=1$, and for $k>1$

$$L_k(H^{(k)})=L(\mu_d L_{k-1}(H_1),\mu_N L_{k-1}(H_2))$$

where $H^{(k)}=H_1\cup H_2$ and H_1 is the first 2^{k-1} elements of $H^{(k)}$ and H_2 is the last 2^{k-1} elements. So, for our special set $h^{(k)}$ we have

$$L_k(h^{(k)})=L(\mu_d L_{k-1}(h_N\circ h^{(k-1)}),\mu_N L_{k-1}(h_d\circ h^{(k-1)}))$$

and $L_1(h^{(1)})=L(h_N,h_d)$.

We can now state the general theorem:

Theorem: Assume $N(\lambda_N+d\lambda_d)$ is at least an order of magnitude smaller than both μ_N and μ_d. Then

$$\begin{aligned}&\det(R^{(k)}(N,h^{(k)})) \\ &\approx N(N-1)\cdots(N-k+1)(\mu_N\mu_d)^{2^k-k-1} \\ &\quad\cdot\left((N-k)(\lambda_N+d\lambda_d)L(\mu_d,\mu_N)^k+(\mu_N\mu_d)L_k(h^{(k)})\right)\end{aligned}$$

and

$$\mathrm{Num}(R^{(k)}(N,h^{(k)}))\approx(\mu_N\mu_d)^{2^k-1}.$$

From this and (A.3) we easily derive the approximation formula for *MTTDL* for the general model of fault tolerance k across nodes and no internal RAID as shown in Figure A1.

The proof of the theorem is a fairly straightforward induction, using the formula (A.5) and the Lemma. We leave out the details. The statements of *MTTDL* in Section 4.3 and 5.2 for k=1, 2 and 3 are easily seen to be special cases of this theorem, after replacing the parameters $h^{(k)}$ by their values as defined in those sections. In particular, we see that the numerator of the quotient is simply $(\mu_N\mu_d)^k$. The denominator contains a term $N(N-1)\cdots(N-k+1)$ and two (possibly) comparable terms depending on the relative orders of magnitude the parameters.