Reliability and Performance of RAIDs

Thomas J.E. Schwarz, S.J. and Walter A. Burkhard

Gemini Storage Systems Laboratory, University of California at San Diego

Abstract—We define and analyze the ACATS declustering scheme for use with the classic Level 5 as well as MDS RAIDs. ACATS provides a paradigm for distributed sparing and checking as well. Using Markov models we calculate the reliability of classic Level 5 as well as MDS RAIDs both with and without ACATS. Our results demonstrate, for fixed hardware configurations, tradeoffs between performance and reliability within these RAID organizations. Our analysis assumes a transaction oriented environment.

I. INTRODUCTION

Magnetic disk storage has experienced modest performance improvements while processor performance has dramatically improved. Disk arrays offer increased bandwidth, narrow the performance gap, and offer large storage capacity for developing applications such as multimedia. Mirrored disks and less hardware intensive Level 5 Redundant Arrays of Independent Disks (RAIDs) have emerged as organizations that provide high reliability and improved performance. For very large disk farms, the reliability of Level 5 RAIDs is not sufficient and additional hardware, usually spare strings, is introduced. However this technique does not prevent data loss from simultaneous, related component or so-called catastrophic failure. A generalization of the Level 5 organization, the MDS RAID, provides some protection in these cases. This paper presents results from T.J.E. Schwarz's Ph.D. thesis [1].

In the remainder of this paper, we first review the Level 5 and the MDS RAID organizations. We introduce our ACATS declustering scheme. We model reliability and performance of the Level 5 RAID organization with distributed sparing and with or without ACATS as well as the MDS RAID organization with or without ACATS. We compare RAIDs with the same hardware configuration.

II. RAID ORGANIZATION

The classic Level 5 RAID organization stores data in reliability groups of disks. The parity of the *message disks* that contained user data is stored in a *check disk.* Data addressing within this organization is typically in terms segments or tracks; we refer to these as *data objects* or simply *objects.* The addresses of data objects on each disk consist of a sequence of values, typically $0, 1, \ldots, t-1$. If we write to a single message disk in the reliability group, we update the check disk using Δ the difference between the old and new message data; the new check disk data is the difference between the old check disk data and Δ. An efficient implementation of the update process reads the old data on the message disk and overwrites them with the new data after one full rotation. In the meantime, the update reads the old check data and overwrites them with the new check data, which is calculated after the old message data are read. Distributing the check data evenly over all disks in a reliability group avoids the bottleneck caused by dedicated check disks that have to be accessed during every update.

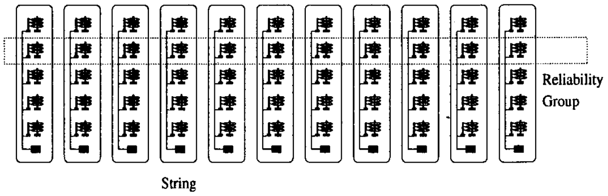

Fig. 1. Disk Array Ensemble with Eleven Strings and Five Reliability Groups

The RAID organization, depicted in Figure 1, is applicable to all our discussions. It consists of the 55 disks, eleven strings, and the central components. The disks are organized into strings that share essential hardware components such as power supply, cabling, cooling, Small Computer System Interface (SCSI) controllers and Host Bus Adapters (HBAs). The central components (not shown in Figure 1) consist in our scheme of a non-volatile cache and the RAID controller. The five reliability groups each consist of eleven disks. The use of non-volatile cache is central in our modeling; the classic Level 5 RAID had no such cache but obtains much poorer reliability and

---

T.J.E. Schwarz, S.J., is now affiliated with Saint Louis University, St. Louis, Missouri, e-mail schwarz@sluaxa.slu.edu; telephone (314) 658-8174, fax (314) 658-3874; W.A. Burkhard, e-mail burkhard@cs.ucsd.edu, telephone (619) 534-2722, fax (619) 534-7029. This work was supported in part by grants from NCR, Dayton, Ohio; NCR, Rancho Bernardo, California; and the University of California MICRO program.

0018-9464/95$4.00 © 1995 IEEE

performance.

Failure can strike a RAID at any level and the data stored in the afflicted physical unit(s) become inaccessible or are destroyed. If a single disk in a reliability group has failed, we can restore the data from the check data in conjunction with all the other data stored in the reliability group. We refer to this process as *reconstruction* and designate its duration by *reconstruction time*. The orthogonal arrangement of reliability groups and strings, as depicted in Figure 1, minimizes the impact of string failure by assigning only one member of a reliability group to a string. If the RAID processes only read operations, a disk or string failure doubles the utilization of each remaining disk in an affected reliability group. Even if the write portion of the RAID load is substantial, reconstruction of data objects lost due to the failure increases the load at disks in the same reliability group considerably. Declustering [2] (also called clustering [3]) limits the utilization increase due to disk failure by organizing objects into smaller reliability groups, that are distributed over the entire RAID. Holland [2] achieves a distribution through mathematical block designs. Below we introduce the ACATS approach to distribution that utilizes "random permutations."

III. MDS RAID Organization

Maximum Distance Separable (MDS) RAIDs, derived from well known linear codes, provide additional redundancy to improve reliability. MDS RAIDs have the same basic organization as Level 5 RAIDs with the defining difference that each reliability group contains n check disks instead of one together with the m message disks. We can always access/reconstruct the data provided we can access m disks within the reliability group; we can tolerate up to n non-essential component failures. Our data organizations are systematic and in failure-free operation accessing a data object requires single data disk access. The details of the MDS RAID organization are given in [1], [4], [5]; we observe that Level 5 RAIDs are a special case of MDS RAIDs. The reliability modeling in these earlier papers does not involve non-volatile cache or ACATS declustering.

In this work, we consider MDS RAIDs having exactly two check disks per reliability group. The update and the data reconstruction operations are similar to those for Level 5, but an update needs three disk update operations and a reconstruction accesses all but one surviving disk in a reliability group.

IV. ACATS Declustering

The *Almost Complete Address Translation Scheme* (ACATS) introduces a layer of virtual disks identical in number to the physical disks. ACATS places a data object, either a disk track or segment of a virtual disk, on a physical disk. A disk address consists of a string address and a disk-in-string address. ACATS provides a mapping between virtual and physical disk addresses in a two stage process, where each stage involves a permutation.

$$\begin{array}{ccccc} \mathtt{a} & \mathtt{b} & \mathtt{c} & \mathtt{d} & \mathtt{e} \\ \mathtt{f} & \mathtt{g} & \mathtt{h} & \mathtt{i} & \mathtt{j} \\ \mathtt{k} & \mathtt{l} & \mathtt{m} & \mathtt{n} & \mathtt{o} \end{array} \xrightarrow{1} \begin{array}{ccccc} \mathtt{d} & \mathtt{e} & \mathtt{a} & \mathtt{b} & \mathtt{c} \\ \mathtt{i} & \mathtt{j} & \mathtt{f} & \mathtt{g} & \mathtt{h} \\ \mathtt{n} & \mathtt{o} & \mathtt{k} & \mathtt{l} & \mathtt{m} \end{array} \xrightarrow{2} \begin{array}{ccccc} \mathtt{i} & \mathtt{o} & \mathtt{a} & \mathtt{l} & \mathtt{m} \\ \mathtt{d} & \mathtt{e} & \mathtt{k} & \mathtt{b} & \mathtt{h} \\ \mathtt{n} & \mathtt{j} & \mathtt{f} & \mathtt{g} & \mathtt{c} \end{array}$$

Fig. 2. ACATS Example with Five Strings and Three Reliability Groups

The first stage determines the physical string address of a virtual data object. The second stage picks the disk in the string. We implement the first stage by a permutation based on data object number. We realize the second stage as a family of permutations depending on the object number and on the string, via a quasi-random selection. We can obtain these permutations by first using a random number generator with the data object number and the string number as seeds and then using the inverse of Knuth's algorithm P [6] to obtain the permutation. In contrast, the first stage permutation can be cyclic shifts.

We give a small example of the process for a RAID with five strings and three reliability groups. Figure 2 shows 15 data objects stored on the virtual disks. The first stage shifts the strings cyclically three to the right. ACATS then permutes all five strings separately. Data object **a** which is located on the first disk in the first string is physically stored on the first disk in the third string.

The ACATS scheme stores data objects from arbitrary reliability groups the each physical disk. If a disk fails, we recover its data objects by reading objects in the same reliability group, which are stored on disks within strings not containing the failed disk. Thus ACATS distributes the reconstruction load over all disks beyond the afflicted string. In addition, ACATS maintains the orthogonality between reliability groups and strings and preserves resiliance against string failure. The first stage of ACATS alone already implements distributed sparing and checking. In our example, data objects `d,i,n` could contain check data and objects `e,j,o` could be used as spare data space. Holland's incomplete block designs (BD) [2] spreads the reconstruction load over the disks remaining within the rank containing the failed disk; a *rank* is a row in our terminology. We can implement ACATS without table look-up in contrast to BD. BD achieves an almost perfect load balancing within a rank, whereas ACATS comes very close and involves almost all disks within the array.

V. RAID Reliability

Fast replacement of failed disks and strings has a strong positive effect on the reliability of a disk array. We refer to this process as *repair* and designate its duration as

TABLE I
EVENT RATES

event		rates (per hour)
essential component failure		1×10^{-7}
disk drive failure		2×10^{-5}
string failure	soft	2×10^{-5}
	hardened	5×10^{-6}
	super-hardened	5×10^{-8}
component repair		2.78×10^{-2}

repair time. We use hot stand-by spare disks and strings to emulate almost instantaneous repairs. In case of disk or string failure, the redundancy in the storage system is used to reconstruct the data stored on the failed unit and to store it on the replacement unit, which then takes over from the failed unit. This reconstruction process can take as little as 20 seconds for a disk failure to 120 seconds for a string failure, so that the chances for another, unrelated failure hitting during this time interval is very low.

Distributed sparing [7] increases the performance of the RAID during normal operating conditions and helps achieve the vital short data reconstruction times. Instead of having a dedicated spare string or disk, the equivalent amount of space is distributed over all disks in the RAID.

We determine RAID reliability by calculating the RAID Mean Time To Data Loss (MTTDL). Strings tend to be the most vulnerable components but can also be most easily hardened. Accordingly we distinguish three varieties of strings: *soft strings* with a minimal set of components, *hard strings* that contain backups for cooling and power supply and *super hardened strings* where each component, except the disk drives, has a backup. The disk drives are the nextmost vulnerable components. Mean Time To Failure (MTTF) values for disk drives are rapidly increasing and for top-of-the-line disk drives are approaching 100 years. Because disks contain mechanical parts, these improvements seem to be limited. We use a conservative MTTF of 5.86 years for our calculations here, partly, because this number has been used in the literature [8]. Disks used in supercomputers sometimes have such a high failure rate because performance is maximized, e.g. by higher rotational speed. In any case, our methods are easily adaptable to different MTTF values and we have observed that our results stay qualitatively valid for all MTTF rates. Finally the RAID central components, the cache and the controller, are vulnerable too. Failure of a disk or a string triggers repair during which we exchange the failed component and use the storage redundancy to load the lost data onto the exchange component. We base our sample calculations on a repair time of 36 hours and thus model a repair at the beginning of the next workday. Faster repair times might entail high personnel costs. We give our assumptions on component failure and repair rates in Table I.

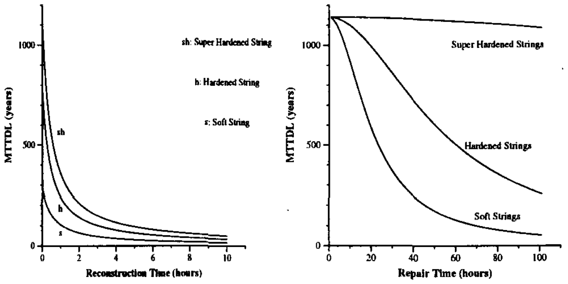

Fig. 3. Sensitivity analysis for the classic Level 5 RAID

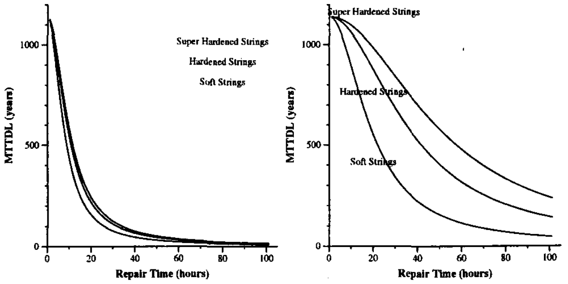

Fig. 4. Sensitivity analysis for repair time for the MDS RAID with ACATS (left) and the classic MDS RAID.

We derive MTTDL figures for the Level 5 RAID with a full, distributed spare string and for the MDS RAID. In both cases we investigate the classic organization and the organization with ACATS. All our RAIDs contain 55 disks, organized into five reliability groups and eleven strings as in Figure 1. For the Level 5 RAID, nine of these strings are message strings, one is the check string and one string serves as the spare string. For the MDS RAID, rather than one check and one spare string, we have a pair of check strings. If we provide a spare disk within the MDS RAID, this disk is evenly distributed over the whole RAID by setting 1/55 of each disk aside as spare space. This single spare disk protects efficiently against disk failures but reduces our storage capacity slightly.

We model the various states of a RAID that results from component failure in a continuous Markov model. We obtain an expression for MTTDL from the inverse of the Markov state transition matrix utilizing Laplace transform techniques. This method is computationally expedient; our estimates for numerical error indicate that the results are valid for at least six decimal digits. We limit the number of states within our models to ensure small error at worst. We cannot present the models here due to space limitations, but we present an overview of their structure. All the details can be found in [1]. Our model for the classic Level 5 RAID with a distributed string of spare disks contains ten states representing various numbers of disk failures, six states representing a string failure and various numbers of failed disks (0-5) and another state

TABLE II
MEAN TIME (YEARS) TO DATA LOSS

RAID	MTTDL Super	MTTDL Hard	MTTDL Soft
Level 5	1134	777	286
Level 5 & ACATS	1119	497	158
MDS	755	581	260
MDS & ACATS	90	80	55
MDS & one spare	983	941	613
MDS & one spare & ACATS	821	760	491

designating two failed strings. The classic MDS RAID is investigated with a model containing 27 states. Most of these describe the RAID after various disk failures and how these disk failures are located in reliability groups. We count the number of reliability groups with one and with two disk failures to define a state. To describe the MDS RAID with a distributed spare disk, we have to add three states to the Markov model. The Markov models for the RAIDs with ACATS are considerably smaller, because ACATS attempts to spread all reliability groups over all 11-combinations of disks. We give the results of our calculations in Table II; our parameters are in Table I.

We can see that in all cases ACATS diminishes reliability. There are two reasons. First while ACATS diminishes the time needed for data reconstruction on a spare after a failure, the difference is not enough to have a significant influence on the reliability. However ACATS limits increases in disk utilization after a single disk failure, as we will see. Secondly ACATS makes RAIDS more sensitive to disk failure than does the classic organization: The classic RAID organization can withstand several disk failures for which no spare space has been provided, as long as they are distributed over many reliability groups. In contrast, a RAID with ACATS and without spares loses data after failures of or in two strings.

MDS based RAIDs show clearly inferior reliability in Table II. This at first surprising fact is easily explained. If two disks fail in the same reliability group in an MDS RAID, any further failure in this group leads to data loss. In a Level 5 RAID however, the data on these two failed disks is replaced on spares, so that any further disk failure and most string failures do not lead to data loss. We can improve MDS RAID reliability by setting one disk aside as a spare, though this lowers the storage capacity. Later, we will introduce a similar approach, referred to as *reconfiguration* into our MDS RAID schemes that gives MDS RAID the same reliability without changing the hardware component configuration or the storage capacity.

From the sensitivity analysis given in Figures 3 and 4, we can conclude that repair time has a major influence. If data reconstruction on spare space is a matter of hours, reliability is very low. On the other hand, if reconstruction takes very little time (20 - 500 seconds), the reconstruction time has no discernible impact on reliability. In general a reconstruction that takes five minutes only shaves a few percent from the reliability of Level 5 RAIDs. By hardening strings or increasing disk MTTF, we can reach reliability levels where the failure of central RAID components becomes the most significant failure mode.

To achieve the same reliability for MDS RAIDs as for Level 5 RAID organizations we can use *reconfiguration* within MDS RAIDs: If a reliability group has lost two disks, then one check disk of another reliability group that has not lost a disk yet is used to replace one disk in the "degraded" reliability group. In a RAID with ACATS a disk failure affects different reliability groups and reconfiguration may not be necessary for all tracks. To emulate the use of the spare string completely, the disks that are being reassigned from one reliability group to another are those located on the same string as a failed disk. For performance purposes, all RAIDs use a different form of reconfiguration: If a message disk has failed and if no spare disk is available, then the data is reconstructed on a check disk. Both reconfiguration processes do not pose a danger to data safety because the moved data is buffered in the non-volatile cache. Furthermore the speed of reconfiguration is not as important as the speed of reconstruction of lost data on spares. By lowering the rate of reconfiguration, we limit its disk utilization increase. An MDS RAID scheme with reconfiguration achieves in our model the same reliability numbers as the corresponding Level 5 RAIDs.

Our Markov model does not reflect the possiblity of related component failure. While we do not know any hard data to model related component failure, the high reliability of the RAIDs (hundreds of years) makes even remote failure modes important. There is convincing evidence [9] that system activity influences failure times in computer systems. As the effect of a component failure is a flurry of activity in a RAID, it is reasonable to assume that related failures will occur in a RAID. An MDS RAID can withstand substantially more double failures than a Level 5 RAID. This better reliability is not reflected in our model, because of the difficulty of making realistic estimates for related failure. Another difference is the small reduction of Level 5 RAID reliability due to actual non-zero data reconstruction times. Consequentially we see MDS based RAIDs showing slightly higher reliability even without taking related component failure into account.

VI. PERFORMANCE

Our analytical performance modeling assumes a transaction-only environment of random single, small accesses to disks. While the results give a valid assessment of the performance potentials and especially the relative merits of schemes, this is not the only important environment. If large blocks of data are accessed at the same

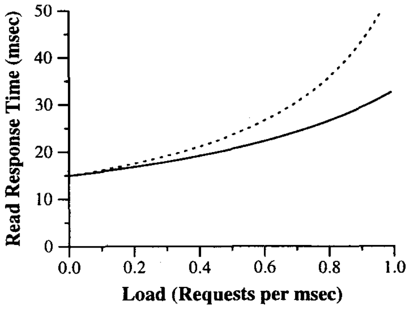

Fig. 5. Read Response Time for the Level 5 RAID with Distributed Sparing and the MDS RAID (dotted)

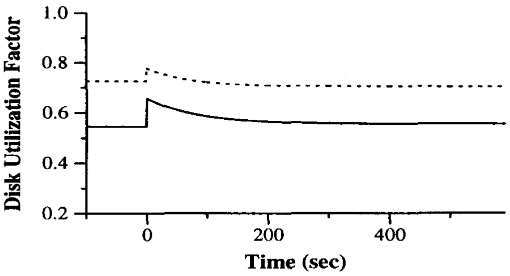

Fig. 6. Disk Utilization Factor after a Disk Failure in the Level 5 RAID with Distributed Sparing (solid) and the MDS RAID (dotted) with ACATS.

time, a technique called striping can be applied. Striping in general defines a linear storage space encompassing all storage areas in a RAID. With a good striping scheme we can update large contiguous storage areas in this linear space by using a faster write operation, in which no Δ values are needed, because the check information is calculated directly from new message data. Striping can thus reduce the weight of the write operations and boost for certain environments the performance figures. If we model a scientific computing environment with few, but large processes, we should use closed queueing systems as opposed to our open queueing systems.

Our RAIDs contain 55 disks organized in eleven strings and five reliability groups. We base our service time estimates on a latency of 7.5 ms and an average seek time of less than 7 ms. We do not assume any controller overhead. Our read service time is estimated at 15 ms and the write service time for an individual disk at 30 ms. During an update, there will be three of these writes within MDS RAIDs and two within Level 5 RAIDs. We assume a read to write ratio of 2:1. The expected service demand is then 30 ms for the Level 5 RAID and 40 ms for the MDS RAID. If the RAID load is Λ (per millisecond), the utilization at each disk is 0.600Λ for the Level 5 RAID without distributed sparing; here we use only 50 of our 55 disk drives. The utilization at each disk is 0.545Λ for the Level 5 RAID with distributed sparing; we use 55 disk drives in this configuration. For the MDS based RAID the disk utilization is 0.727Λ; here we use all 55 disk drives. The poorer performance of the MDS RAID is explained by: (1) the impact of an additional disk access during a write and (2) the beneficial impact of distributed sparing for Level 5, which has the individual disk at only 10/11 of capacity if the RAID is fully used. The latter effect causes the lower utilization of the Level 5 RAID with distributed sparing as opposed to the Level 5 RAID without distributed sparing, which uses only 50 disks.

We illustrate the impact of these different utilizations by giving the read response times for the Level 5 RAID with distributed sparing and the MDS RAID depending on the load at the RAID in Figure 5. The shape of the curves is typical for response times depending on load.

After a single component failure, all data is still available. The data reconstruction process in the Level 5 RAID gathers data from all disks in the reliability group containing the failed disk, reconstructs lost data, and then stores it in spare space. An indespensible part of the reconstruction process is *forced reconstruction*, in which the RAID controller issues track reconstruction demands. Some off-the-shelf disks can perform track operations, referred to as on-arrival-caching, in which the first full segment passing under the head is accessed. We can lower the total reconstruction load by redirecting writes and piggy-backing reads to the failed disk [3]. Write redirection stores the data of an update to an unreconstructed segment immediately on the spare disk. Read piggy-backing not only reconstructs lost data, but also stores the data on the spare disk. This *opportunistic reconstruction* is difficult to model analytically, because we have to make assumptions regarding accesses to the failed disk. Our simple model gives nevertheless some insight into the reconstruction process. The solid line in Figure 6 shows the impact of using only opportunistic reconstruction after a disk failure on the disk utilization. In Figures 6, 7, 8, and Table III we report *disk utilization factor*, that is, the disk utilization divided by the load Λ. The peak immediately after failure, occurring at time 0, is caused by access demands to the failed disk. Using opportunistic reconstruction increases the utilization and leads to a rapid decrease in utilization. The shape of the curve is determined by the access pattern to the failed disk. At the end of the reconstruction process, the utilization is slightly increased compared to the *status quo ante*. Using forced reconstruction we can stabilize maximum disk utilization at the peak value and terminate the reconstruction process sooner which increases RAID reliability by restoring data storage redundancy quicker. Alternatively we can increase the utilization during the reconstruction process.

The MDS RAID does not use sparing. A reconfiguration process uses the data redundancy to reconstruct message data from the failed disk and store them at one of the check disks in the reliability group. The disk utilization is given as the dotted line in Figure 6. Because the

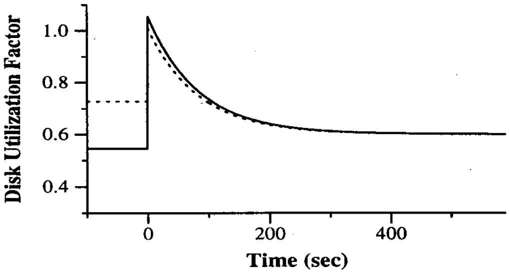

Fig. 7. Disk Utilization Factor after a String Failure in the Level 5 RAID with Distributed Sparing (solid) and the MDS RAID (dotted) with or without ACATS

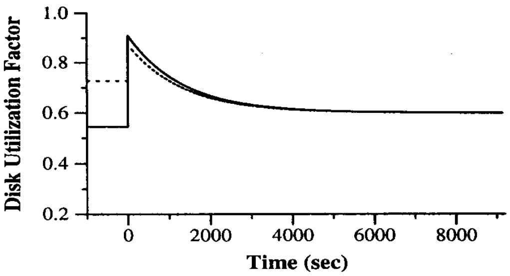

Fig. 8. Disk Utilization Factor after a String Failure in the Level 5 RAID with Distributed Sparing (solid) and the MDS RAID (dotted) with or without ACATS using segment reconstruct.

reconfiguration process does not reconstruct check data, the absolute utilization increase is smaller than for Level 5 reconstruction. After the reconfiguration process is terminated, the disk utilization is lower, because one reliability group now contains one disk less. If there would be five disk failures in a row, the MDS RAID and the Level 5 RAID would both be reorganized to a Level 5 RAID without sparing and the disk utilization would then be 0.600Λ.

Figure 7 depicts the disk utilization after a string failure, which is identical to the disk utilization after a disk failure in the classic organizations. Because the MDS RAID reconfiguration involves less work, the utilization after failure is actually lower than in the Level 5 RAID. The disk utilization factor after the end of the reconstruction or reconfiguration process is equal for both RAID organizations, namely 0.600Λ. Table III presents the utilization factor data.

The peak utilization after string failure determines performance guarantees at a RAID throughout episodes of component failure. A performance improvement can be gained through a less aggressive reconstruction process, based on opportunistic reconstruction of (fixed sized) segments only. Figure 8 presents the improved utilization factor data for the scheme. The performance improvement however has significant costs in reliability as the reconstruction time now is measured in hours. We base the calculations on 16 segments per track. The performance guarantee of the MDS RAID is still better. The long reconstruction/reconfiguration time makes it advisable to switch to track reconstruction once a sufficient portion of the failed disk or disks have been reconstructed and then use forced reconstruction to keep the disk utilization level at the peak value. If the RAID load varies considerably, other approaches can be used such as reconstruction during low load times, but the Level 5 RAID will experience decreases in data security through drawn-out data reconstruction processes in contrast to the MDS RAID.

TABLE III
PEAK DISK UTILIZATION
FACTOR BEFORE AND AFTER DISK FAILURE

RAID	Normal	Peak
Level 5	0.545	1.055
Level 5 & ACATS	0.545	0.657
MDS	0.727	1.090
MDS & ACATS	0.727	0.778

VII. CONCLUSIONS

The classic Level 5 RAID and the MDS RAID with the reconfiguration protocol provide the highest reliability. In the fault-free state, the Level 5 RAID shows the best performance but in the string failure state, the MDS RAID has a small performance advantage.

Declustering methods such as ACATS have a negative impact on reliability but have indeed better performance immediately after a disk failure. ACATS has no impact on performance immediately after a string failure.

REFERENCES

[1] T.J.E. Schwarz, "Performance and reliability of disk arrays," Ph.D. thesis, University of California, San Diego, June 1994.

[2] M.C. Holland, "On-line data reconstruction in redundant disk arrays," Ph.D. thesis, Carnegie Mellon University, 1994.

[3] R.R. Muntz and J.C.S. Lui, "Performance analysis of disk arrays under failure," *Proceedings of the 16th VLDB Conference*, Brisbane, June 1990, pp. 162-173.

[4] W.A. Burkhard and J. Menon, "Disk array storage system reliability," *Digest of Papers The Twenty-Third International Symposium on Fault-Tolerant Computing*, Toulouse, 1993, pp. 432-416.

[5] T.J.E. Schwarz and W.A. Burkhard, "RAID organization and performance," *Proceedings of the 12th International Conference on Distributed Computing Systems*, Yokohama, 1992, pp. 318-325.

[6] D.E. Knuth, *The Art of Computer Programming, Volume 2 / Seminumerical Algorithms*, Addison-Wesley, 1969.

[7] J. Menon and R.L. Mattson, "Distributed sparing in disk arrays," *Proceedings of the COMPCOM Conference*, San Francisco, 1992, pp. 410-416.

[8] G.A. Gibson, "Redundant disk arrays: reliable, parallel secondary storage," Ph.D. thesis, University of California, Berkeley, 1990.

[9] R.K. Iyer, D.J. Rosetti and M.C. Hsueh, "Measurement and modeling of computer reliability as affected by system activity," *ACM Transactions on Computer Systems*, vol. 4, 1986, pp. 214-237.