Highly Available and Heterogeneous Continuous Media Storage Systems

Roger Zimmermann and Shahram Ghandeharizadeh, *Member, IEEE*

***Abstract*—A number of recent technological trends have made data intensive applications such as continuous media (audio and video) servers a reality. These servers store and retrieve large volumes of data using magnetic disks. Servers consisting of multiple nodes and large arrays of heterogeneous disk drives have become a fact of life for several reasons. First, magnetic disks might fail. Failed disks are almost always replaced with newer disk models because the current technological trend for these devices is one of annual increase in both performance and storage capacity. Second, storage requirements are ever increasing, forcing servers to be scaled up progressively. In this study, we present a framework to enable parity-based data protection for heterogeneous storage systems and to compute their mean lifetime. We describe the tradeoffs associated with three alternative techniques: independent subservers, dependent subservers, and disk merging. The disk merging approach provides a solution for systems that require highly available secondary storage in environments that also necessitate maximum flexibility.**

***Index Terms*—Continuous media servers, fault tolerance, data protection, high availability, heterogeneous storage, magnetic disk replacement, streaming media servers.**

I. Introduction

APPLICATIONS that utilize digital continuous media, such as video and audio clips, require vast amounts of storage space [1]. Large archives may consists of hundreds, if not thousands of disks, to satisfy both the bandwidth and storage requirements of the working sets imposed by different applications. Although a single disk is fairly reliable, with a large number of disks, the aggregate rate of disk failures can be too high. At the time of this writing, the mean time to failure (MTTF) of a single disk is on the order of 1 000 000 h; this means that the MTTF of *some* disk in a 1000–disk system is on the order of 1000 h (approximately 42 days).

With those servers that assume a hierarchical storage structure, a disk failure may not result in an actual loss of data, because the entire database is tertiary resident [1]–[3]. The disks cache the most frequently accessed objects to minimize the number of references to the tertiary. Data redundancy at the disk level continues to be important because it is undesirable for a single disk failure to impact all the active displays. With no data redundancy and a disk failure, some requests are forced to retrieve the missing data from tertiary, which usually has a much lower bandwidth than the aggregate bandwidth of the disk subsystem. This would diminish the number of simultaneous displays that can be supported. Even with redundant data, multiple failures might force the system to terminate the service of some active displays. Consequently, we use the *mean time to service loss* (MTTSL) to quantify the fault resilience characteristics of the algorithms discussed in this paper. The terms used throughout our presentation and their definitions are listed in Table I.

A common technique to protect against both data and service loss is to add redundancy to the system, either by mirroring data or adding parity information [4], [5]. There is a vast body of literature analyzing techniques in support of homogeneous disk subsystems [6] but very few are concerned with heterogeneous disk subsystems [7], [8]. From a practical perspective, these techniques must be extended to support heterogeneous subsystems. This is due to the current technological trends in the area of magnetic disks, namely, the annual 40%–60% increase in performance and 50%–100% decrease in storage cost [9]. Consequently, it is very likely that failed disks will be replaced by newer models. In addition, with scalable storage subsystems, a system might evolve to consist of several disk models.

There are multiple ways of configuring the hardware and organizing data. These choices are a tradeoff in MTTSL, cost, and need for detective techniques that dissolve bottlenecks by replicating the data. In this study, we investigate three alternative organizations.

1) *Independent subservers [2], [7], [10]:* With this organization, a heterogeneous collection of disks is organized into a collection of subservers, each consisting of a homogeneous array of disk drives. A file (e.g., a movie) is assigned to one subserver—see Fig. 1(a). Hot read-only files (e.g., popular movies) might be replicated across the subservers to avoid formation of hot spots and bottlenecks. The configuration may employ a detective technique to detect hot spots and replicate the data to dissolve these bottlenecks.
2) *Dependent subservers:* Similar to the previous technique, this technique constructs a collection of subservers consisting of homogeneous disks, however, it stripes a file across the subservers in order to distribute the load of a sequential retrieval across all subservers—see Fig. 1(b). (This is an extension of Streaming RAID [11] to a heterogeneous collection of subservers.) When compared with technique 1, this strategy prevents the formation of bottlenecks and no longer replicates read-only data. However,

Manuscript received March 27, 2002; revised May 5, 2003. This work was supported in part by the National Science Foundation under Grants EEC-9529152 (IMSC ERC), IIS-0082826, IRI-9203389, IRI-9258362 (NYI award), and CDA-9216321, and by a Hewlett-Packard unrestricted cash/equipment gift. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Nelson L. S. Fonseca.

The authors are with the Department of Computer Science, University of Southern California, Los Angeles, CA 90089 USA (e-mail: rzimmerm@imsc.usc.edu; shahram@pollux.usc.edu).

Digital Object Identifier 10.1109/TMM.2004.837231

TABLE I
LIST OF TERMS USED REPEATEDLY IN THIS STUDY AND THEIR RESPECTIVE DEFINITIONS

Term	Definition
$MTTF = \frac{1}{\lambda}$	Mean time to failure; mean lifetime of an individual, physical disk with failure rate λ
$MTTSL$	Mean time to service loss
$MTTR = \frac{1}{\mu}$	Mean time to repair of a physical disk with repair rate μ
d_i^p	Physical disk drive i
d_i^l	Logical disk drive i
p_i	Number of logical disks that map to physical disk i
D	Number of physical disk drives
D^l	Number of logical disk drives
G	Parity group size
$\mathcal{G}_i$	Parity group i; a data stripe allocated across a set of G disks and protected by a parity code
$R(t)$	Reliability function
$\mathcal{N}$	Total number of streams supported
$\mathcal{B}$	File system block size
S	Storage space of a physical disk drive

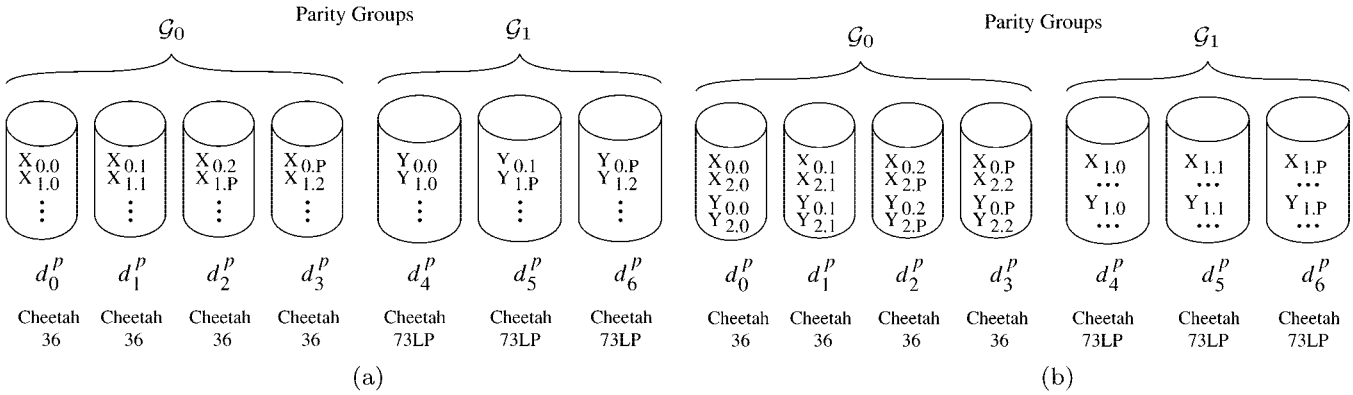


Fig. 1. Partitioning techniques 1 and 2. Each data stripe, e.g., X_0, is declustered ($X_{0.0}$, $X_{0.1}$, $X_{0.2}$, with $X_{0.P}$ indicating the parity block). (a) Technique 1: independent subservers. Objects X, Y, etc., are assigned to only one parity group. (b) Technique 2: dependent subservers. Objects X, Y, etc., are assigned to all parity groups.

its MTTSL is inferior due to data dependence among the subservers.

3) *Disk merging [12]:* This technique constructs a logical collection of disk drives from an array of heterogeneous disk drives. The logical disks appear homogeneous to the upper software layers of the system. A logical disk drive might be realized using a fraction of the bandwidth and storage space provided by several physical disk drives. For example, in Fig. 2, logical disk number 2 $\left(d_2^l\right)$ is realized using a fraction of the bandwidth provided by physical disk drives 0, 1, and 2 (d_0^p, d_1^p, and d_2^p). When compared with dependent subservers (technique 2), this technique minimizes the amount of memory required by continuous media and hence results in the lowest cost per stream. Moreover, it provides effective support for diverse configurations. For example, if a system administrator extends the storage subsystem with a single new disk drive, this paradigm can utilize both the storage space and bandwidth of this disk drive. With the other two techniques, it would be difficult to form a subserver that consists of a single disk drive (whose failure would result in reorganization of data). On the down side, the MTTSL of this technique is inferior (but usually sufficient) to the other two strategies.

With the homogeneous view created by disk merging, a parity-based redundancy scheme such as RAID can be applied, as long as it is modified to handle the following constraint: a physical disk drive should not form multiple logical disks of a single parity group. Otherwise, the failure of this physical disk would result in loss of data, i.e., an entire parity group would falter at the logical level. In Section II, we describe analytical models to compute the MTTSL of each technique. Next, Section III quantifies the performance tradeoffs associated with each strategy. These results quantify the qualitative tradeoffs discussed in this section. Conclusions are contained in Section IV.

II. PARITY-BASED REDUNDANCY FOR HETEROGENEOUS STORAGE

The MTTSL is the premier measure to assess the fault resilience of a storage system. In this section, we derive the MTTSL for heterogeneous storage systems. As we will see in Sections II-A–II-C, the MTTSL denotes the inverse of a system's failure rate. Hence, a storage system with a MTTSL of 1000 years has a chance of failure once every 1000 years, or about 0.01 failures during its normal life span of up to ten years. MTTSL values that are several orders of magnitude greater than the physical lifetime of the storage system are desirable because they indicate that serious failures will be rare. We start with an overview of our target architecture followed by a disk merging configuration planner and then proceed to

reliability modeling. We introduce a Markov model to compute the MTTSL of a single parity group in a heterogeneous storage system. Finally, we extend the model to multiple parity groups.

To support data intensive applications we assume a multi-node server architecture. Each node is attached to a set of local disks via an I/O bus, such as SCSI. The nodes are linked with each other through a high-speed interconnection network. While we recognize the importance of investigating both disk failures and node failures, we limit the focus of this paper to disk failures.

With redundant data, a single failure does not lead to data loss. Furthermore, if the failed disk is replaced with a new device, then the data on the new disk can be rebuilt. Therefore, a server can be said to operate in either of three modes: 1) *normal* mode, i.e., all nodes are fully operational; 2*degraded* mode, i.e., some disk or node has failed; and 3) *rebuild* mode, i.e., the data on a repaired disk or node is being restored. All of these three modes must be considered when designing reliability techniques to mask disk and node failures, such that the service to the applications can be continued.

A. Configuration Planner for Disk Merging

As illustrated in Fig. 2, with the disk merging technique a physical disk is divided into p_i logical disks, where $p_i > 0$ is a real value. Exactly how many logical disks are created on top of the same physical hardware can vary. In this section we describe a configuration planner which computes logical to physical mappings based on two optimization criteria: 1) high bandwidth utilization and 2) high storage space utilization.

Ideally, we would like to achieve the highest utilization of both bandwidth and storage space. However, in a heterogeneous storage system that is not always possible. To illustrate, consider the following example. We add a new disk to our system which has twice the storage capacity but the same data transfer rate as the existing disk drives. Clearly, if we use all its space and the data stored has a similar access frequency as the data stored on the existing disks, then the new disk's bandwidth will become a bottleneck. Hence, the configuration planner needs to be instructed whether to optimize for space or throughput. The configuration planner is presented in detail in [14] and we will summarize its operations here.

1) Planner Operation: Fig. 3 shows the schematic structure of the configuration planner. It works in two stages: *Stage 1* enumerates all the possible configuration tuples based on a user supplied description of the disk subsystem under consideration. It utilizes a database that contains the parameters of all the disk models that are known to the planner. Its calculations are based on the analytical models for disk merging [12], [14]. *Stage 2* filters the output of Stage 1 by removing configurations that do not meet the performance objectives of the application. The result of Stage 2 is a set of proposed configurations that provide full bandwidth utilization ordered by the lowest cost per stream or, alternatively, make the best use of the available storage space.

The input parameters to Stage 2 are: a) the minimum desired number of streams; b) the maximum desired cost per stream; and c) a hypothetical cost per megabyte (MB) of disk space that cannot be utilized by continuous media applications. An optimization component calculates the logical cost C per stream for each tuple Q' that meets the minimum requirements, according to the following cost model:

$$C = \frac{(2 \times \mathcal{B} \times M\$) + \sum_{i=0}^{D-1} D\$_i}{\mathcal{N} \times D^l} + \frac{W \times S \times W\$}{100\% - W} \quad (1)$$

where $M\$$ is the cost per MB of memory, $D\$_i$ denotes the cost of each physical disk drive i, $W\$$ is the cost per MB of unused storage space W (S being the total space), $\mathcal{B}$ is the optimal block size, and $\mathcal{N}$ represents the total number of streams that can be supported.

By including a hypothetical cost for each MB of storage that cannot be utilized for continuous media, Stage 2 can optionally optimize for: i) the minimum cost per stream at full bandwidth; ii) the maximum storage space for continuous media; or iii) a weighted combination of i) and ii).

The final, qualifying tuples Q' will be output by Stage 2 in sorted order of ascending total cost per stream C as shown in the example of Table III. For many applications the lowest cost configuration will also be the best one. In some cases, however, a slightly more expensive solution might be advantageous. For example, the total number of streams supported, $\mathcal{N}$, is different for each configuration. Hence, a designer might choose the third configuration from the top in Table III at a slightly higher cost per stream but for an increased number of displays (519 versus 425). Alas, in some cases the final decision for a certain configuration is application-specific, but the planner will always provide a comprehensive list of candidates to select from. The final step is to assign the D^l logical disks generated by the planner to the physical devices of the storage system in the following manner. The first $\lfloor p_0 \rfloor$ complete logical disks are assigned to physical disk d_0^p. If d_0^p can support any fractional disks, then the next logical disk is constructed from the fractions of d_0^p and other fractional physical disks. An algorithm for finding and combining the right fractions and a discussion of its complexity are contained in of [14, App. B]. Next, the planner assigns $\lfloor p_1 \rfloor$ logical disks to d_1^p, and so on, until all logical disks are mapped.

B. Parity Group Assignment

In parity-based systems, the disks are partitioned into *parity groups*. The parity information is computed across the blocks in a parity group, most commonly with an XOR function. The large storage space overhead of mirroring is reduced since for a parity group size of G only one $1/G$th of the space is dedicated to data redundancy. In the basic case, the disks of a storage system are partitioned into nonoverlapping parity groups. This scheme can tolerate one disk failure per parity group. However, when a parity group operates in degraded mode, each access to the failed disk triggers the retrieval of blocks from all of the disks within this parity group to reconstruct the lost data. Thus, the load on all operational disks increases by 100% under failure, making this parity group a hot spot for the entire system. To distribute the additional load more evenly, parity groups may be rotated such that they overlap [15]. Further improvements can be achieved by assigning blocks pseudo-randomly to parity groups [16]. To provide a focused presentation, we will concentrate on the simple scenario of nonoverlapping parity groups.

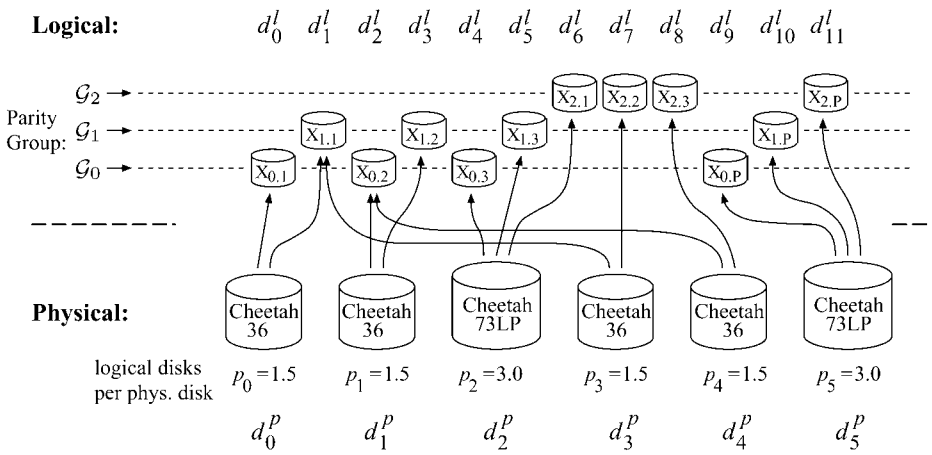


Fig. 2. Technique 3: physical (heterogeneous) and logical (homogeneous) view of a multidisk storage server employing disk merging. Six physical disks are mapped to 12 logical disks. In addition, a possible parity group assignment to create nonoverlapping parity groups is shown. All of a physical disk's logical disks must map to different parity groups ($G = 3$).

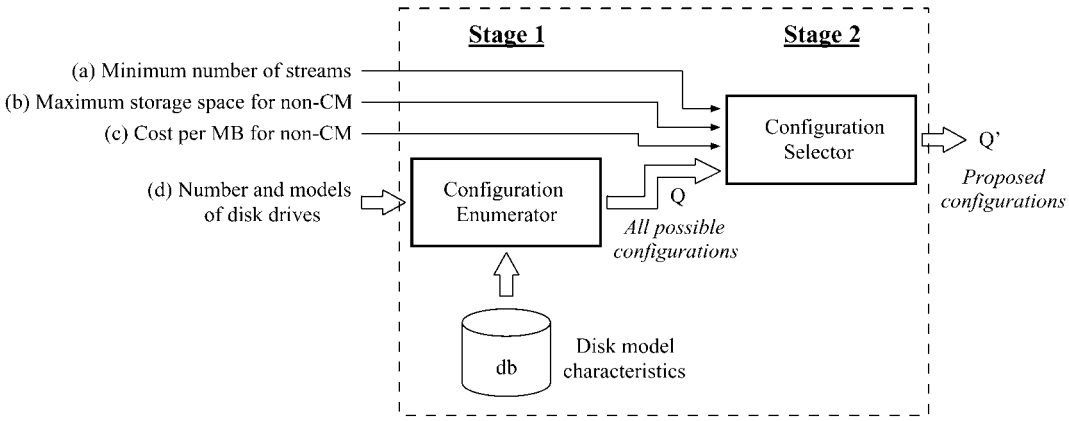


Fig. 3. Configuration planner structure.

TABLE II
SINGLE DISK RELIABILITY FOR THREE COMMERCIAL DISK DRIVES. IN THESE EXAMPLES, THE MANUFACTURER, SEAGATE TECHNOLOGY LLC, REPORTED THE RELIABILITY AS MEAN-TIME-BETWEEN-FAILURES (MTBF). A SIMPLE APPROXIMATION FOR MTBF IS $MTBF = MTTF + MTTR$ WHERE MTTR DENOTES THE MEAN-TIME-TO-REPAIR [13, PP 206]. BECAUSE FAILED DISKS ARE TYPICALLY REPLACED AND NOT REPAIRED, THE NOTION OF MTTR REFERS TO REPLACING THE DISK AND REBUILDING ITS CONTENT ONTO A NEW DISK. THIS PROCESS CAN USUALLY BE COMPLETED WITHIN SEVERAL HOURS. HENCE $MTTR \ll MTTF$ AND THUS FOR MOST PRACTICAL PURPOSES THE FOLLOWING APPROXIMATION CAN BE USED: $MTTF \approx MTBF$

Disk Series	Storage Capacity	MTBF (power-on hours)
Cheetah 18	18 GB	1,000,000
Cheetah 36	36 GB	1,000,000
Cheetah 73LP	73 GB	1,200,000

When the logical disks that were created with the disk merging technique are assigned to parity groups, the following constraint must be considered. Some logical disks may map to the same physical device and hence be dependent on each other, i.e., a failure at the physical level may cause multiple logical disks to become unavailable simultaneously. Consequently, two dependent logical disks cannot be assigned to the same parity group. The reason is that with a traditional XOR-based parity computation exactly *one* data block of a stripe can be reconstructed as long as *all the other* blocks of that particular stripe are available. Consequently, the number of independent parity groups D^l/G needs to be larger than any number of logical disks that map to a single physical disk. This can be formally expressed with the following parity group size constraint

$$G \leq \left\lfloor \frac{D^l}{\max(p_0, p_1, \ldots, p_{D-1})} \right\rfloor \tag{2}$$

where G denotes the parity group size, D^l represents the total number of logical and D the number of physical disks, and p_i denotes the number of logical disks that map to physical disk d_i^p (for example $p_0 = 1.5$ in Fig. 2). Note that with the disk merging technique, a p_i value which is not an integer indicates that some logical disk is only partially mapped to physical disk i. For example, in Fig. 2 only 50% of d_1^l is mapped to d_0^p (for further details, see [12]).

Fig. 2 shows an example storage system with six physical disk drives that map to 12 logical disks. The parity group size G is required to be either less than or equal to $\lfloor (D^l/\max(p_0, \ldots, p_5)) \rfloor = \lfloor (12/\max(1.5, 1.5, 3.0, 1.5, 1.5, 3.0)) \rfloor = \lfloor (12/3.0) \rfloor = 4$. Hence, the maximum parity group size of 4 can be accommodated by creating $12/4 = 3$ or more parity groups $\mathcal{G}_i$. For illustration purposes, we will use a simple, nonoverlapping parity group scheme. One possible assignment of the 12

TABLE III
PLANNER *STAGE 2* OUTPUT FOR 30 DISKS WITH DISK MERGING (10 × ST31200WD, 10 × ST32171WD, AND 10 × ST34501WD). THESE FINAL TUPLES Q' ARE SORTED ACCORDING TO THEIR INCREASING, ADJUSTED COST C' (FOURTH COLUMN FROM THE RIGHT) WHICH INCLUDES A HYPOTHETICAL COST FOR EACH MB THAT CANNOT BE UTILIZED FOR CONTINUOUS MEDIA ($W\$ = \$0.05$ PER MB IN THIS EXAMPLE, $M\$ = \$1$ PER MB OF MEMORY AND $D\$_i = \$500$ PER DISK DRIVE). THE MINIMUM NUMBER OF STREAMS DESIRED WAS 300

Total # of Streams $\mathcal{N}$	Total # of Log. Disks D^l	Optimal Block Size $\mathcal{B}$ [MB]	Wasted Space W [%]	Cost C [$]	Adjusted Cost C' [$]	Logical Disks per Physical Disk $p_0, ..., p_9$	$p_{10}, ..., p_{19}$	$p_{20}, ..., p_{29}$
425	425	0.5054	13.670	36.30	542.50	6.5	13.7	22.3
455	65	0.6955	13.863	34.36	547.72	1.0	2.1	3.4
519	173	2.3998	14.029	33.70	553.19	2.5	5.6	9.2
525	105	2.9167	14.058	34.40	554.98	1.5	3.4	5.6
420	105	0.5300	14.058	36.77	557.35	1.5	3.4	5.6
515	515	1.8728	14.181	32.87	557.98	7.5	16.7	27.3
360	360	0.2970	14.373	42.26	574.49	5.5	11.7	18.8
478	239	0.9421	14.730	33.26	578.71	3.5	7.8	12.6
332	166	0.2463	14.452	45.67	580.84	2.5	5.4	8.7
318	106	0.2325	15.719	47.63	629.69	1.5	3.5	5.6
366	61	0.2938	17.140	41.57	676.27	1.0	1.9	3.2
364	182	0.2938	17.593	41.80	693.26	3.0	5.8	9.4

logical disks $d_0^l, \ldots, d_{11}^l$ to three parity groups $\mathcal{G}_0, \ldots, \mathcal{G}_2$ is as follows (also illustrated in Fig. 2): $\mathcal{G}_0 = \{d_0^l, d_2^l, d_4^l, d_9^l\}$, $\mathcal{G}_1 = \{d_1^l, d_3^l, d_5^l, d_{10}^l\}$, and $\mathcal{G}_2 = \{d_6^l, d_7^l, d_8^l, d_{11}^l\}$.

From the above parity group assignment we can further determine which physical disks participate in each of the parity groups ("$\mapsto$" denotes "maps to")

$$\begin{aligned}\mathcal{G}_0 &\mapsto d_0^p, d_1^p, d_2^p, d_5^p\\ \mathcal{G}_1 &\mapsto d_0^p, d_1^p, d_2^p, d_4^p, d_5^p\\ \mathcal{G}_2 &\mapsto d_2^p, d_3^p, d_4^p, d_5^p.\end{aligned}$$

C. Reliability Modeling

With the help of data replication or parity coding, the reliability of a disk array can be greatly improved. Several studies have quantified the reliability of homogeneous disk arrays in the context of continuous media servers with mirroring [17], parity coding schemes (RAID) [1], [5], [6], [18], [19], or a combination of both [20]. The concepts of reliability or fault tolerance involve a large number of issues concerning software, hardware (mechanics and electronics), and environmental (e.g., power) failures. A large body of work already exists in the field of reliable computer design and it will provide the foundation for this section. Because it is beyond the scope of this paper to cover all the relevant issues, we will restrict our presentation to the reliability aspects of the disk hardware.

1) Analytical Model for Reliability: The *reliability function* of a system, denoted $R(t)$, is defined as the probability that the system will perform satisfactorily from time zero to time *t*, given that it is initially operational [13, pp7]. When the higher failure rates of a component at the beginning (infant mortality or burn-in period) and at the end (wear-out period) of its lifetime are excluded, then there is strong empirical evidence that the *failure rate* λ during its normal lifetime is approximately constant [13], [19]. This is equivalent to an exponential distribution of the product's lifetime and gives rise to a reliability function $R(t)$ that can be expressed as

$$R(t) = e^{-\lambda t}. \tag{3}$$

Perhaps the most commonly encountered measure of a system's reliability is its MTTF, which is defined as follows:

$$\text{MTTF} = \int_0^\infty R(t)dt. \tag{4}$$

With the exponential lifetime distribution $R(t)$ substituted from (3), the MTTF simply becomes

$$\text{MTTF} = \int_0^\infty e^{-\lambda t}dt = \frac{1}{\lambda}. \tag{5}$$

In a heterogeneous storage environment it is possible for each physical disk drive d_i^p to have its own MTTF_i and hence its own failure rate λ_i (see Table II).

The mean lifetime of a system can be greatly prolonged if it contains redundant modules that can be repaired when a component failure occurs. This is the case for parity based storage systems, where a parity group can run in degraded mode until a failed disk is replaced and its data is recovered. Note that the repair time for a nonredundant system, such as a single disk drive, is by definition unimportant for its analytically computed reliability because if any of its components fail, then the whole system has failed[1] (see the discussion of Table II). A redundant system can continue to operate while being repaired (albeit with less performance) and its mean time to repair (MTTR) is crucial because it provides a "window of opportunity" during which a second component failure may result in a system failure. Hence, a short MTTR is very desirable. The MTTR [2] is often difficult to model analytically, and it must usually be estimated or measured [13, pp205]. If the operating environment of the system is known, then an estimate can be given. For example, if spare disks are at hand and service personnel is part of the staff at the site location, a MTTR of a few hours should be realistic. If spare parts must be ordered or service personnel called in, then the repair time may be in the order of days or even weeks. For the analytical models, the repair rate is commonly denoted μ

[1]In practice, being able to salvage the valuable data on a device in a timely manner is often important.

[2]It is customary to refer to MTTR as the mean-time-to-repair, even though for most practical purposes a failed magnetic disk will be replaced and not repaired. In such a case, the MTTR should include the time to rebuild the lost data on the new disk.

and for exponential distributions MTTR $= (1/\mu)$. In a heterogeneous environment, the repair rate may not be the same for all disk types; hence in the most general case each disk has its own repair rate μ_i.

2) Markov Model for a Single Parity Group: Markov models provide a powerful tool for basic reliability modeling if the system is composed of several processes (such as a failure process and a repair process) [13], [19]. Each component of such a model is assumed to be independent. However, logical disks produced by the disk merging technique are not a priori independent, because several of them may map to the same physical disk drive. Conversely, fractions of a single logical disk may be spread across several physical disks. In Section II-C.III we will derive the mean time to failure for each individual logical disk in a storage system, such that the necessary independence for the Markov model is preserved.

Fig. 4 shows the Markov model for a single parity group comprised of G independent disk drives. The states are labeled with the number of disk failures that the array is experiencing. For example, in state 0 all disks are operational. Then with probability λ_i disk d_i becomes unavailable and a transition is made to one of the states labeled 1. With probability μ_i repairs are completed and the array is again in state 0. Or, with probability λ_j a second disk fails and the transition to state 2 indicates that an unrecoverable failure has occurred. As illustrated, the number of states of the model is $(G \times (G-1)) + 2$, i.e., it is quadratic with respect to the parity group size G. The evaluation of such a model is computationally quite complex, especially for larger values of G. Hence, we propose the following two simplifying assumptions.

1) The repair rate μ_i is the same for all the disks in the system, i.e., $\mu = \mu_0 = \mu_1 = \cdots = \mu_{G-1}$. It is likely that the time to notify the service personnel is independent of the disk type and will dominate the actual repair time. Furthermore, disks with a higher storage capacity are likely to exhibit a higher disk bandwidth, leading to an approximately constant rebuild time.
2) The probability of a transition from any of the states 1 to state 2 (a second failure) is the sum of the failure rates of all the remaining disks of the group (i.e., $\sum_{j=0}^{G-1} \lambda_j$ minus the one failure rate that led to the transition from state 0 to 1). In the most general case (because of heterogeneity), each disk can have a different failure rate. Hence, the correct solution requires $G-1$ cases in each of which we subtract the failure rate of the first disk that failed from the sum. To establish a single case and a closed form solution we propose to always subtract the smallest failure rate $\lambda_{\min}$ of any disk in the group. Hence, by ordering (without loss of generality) λ_j according to decreasing values, $\lambda_0 \geq \lambda_1 \geq \cdots \geq \lambda_{G-1}$, we can express the probability of a transition from state 1 to 2 to be $\sum_{i=0}^{G-2} \lambda_j$. This approximation will lead to the highest value for the sum $\sum_{i=0}^{G-2} \lambda_j$ and therefore the shortest parity group lifetime. In other words, our estimate for the lifetime is conservative, and in most cases the real lifetime will be slightly longer.

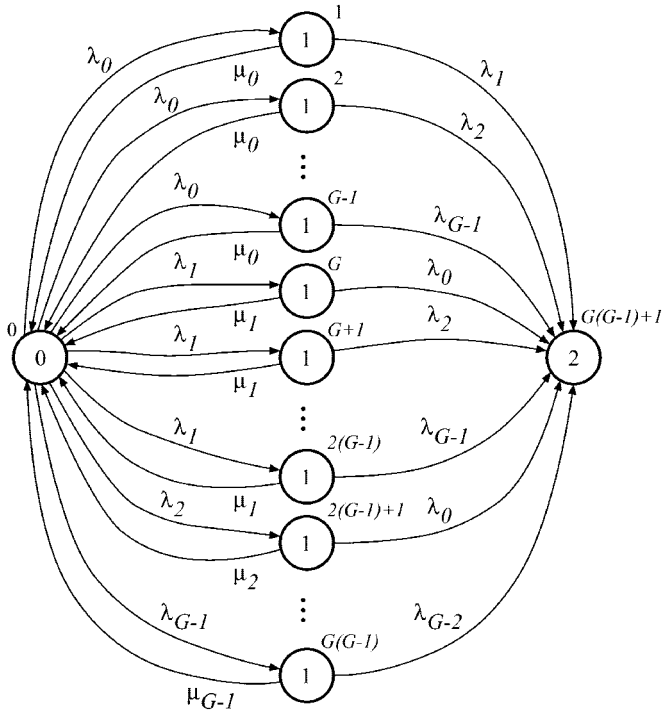


Fig. 4. Markov model for a heterogeneous disk array (one parity group). The labels in each state denote the number of disk failures encountered by the array. State 2 is a trapping state, i.e., the probability of exiting is zero, meaning the array has failed.

With these assumptions, the Markov model is simplified to three states as shown in Fig. 5 and can be solved using a set of linear equations and Laplace transforms [13], [19]. In Section III, we will illustrate that the simplifications result only in minor differences between the simulation experiments and the analytical results.

If the expression of $R(t)$ does not need to be obtained, MTTSL can be found more easily [19]. Beginning in a given state S_i, the expected time until the first transition into a different state S_j can be expressed as

$$E[S_i \text{ to } S_j] = E[\text{time in } S_i \text{ per visit}] + \sum_{k \neq i} P(\text{transition } S_i \text{ to } S_k) \times E[S_k \text{ to } S_j] \quad (6)$$

where

$$E[\text{time in } S_i \text{ per visit}] = \frac{1}{\sum \text{rates out of } S_i} \quad (7)$$

and

$$P(\text{transition } S_i \text{ to } S_k) = \frac{\text{rate of transition to } S_k}{\sum \text{rates out of } S_i}. \quad (8)$$

The solution to this system of linear equations includes an expression for the expected time beginning in state S_0 and

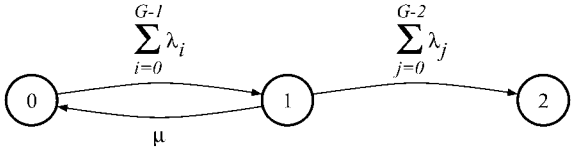


Fig. 5. Simplified Markov model for a heterogeneous disk array.

ending on the transition into state S_2, that is, for MTTSL. For the Markov model in Fig. 5, this system of equations is

$$E[S_0 \text{ to } S_2] = \frac{1}{\sum_{i=0}^{G-1}\lambda_i} + \frac{\sum_{i=0}^{G-1}\lambda_i}{\sum_{i=0}^{G-1}\lambda_i} E[S_1 \text{ to } S_2] \tag{9}$$

$$E[S_1 \text{ to } S_2] = \frac{1}{\mu + \sum_{j=0}^{G-2}\lambda_j} + \frac{\mu}{\mu + \sum_{j=0}^{G-2}\lambda_j} E[S_0 \text{ to } S_2] + \frac{\sum_{j=0}^{G-2}\lambda_j}{\mu + \sum_{j=0}^{G-2}\lambda_j} E[S_2 \text{ to } S_2] \tag{10}$$

$$E[S_2 \text{ to } S_2] = 0. \tag{11}$$

The resulting mean lifetime of a single parity group of independent heterogeneous disks is shown in (12)

$$\text{MTTSL} = \frac{\mu + \sum_{i=0}^{G-1}\lambda_i + \sum_{j=0}^{G-2}\lambda_j}{\sum_{i=0}^{G-1}\lambda_i \times \sum_{j=0}^{G-2}\lambda_j}. \tag{12}$$

Under most circumstances, the repair rate $\mu = 1/\text{MTTR}_{\text{disk}}$ will be much larger than than any of the failure rates $\lambda_i = 1/\text{MTTR}_{\text{disk}_i}$. Thus, the numerator of (12) can be simplified as presented in (14) (see [19, p. 141])

$$\text{MTTSL} = \frac{\mu}{\sum_{i=0}^{G-1}\lambda_i \times \sum_{j=0}^{G-2}\lambda_j} \tag{13}$$

$$= \frac{1}{\sum_{i=0}^{G-1}\frac{1}{\text{MTTF}_{\text{disk}_i}} \times \sum_{j=0}^{G-2}\frac{1}{\text{MTTF}_{\text{disk}_j}} \times \text{MTTR}_{\text{disk}}}. \tag{14}$$

If all the failure rates λ_i are equal, i.e., $\lambda = \lambda_0 = \lambda_1 = \cdots = \lambda_{G-1}$, then (14) will, fortunately, correspond to the derivation for a homogeneous disk RAID level-5 array with group size G (see [5])

$$\text{MTTSL}_{homogeneous} = \frac{\text{MTTF}_{\text{disk}}^2}{G \times (G-1) \times \text{MTTR}_{\text{disk}}} \tag{15}$$

where $\text{MTTF}_{\text{disk}}$ is the mean-time-to-failure of an individual disk, $\text{MTTR}_{\text{disk}}$ is the mean-time-to-repair of a single disk, and G is the parity group size. Both (14) and (15) capture the probability of an independent double disk failure within a single parity group.

Example 1: Consider a parity coded disk array consisting of three Cheetah 36 and two Cheetah 73LP disk drives $(G = 5)$. The mean time to failure of the Cheetah 36 series is $\text{MTTF}_{\text{disk}} = 1\,000\,000$ h and for the Cheetah 73LP series it is $\text{MTTF}_{\text{disk}} = 1\,200\,000$ h (see Table II). Assuming an average repair time of $\text{MTTR}_{\text{disk}} = 6$ h, then the MTTSL approaches 1 064 000 years. □

By comparison, a homogeneous array consisting of five Cheetah 36 disks would have a similar mean lifetime of MTTSL = 951 000 years. The numerical results confirm our initial goal of analytically computing the MTTSL for heterogeneous disk arrays.

3) Logical Disk Independence: The Markov model of the previous section assumes that all the disks are independent. This assumption is not guaranteed for disk merging, because one logical disk may map to several physical disks and vice versa. Hence, to be able to apply (14) at the logical level of a disk merging storage system, we will need to derive the mean time to failure of each individual, logical disk. Two cases are possible: 1) a logical disk maps to exactly one physical disk and 2) a logical disk maps to multiple physical disks. Consider each case in turn.

If a logical disk maps completely to one physical disk then its life expectancy is equivalent to the lifetime of that physical disk. Consequently, it inherits the mean time to failure of that disk. For example, the mean lifetimes of the logical disks d_0^l and d_3^l of Fig. 2 are 1 000 000 h each.

If, on the other hand, a logical disk depends on multiple physical disks, then it will fail whenever any one of the physical disks fails. Hence, we can apply the harmonic sum for failure rates of independent components [6]

$$\text{MTTF}_{d^l} = \frac{1}{\frac{1}{\text{MTTF}_{d_{0'}^p}} + \frac{1}{\text{MTTF}_{d_{1'}^p}} + \cdots + \frac{1}{\text{MTTF}_{d_k^p}}}. \tag{16}$$

As an example, consider applying the above two observations to the the four logical disks in parity group $\mathcal{G}_0$ of Fig. 2. This will result in the following mean lifetime for each logical disk:

$$\begin{aligned}
\text{MTTF}_{d_0^l} &= 1\,000\,000 \text{ h} \\
\text{MTTF}_{d_2^l} &= 500\,000 \text{ h} \\
&= \frac{1}{\frac{1}{1\,000\,000} + \frac{1}{1\,000\,000}} \text{ h} \\
\text{MTTF}_{d_4^l} &= 1\,200\,000 \text{ h} \\
\text{MTTF}_{d_9^l} &= 1\,200\,000 \text{ h}.
\end{aligned} \tag{17}$$

Recall that, if multiple logical disks map to the same physical disk, then a failure of that physical drive will concurrently render all its logical drives unavailable. For the aforementioned reason, all logical disks that map to the same physical disk must be assigned to different parity groups (see Fig. 2).

4) Multiple Parity Groups: Large storage systems are typically composed of several parity groups. All of the groups need to be operational for the system to function properly. If the

TABLE IV
ANALYTICAL RESULTS FOR THREE TECHNIQUES WITH THE SAME DISK STORAGE SYSTEM CONSISTING OF 15 DISK DRIVES (FIVE CHEETAH 18, 5 CHEETAH 36, AND FIVE CHEETAH 73LP). THE PARITY GROUP SIZE IS $G = 5$ AND $MTTR = 6$ HOURS FOR ALL THREE CONFIGURATIONS

Technique		1. Independent Subservers	2. Dependent Subservers	3. Disk Merging
Number of subservers		3	3	1
Number of simultaneous streams @ 3.5 Mb/s		192	395	395
Memory[a]	[MB]	1,600	2,395.5	2,293.4
Storage capacity[b]	[GB]	216	457	429
$MTTSL$	[years]	$> 6.6 \times 10^{14}$ [c]	200,000	127,600
Cost per stream[d]	[$]	40.73	20.20	20.15

[a]Memory size based on double-buffering.
[b]The total capacity of all the disks is 635 GB. Approximately 20% is used for parity information.
[c]Reliability based on triple modular redundancy [13, pp 215], i.e., two of three subservers must function. In effect the system is doubly protected with parity groups and TMR.
[d]Cost based on a price of $500 per disk drive and $0.20 per MB of memory.

groups are assumed to be independent, then the system can be thought of as a series of components. The overall reliability of such a configuration is

$$R_{\text{Series}}(t) = \prod_{i=1}^{D/G} R_i(t) \tag{18}$$

where D/G is the number of parity groups and $R_i(t)$ is the individual reliability function of each group.

The overall mean lifetime MTTSL of a series of groups can then be derived from the harmonic sum of the individual, independent failure rates of all the components, as shown in (19) [6]

$$\text{MTTSL}_{\text{System}} = \frac{1}{\frac{1}{\text{MTTSL}_1} + \frac{1}{\text{MTTSL}_2} + \cdots + \frac{1}{\text{MTTSL}_{D/G}}}. \tag{19}$$

Consider the simple example shown in Fig. 2. Three parity groups are formed from 12 logical disks, which in turn have been constructed from six physical devices. The physical $\text{MTTF}_{\text{disk}}$ are assumed to be 1 000 000 h (d_0^p, d_1^p, d_3^p, and d_4^p), respectively, 1 200 000 h (d_2^p and d_5^p), corresponding to Cheetah 36 and Cheetah 73LP devices (see Table II). Each logical disk inherits its $\text{MTTF}_{\text{disk}}$ from the physical drive it is mapped to. For example, $\text{MTTF}_{d_0^l} = 1\,000\,000$ h.

The resulting mean lifetimes for both group $\mathcal{G}_0$ and $\mathcal{G}_1$, are $\text{MTTSL}_{\mathcal{G}_0} = \text{MTTSL}_{\mathcal{G}_1} = 1{,}063{,}558$ years, while the $\text{MTTSL}_{\mathcal{G}_2}$ is 1 831 368 years [(12)] . The mean time to failure for the whole storage system is therefore

$$\text{MTTSL}_{\text{System}} = \frac{1}{\frac{2}{1{,}063{,}558} + \frac{1}{1{,}831{,}368}} = 412{,}113 \text{ years}. \tag{20}$$

Such an extraordinarily high reliability will minimize the chance of failure during the physical lifetime, say ten years, of the storage system.

III. EXPERIMENTAL RESULTS

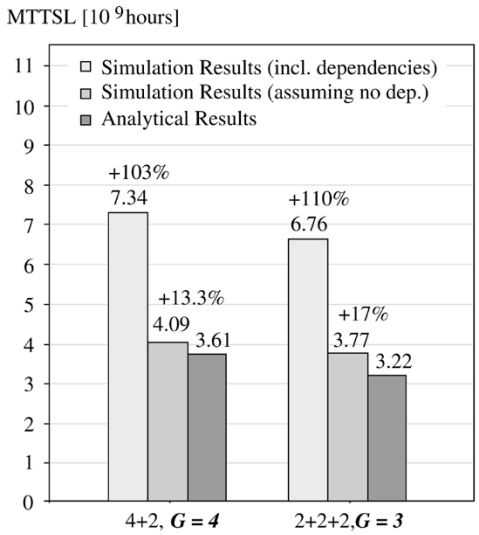
A. Analytical Analysis

We compared the three HERA techniques, independent subservers, dependent subservers, and disk merging with each other based on a disk subsystem configuration consisting of five Seagate Cheetah 18 (model ST118 202LC), five Cheetah 36 (ST136 403LC), and five Cheetah 73LP (ST373 405LC) disk drives. Our main goal was to observe the resource utilization and the MTTSL for all three techniques.

Table IV summarizes the results obtained from the analytical models of Section II. The five Cheetah 18 disks provided a total of 90 GB of storage, the Cheetah 36 180 GB, and the Cheetah 73LPs 365 GB, for a total of 635 GB. For all techniques, a parity group size of $G = 5$ was chosen, resulting in a maximum usable aggregate capacity of 80% or 508 GB. The total bandwidth for all the 15 disk drives is approximately 400 MB/s. Once seek operations, rotational latencies, etc., are factored in and the most cost-effective operating point is selected a maximum of 790 simultaneous streams can be supported[3]. Because the load doubles within a parity group that experiences a disk failure, we limit the system utilization to 50% or 395 streams. Hence only a double-disk failure in a parity group will cause the termination of some streams and therefore a loss of service.

For technique 1, the independent subserver paradigm, we based our observations on a triple modular redundancy (TMR) scheme, where two out of three units need to function for continued operation (we replicated movies accordingly) [13, pp 215]. Because the Cheetah 18 and Cheetah 36 subservers provide less than half of the total bandwidth and storage capacity, only 96 streams can be supported by this configuration. However, this decision results in the highest MTTSL and cost per stream among the three techniques. With technique 2, dependent subservers, a much higher utilization of bandwidth and storage space can be achieved. However, the mean lifetime is also considerably lower. The disk merging system provides a high storage capacity and achieves the same throughput as technique 2, but it reduces the amount of memory required slightly for continuous media which, in turn, reduces the cost per stream. The MTTSL of this technique is the lowest among the three paradigms, but it is still sufficiently high for most practical applications. The main advantages of disk merging over technique 2 are as follows. First, a single block size is used across the complete storage system. This greatly simplifies caching and other server component implementations (i.e., the block to packet conversion and the scheduling). Second, while

[3]The bandwidth of a stream is assumed to be 3.5 Mb/s, e.g., MPEG-2.



(a)

Configuration	4+2, $G = 4$ (see Figure 2)	2+2+2, $G = 3$
Physical disks	6 (4+2)	6 (2+2+2)
Disk types	2 Cheetah 36, Cheetah 73LP	3 Cheetah 18, Cheetah 36, Cheetah 73LP
Logical disks	12	18
Parity groups	3 ($G = 4$)	6 ($G = 3$)

(b)

Fig. 6. (a) Two heterogeneous test storage systems. (b) Configuration parameters.

with technique 2 each subserver needs to consist of enough disk drives to form at least one complete parity group, there is no such requirement for disk merging. Even a single disk of a different model can be utilized. Overall, disk merging provides similar performance to dependent subservers while supporting more flexible storage configurations.

B. Simulation Experiments

In a second step—to verify our analytical models—we compared their output with the results obtained from a reliability simulator that was specifically designed for this purpose. Initially, the simulator was provided with a physical and logical storage configuration as well as a mapping between the two. The simulator then generated random, exponentially distributed failures based on the mean time to failure of each of the physical disks. A failed disk was assumed to be repaired within a fixed time of $\text{MTTR}_{\text{disk}} = 6$ h. The simulator monitored the parity groups consisting of logical disks for two or more concurrent failures. The time between two double-failures (termed *events*) was recorded as the lifetime of the storage system. To obtain an average value a total of 10 000 system failure and recovery cycles were simulated to obtain the mean time to service loss, MTTSL. Note that the simulator used the real failure rate of the disks involved at all times and hence did not make the simplifying assumption 2. of the analytical Markov model (see Section II-C-II).

1) Comparison of Analytical Versus Simulation Results: For our first experiment we selected two heterogeneous storage systems that were configured with logical disks by the configuration planner of Section II-A. Fig. 6(b) lists the details of the two setups. Both configurations consisted of six physical disk drives, however, one was constructed with two types (4 × Cheetah 36 and 2 × Cheetah 73LP), while the other consisted of three types (2 × Cheetah 18, 2 × Cheetah 36 and 2 × Cheetah 73LP). Fig. 6 illustrates the results. The experiments were performed with two different simulation modes resulting in two values per configuration. The lower value assumes that all logical disks are truly independent in their failures. However, this is not the case when parts of one physical disk map to several logical disks. In this mode, a physical disk failure that causes two (or three, etc.) parity groups to fail simultaneously is recorded as two (respectively three, etc.) separate events. This value underestimates the system reliability achieved in practice, however it is shown here because it reflects what the analytical model computes. As illustrated, these simulation results are very close to the analytical prediction (within 17%). In simulation mode two (the higher value), multiple related failure occurrences are counted as just one event and hence the result is more indicative of the real expected lifetime of a system. In all cases, the the analytical model never overestimates the reliability (i.e., it always produces a conservative estimate).

2) Impact of Logical-to-Physical Disk Mapping and Comparison With RAID5: The reliability of storage systems that consist of different physical devices cannot be compared directly with each other. In this section, we chose one physical storage system with eight identical disks to perform various comparisons. Table V lists the six different setups for the reliability tests. The physical storage system consisted of a set of eight disk drives. Six different configurations (a)–(f) were realized and tested. All physical disks could accommodate one, two or three logical disk drives. These configurations were chosen to allow direct comparisons between four different disk merging and two homogeneous RAID 5 setups. The six storage configurations are listed in the order of increasing complexity in Table V.

Configurations (a) and (b) represented two standard, homogeneous RAID level–5 arrays with a parity group size of 4. They were introduced as a reference and baseline for comparisons with the disk merging variants. The difference between (a) and (b) was the assumed reliability of the disks, with $\text{MTTF}_{\text{disk}} = 1\,200\,000$ h and $\text{MTTF}_{\text{disk}} = 1\,200\,000$ h, respectively. For configuration (c) each physical disk with a $\text{MTTF}_{\text{disk}} = 1\,200\,000$ h was split into two independent logical drives for a total of 16 logical disks. Four parity groups of

TABLE V
SIX STORAGE CONFIGURATIONS BASED ON EIGHT PHYSICAL AND 16 LOGICAL DISKS EACH [NO LOGICAL DISKS IN CASE OF (a) AND (b)] AND A PARITY GROUP SIZE OF $G = 4$

	Logical Disk Fractions	$MTTF$	Description
(a)	N/A	All 1,200,000 h	RAID level 5 array for comparison.
(b)	N/A	All 1,000,000 h	RAID level 5 array for comparison.
(c)	No	All 1,200,000 h	Each physical disk maps to two complete logical disks (1.0+1.0).
(d)	No	1,000,000 h and 1,200,000 h	Same as (c) but half of the physical disks have a $MTTF$ of only 1,000,000 hours.
(e)	Yes	All 1,200,000 h	Four of the physical disks map to three logical disks each (1.0+0.5+0.5).
(f)	Yes	1,000,000 h and 1,200,000 h	Same as (e) but half of the physical disks have a $MTTF$ of only 1,000,000 hours.

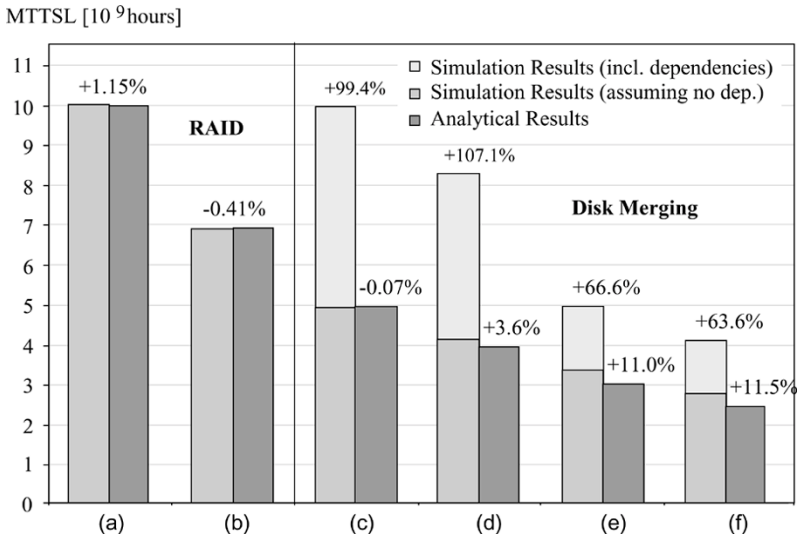


Fig. 7. MTTSL for six storage configurations based on eight physical and 16 logical disks [(c) through (f)] and a parity group size of $G = 4$. See Table V for detailed configuration parameters.

size 4 were formed from these 16 disks. To maximize independence, no two disks of a single parity group mapped to the same physical disk. Configuration (d) was an extension of (c) in that it assumed a reduced mean lifetime of $\text{MTTF}_{\text{disk}} = 1\,200\,000$ h for half of the physical disks. Configurations (e) and (f) were similar to (c) and (d) but with the additional complexity of mapping four of the logical disks to two physical disks each, i.e., four physical disks were divided into $1.0+0.5+0.5$ logical disks. Again, care was taken to maximize independence among the parity groups.

Fig. 7 shows the resulting mean lifetimes for each of the configurations (a) through (f). The RAID 5 configuration of Fig. 7(a) provides a MTTSL of 10.1×10^9 h. Because of the quadratic nature of (15), the same RAID 5 system will have a reduced mean lifetime of approximately $69\% \equiv (1/1.2^2)$ if the reliability of the disks is divided by a factor of 1.2—see Fig. 7(b). The simulation results for Fig. 7(c)–(f) shows two values each. As indicated in the previous section, the lower value assumes that all logical disks are independent and hence this reflects the assumptions of the analytical model. As illustrated, these simulation results are very close to the analytical prediction.

The simplest and most reliable disk merging configuration, shown in Fig. 7(c), achieves approximately the same actual reliability as compared with the RAID 5 setup of (a). The reason for the analytical reliability of 50% is the increased number of parity groups (4) as compared with the RAID setup (2). Configuration (d) provides a lower reliability because half of the physical disks have a reduced $\text{MTTF}_{\text{disk}}$ of 1 000 000 h. The analytical models for mixed mean time to failures of Section II-C-II introduced some simplifying assumptions which lead to a conservative estimate of the mean lifetime. Consequently, in Fig. 7(d) both simulation results are higher than the analytical predictions. In configurations (e) and (f), a physical disk may be part of up to three different parity groups, which further reduces the overall reliability of the system. However, all the disk merging configurations achieve a level of reliability at the same order of magnitude as their homogeneous counterparts, while providing increased configuration flexibility.

IV. CONCLUSION AND FUTURE DIRECTIONS

In this study we investigated parity-based fault tolerance techniques for heterogeneous storage systems. We introduced a framework that extends RAID to allow a mix of physical disk drives to be protected from data and service loss. The result was an analytical model to compute the MTTSL, which we verified through simulation experiments. The examples

provided were specific to nonoverlapping parity groups. For the disk merging technique, we have specified both design rules and algorithms that provide the necessary independence of logical disks among each other and across parity groups for the successful application of parity-based data redundancy.

The presented planner produced multiple configurations to supply optimized solutions for different applications. In this paper, we described a lower bound on the number of parity groups required for successful implementation of our disk merging technique during normal mode of operation. Note that this planner must be extended if the application desires to specify requirements in the presence of one or more failures. Design of such a planner must consider both the placement of data and scheduling techniques in support of continuous display in the presence of failures (see [1] and [16]). The specified requirement may results in a different number of parity groups. Such a planner requires further investigation and will shape our future research activities.

REFERENCES

[1] S. Berson, L. Golubchik, and R. R. Muntz, "Fault tolerant design of multimedia servers," in *Proc. ACM SIGMOD Int. Conf. Management of Data*, 1995, pp. 364–375.

[2] S. Ghandeharizadeh and C. Shahabi, "Management of physical replicas in parallel multimedia information systems," in *Proc. Foundations of Data Organization and Algorithms (FODO) Conf.*, Oct. 1993.

[3] D. J. Gemmell, H. M. Vin, D. D. Kandlur, P. V. Rangan, and L. A. Rowe, "Multimedia storage servers: A tutorial," *IEEE Computer*, May 1995.

[4] D. Bitton and J. Gray, "Disk shadowing," in *Proc. Int. Conf. Very Large Databases*, Sept. 1988, pp. 331–338.

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

[6] P. M. Chen, E. K. Lee, G. A. Gibson, R. H. Katz, and D. A. Patterson, "RAID: High-performance, reliable secondary storage," *ACM Comput. Surveys*, vol. 26, no. 2, pp. 145–185, June 1994.

[7] A. Dan and D. Sitaram, "An online video placement policy based on bandwidth to space ratio (BSR)," in *Proc. ACM SIGMOD Int. Conf. Management of Data*, San Jose, CA, May 1995, pp. 376–385.

[8] L. T. Chen, D. Rotem, and S. Seshadri, "Declustering databases on heterogeneous disk systems," in *Proc. 21st Int. Conf. Very Large Data Bases*, Zürich, Switzerland, Sept. 1995, pp. 110–121.

[9] E. Grochowski. (2001) Internal Data Rate Trend & Storage Price Projections. IBM Almaden Research Center, San Jose, CA. [Online]. Available: http://www.almaden.ibm.com/sst/

[10] J. L. Wolf, P. S. Yu, and H. Shachnai, "DASD dancing: A disk load balancing optimization scheme for video-on-demand computer systems," in *Proc. ACM SIGMETRICS*, Ottawa, ON, Canada, May 1995.

[11] F. A. Tobagi, J. Pang, R. Baird, and M. Gang, "Streaming RAID-a disk array management system for video files," in *Proc. 1st ACM Conf. Multimedia*, Anaheim, CA, Aug. 1993, pp. 393–400.

[12] R. Zimmermann and S. Ghandeharizadeh, "Continuous display using heterogeneous disk-subsystems," in *Proc. 5th ACM Multimedia Conf.*, Seattle, WA, Nov. 1997, pp. 227–236.

[13] D. P. Siewiorek and R. S. Swarz, *The Theory and Practice of Reliable Systems Design*. Bedford, MA: Digital Press, 1982.

[14] R. Zimmermann. (1999) Continuous Media Placement and Scheduling in Heterogeneous Disk Storage Systems. Univ. of Southern California, Los Angeles. [Online]. Available: ftp://ftp.usc.edu/pub/csinfo/tech-reports/papers/99-699.pdf

[15] R. Tewari, R. P. King, D. Kandlur, and D. M. Dias, "Placement of multimedia blocks on zoned disks," in *Proc. IS&T/SPIE Multimedia Computing and Networking*, San Jose, CA, Jan. 1996.

[16] B.Özden, R. Rastogi, P. Shenoy, and A. Silberschatz, "Fault-tolerant architectures for continuous media servers," in *Proc. ACM SIGMOD Int. Conf. Management of Data*, June 1996, pp. 79–90.

[17] A. Mourad, "Reliable disk striping in video-on-demand servers," in *Proc. 2nd IASTED/ISMM Int. Conf. Distributed Multimedia Systems and Applications*, Stanford, CA, Aug. 1995, pp. 113–118.

[18] R. R. Muntz and J. C. S. Lui, "Performance analysis of disk arrays under failure," in *Proc. 16th Very Large Databases Conf.*, Brisbane, Australia, 1990, pp. 162–173.

[19] G. A. Gibson, "Redundant disk arrays: Reliable, parallel secondary storage," Ph.D. dissertation, Univ. of California at Berkeley, 1991.

[20] E. W. Biersack and C. Bernhardt, "A fault tolerant video server using combined RAID 5 and mirroring," in *Proc. Multimedia Computing and Networking 1997 Conf. (MMCN'97)*, San Jose, CA, Feb. 1997, pp. 106–117.

Roger Zimmermann received the Ph.D. degree in computer science from the University of Southern California (USC), Los Angeles, in 1998.

He is currently a Research Assistant Professor with the Computer Science Department and an Investigator with the Integrated Media Systems Center, both at USC. His research activities focus on streaming media architectures, Web services, and database integration. His work on streaming media during the past several years has concentrated on core algorithms including statistical admission control that models the variability in numerous system parameters, data transmission flow control, and online storage scalability. The results have been implemented and verified in a second-generation streaming system called Yima. Yima is the basis of the Remote Media Immersion (RMI) project which has been publicly demonstrated on several occasions, for example at the Internet2 Fall 2002 Member meeting and the Internet2 Music Education Symposium at the New World Symphony in Miami (2003). Recently, he has been investigating high-speed data recording architectures (project HYDRA) with the goal of providing an effective, integrated large-scale streaming and recording platform for various time-sensitive digital media.

Dr. Zimmermann has served as Program Committee Member and Session Chair for several international conferences. Additionally, he is a member of the Editorial Board of the *International Journal of Multimedia Tools and Applications* and ACM DiSC.

Shahram Ghandeharizadeh (M'90) received the Ph.D. degree in computer science from the University of Wisconsin, Madison, in 1990.

Since 1990, he has been on the faculty at the University of Southern California (USC), Los Angeles. During the 1990s, he lead a team of graduate students to design and implement Mitra, a scalable continuous media server. Mitra was a pioneering system that supported the display of clips with both a low and a high bandwidth requirement. In 1997, Panasonic licensed Mitra for research and development purposes. In addition, he collaborated with Profs. R. Hall and Prof. D. Jacob on the design and implementation of Heraclitus[Alg,C], a system in support of hypothetical (what-if) query processing. His current research interests include execution of plans that utilize Web Services, peer-to-peer systems, streaming architectures, and management and processing of continuous streams from moving sensors.

Dr. Ghandeharizadeh is the recipient of the 1992 National Science Foundation Young Investigator's Award for his research on the physical design of parallel database systems. In 1995, he received an award from the School of Engineering at USC in recognition of his research activities.