Tape Group Parity Protection

Theodore Johnson
johnsont@research.att.com
AT&T Laboratories
Florham Park, NJ

Sunil Prabhakar
sunil@cs.purdue.edu
Purdue University
West Lafayette, IN

Abstract

We propose a new method of ensuring the redundant storage of information on tertiary storage, especially tape storage. Conventional methods for redundant data storage on tape include mirroring (storing the data twice) and Redundant Arrays of Inexpensive Tapes (RAIT). Mirroring incurs high storage costs, while RAIT is inflexible because special hardware is used and all tapes in a stripe must be loaded to read a file. Our method, Tape Group Parity Protection (TGPP), writes parity data asynchronously. The data area in each tape is divided into fixed-size regions. *A collection of tape regions from different tapes are collected into a* protection group. *Parity data of the tape regions are computed, stored, and eventually migrated to tape. By using the disk cache to buffer the unmigrated parity data, all write operations can be performed asynchronously. Because each tape can be written independently, no special hardware is required to read the tapes, and individual tapes can be manipulated and transferred. In this paper we discuss TGPP, alternatives for managing TGPP groups, and their relative performance.*

1 Introduction

In this paper, we propose a new method of ensuring the redundant storage of information on tertiary storage, especially tape storage. The method we propose is less expensive, more flexible, and has higher performance in many circumstances than previously proposed methods.

Tertiary storage systems built from tapes and tape drives managed by a robotic storage library can provide massive storage at prices that are one to two orders of magnitude less than on-line storage. Tape storage is often used as an inexpensive backup for on-line storage, increasing the reliability of computer-stored data by providing a redundant storage location.

Hierarchical storage management (HSM) systems use tape storage to greatly expand the capacity of a fixed disk based file system. Files are migrated from the disk-resident file system to tape storage when the disk-resident file system runs out of space, and files are migrated from tape to fixed disk when they are referenced and are not available on-line. Most files in an HSM are stored only on tape. To ensure the reliability of redundant storage, conventional HSM systems need to make multiple copies of tape-resident data. For example, this strategy is suggested by Veritas [6]. Mirroring tape-resident data is expensive in terms of media consumed, robotic storage library slots used, and tape drive bandwidth required to write duplicate tapes.

An alternative is to apply the RAID technology [1] developed for magnetic disk drives to tape storage. RAID technology (i.e., Redundant Arrays of Inexpensive Disks) works as follows. A collection of N+1 disk drives is aggregated and made to appear as a single disk drive. The user's data is written across N of the drives, and the additional drive stores parity information. In a typical implementation (e.g., RAID level 4 or 5), a user's data is written to the blocks of the data disks in a round-robin fashion, and the parity block is the bit-wise exclusive-or of the data blocks. The throughput of the RAID disk array increases by up to a factor of N because N of the drives can be used simultaneously for useful work. The RAID disk array can withstand the loss of any single drive without losing the user's data, because the lost data can be reconstructed from the non-failed data and the data stored on the parity disk. We note that there are many variations on the choice of parity disk for a given stripe of data blocks.

When RAID technology is applied to tapes, it becomes RAIT (Redundant Arrays of Inexpensive Tapes) [2]. User data is written to N tapes, and an additional tape stores parity blocks (e.g., RAID level 4). A discussion of RAIT products can be found on-line [5] and vendors who sell RAIT products can be found by asking a web search engine for pages that include the word "RAIT".

RAIT technology has several drawbacks, related to the tight connection between blocks in a stripe. Data blocks must be read synchronously from tape to reconstruct the tape-resident file. Therefore, all of the tapes in a RAIT stripe (except for the parity tape) must be mounted before data can be read. The synchronization delays due to tape striping can significantly decrease the performance of the equipment. Several problems arise in accessing data that is striped across multiple tapes [3]. A sufficient number of tape drives to read the entire file must be available when the data is requested, else the request to read the data must block until the drives are available. Special equipment is usually required to implement RAIT. A single tape in the stripe cannot be exported to a different system; all tapes in the stripe must travel as a single unit. Similarly, it is difficult to read striped tapes using conventional equipment, because files are interleaved across multiple tapes and must be reconstructed. Finally, all data in the stripe is lost if two tapes are damaged.

We propose the alternative solution of *tape group parity protection* (TGPP). Like RAIT, TGPP ensures redundancy by writing to tape the parity blocks of a collection of tapes (which ensures a low storage overhead). However, all writes can be performed independently and asynchronously. The independence and asynchrony provide the flexibility of TGPP: no special hardware is needed to read or write data. Individual tapes can be read from, written to, copied, exported, or recycled (though the parity protection might be lost). Damaging two tapes in a protection group does not necessarily mean that all data is lost.

2 Tape Group Parity Protection

The principle of tape group parity protection (TGPP) is simple. As data is written to tape, parity information is collected and stored on-line. When a file of parity data contains contributions from a sufficient number of tapes, it is migrated to tape. In this section, we describe the mechanism for implementing TGPP.

The unit of protection on a tape is a *region*. A region can be as small as a tape block, or it can contain the entire tape. We describe TGPP as protecting regions instead of blocks or tapes for flexibility. The discussion in this paper generally assumes that regions are of a fixed size. We note that a region refers to a unit of data on a tape (e.g., 1 Gbyte) instead of a unit of tape length. The tape length corresponding to a unit of data is difficult to determine due to the automatic masking of bad tape blocks and compression.

A *protection group* is a collection of n *regions* each of length B bytes on different tapes. Figure 1 illustrates a protection group. The maximum number of regions, N, in a protection group is the *group width*. A protection group is *full* when $n = N$, and is *non-full* otherwise. When a protection group is created, $n = 0$. Regions are added to the protection group, up to the maximum of N regions. A protection group is analogous to a disk stripe protected by a parity block in a RAID system, except that the regions as-

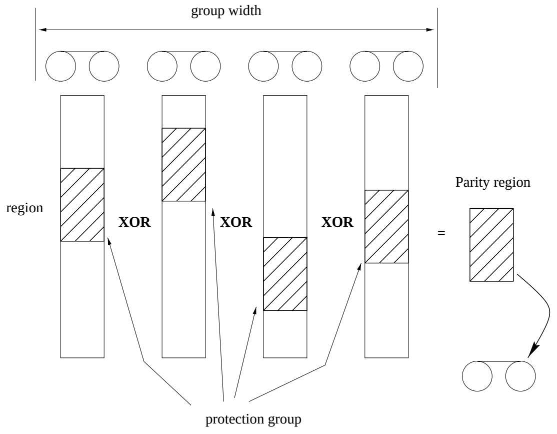

Figure 1: Blocks in a protection group.

sociated into a protection group is not pre-determined and no data striping is implied.

Each protection group has a *parity region* associated with it, which is computed to be the bit-wise exclusive-OR of the data in the n regions. When a region is added to a protection group, it is initially empty. As data is written to a region of tape, the region becomes filled. For the purpose of computing the parity region, the empty portion of a region is considered to be filled with zeros.

When a fresh tape is written to, or when the current region on the tape is filled, a new region is created for the tape. This region is assigned to a protection group. If no suitable protection group exists, a new one is created. When a region or a protection group is created, it is *open*. A region becomes *closed* when the tape is filled or when all bytes in the region have been written. A protection group becomes *closed* when each region becomes closed and no regions will be added to the protection group in the future (e.g., because it is closed).

The parity region of an open protection group is disk resident. As writes occur on a region, the parity region is updated, see Figure 2. The maximum size of the protection region is B bytes, but the zero-filled region does not need to be explicitly represented. When the protection group is closed, the parity region can be written to tape. Probably the simplest option for managing tapes containing the parity regions is to use the HSM tape management. The information about the protection group required to rebuild damaged regions (e.g., the locations of the regions of the protection group and the location of the parity region) are stored in a metadatabase, in the same way that the locations of tape-resident files are stored in a metadatabase.

If a tape is damaged, the damaged regions of the tape

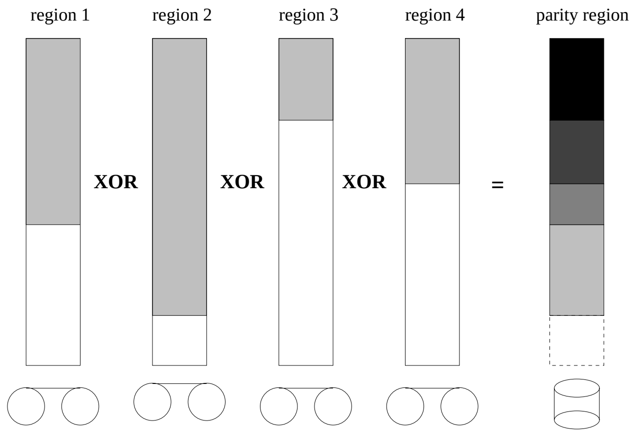

Figure 2: Updating the parity region.

(perhaps the entire tape) can be recovered by taking the exclusive-OR of the undamaged regions in the protection group with the parity region of the protection group. This process is repeated until the damaged portion of the tape is reconstructed. If a tape is exported or deleted from the system, the parity regions of the protection groups on the tape are invalidated, and the protection groups lose their protection. The parity regions can be fixed by taking the exclusive-or of the parity regions with the data from the exported or deleted tape. We note that a similar problem occurs when tapes are compacted to remove holes from deleted files. For this reason, TGPP is most applicable to archives that retain all old versions of files.

The metadatabase that stores TGPP protection group information is a vital component of data protection, and must be protected in the same way that the HSM metadatabase is (e.g., by storage on mirrored or RAID disks, by maintaining second copies, by taking backups). We note that the TGPP metadatabase is separate from the HSM metadatabase. So if the TGPP metadatabase is lost, no HSM data will be lost (although the parity protection might be lost). If the parity regions are written in a self-describing format (i.e., with headers and trailers containing metadata), then the TGPP metadatabase can be reconstructed, albeit slowly.

The files written to tape might span two or more regions. This issue is not a problem, as the parity updates for a file can easily be split to several parity regions. Similarly, a region can contain data from multiple files. However, the management of the parity regions can be made easier if writes to tape are always made in fixed size blocks such that B is an integer multiple of the block size.

The TGPP algorithm is very flexible. There are no requirements that regions in a protection group be written synchronously. Conversely, there is no prohibition against tape striping. The regions that form a protection group can be located on an arbitrary collection of tapes, and the parity regions can be placed on arbitrary tapes. We note, however, that all components of a protection group should be placed on different tapes to ensure redundant storage. Files written to a TGPP protected tape can be read without consideration for protection groups or region boundaries. The connection between regions in a protection group is used only for data recovery. Generally, TGPP can be implemented as a module that co-operates with and uses the resources of the existing HSM.

3 TGPP Management Policies

As mentioned previously, the TGPP algorithm can be adapted to a large number of data management policies. These policies can attempt to minimize storage overhead (for the parity regions of the open protection groups), minimize the number of open (written to but not full) tapes, increase transfer rates, minimize data recovery time, and/or enhance manageability. To achieve these goals, we can adjust the region size, the policy for forming protection groups, and the policy for writing parity regions to tape.

In this section, we explore some alternative TGPP management strategies. We note that different collections of files can be written to tape with different TGPP management strategies. Many HSMs provide *tape families*, which are a collection of tapes dedicated to a collection of related files[1]. The use of tape families allows the system administrator to tailor migration, storage, and archival policies to individual projects. Two of the TGPP management strategies that we propose are designed to work with the HSM tape families.

In order to determine the best TGPP management policy, we provide some back-of-the-envelope performance computations, and use a discrete time simulation model. The primary costs of TGPP are the on-line storage overhead for parity regions, and the cost to reconstruct a damaged region or tape. We compute the number of open protection groups (which is proportional to the storage overhead), and the number of tapes that must be accessed to reconstruct a damaged tape (which is proportional to the reconstruction cost). We note that the number of tapes that must be accessed to reconstruct a damaged region is always N, the protection width. A secondary cost is the number of concurrently open tapes (written to but not full) that must be maintained.

Let:

[1] Tape families are sometimes called *tape groups*, but we use "tape families" to avoid confusion with protection groups.

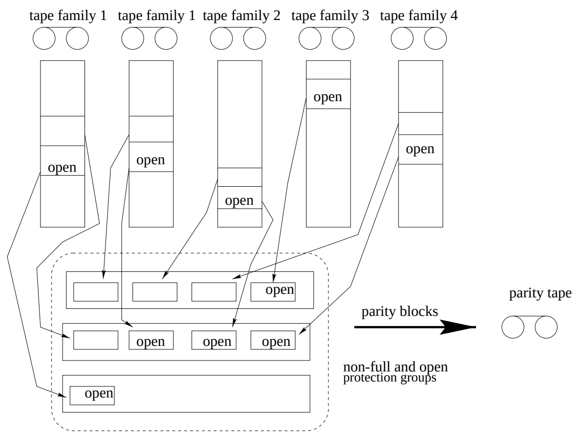

Figure 3: The Immediate TGPP management strategy.

- F be the number of tape family defined within the HSM.
- N be the protection width.
- R be the number of regions per tape.
- D_T be the tape size.
- D_R be the region size. $D_R = D_T/R$.
- N_P be the number of open protection groups.
- D_P be the on-line storage space for open protection group parity regions. $D_P = N_P * D_R$.
- N_R be the number of tapes accessed to reconstruct a damaged tape.

3.1 Immediate

The Immediate strategy tries to fill open protection groups as quickly as possible (See Figure 3). The protection width is set to N. The system keeps a list of non-full protection groups (those with less than N regions assigned to them). Whenever a new region opens, it is assigned to a non-full open protection group such that no other region from the same tape is a member of the protection group. If no such protection group exists, a new one is created. As soon as a protection group is closed, it can be written to tape. A single tape group, with a single open tape, is used to store the parity regions.

The immediate strategy attempts to minimize the storage overhead of implementing TGPP. We can make a simple worst-case analysis of the storage overhead. A region on a tape can join a protection group as long as no other region from the same tape is part of the protection group. Therefore, the maximum number of non-full protection groups that contain only closed regions is R. In addition each group might have an open region that prevents a protection group from closing. In total, the space overhead of the Immediate strategy is

$$D_P \le D_T + F * D_R = D_T(1 + F/R)$$

Let us next compute the number of tapes that must be mounted to rebuild a damaged tape. Each of the R regions on a tape is associated with $N-1$ other regions and a parity region In the worst case, each of the regions and parity regions are stored on different tapes, so that

$$N_R = N * R$$

While crude, these formulas show the tradeoff of the region size D_R. As D_R increases, more temporary storage is required for the parity regions. However, R decreases, so fewer tape mounts are required to rebuild a damaged tape.

The formulas that we derive are rather pessimistic. For example, some protection groups will have several open regions. If several groups receive data at about the same rate, they will tend to be matched in the protection groups. Therefore each protection groups for the regions on a tape will tend to be stored on the same set of tapes.

To better understand these potentially complex interactions, we wrote a simple simulator. Whenever data is migrated, each tape family $f = 1 \ldots F$ has probability P_f of receiving the data. Each tape family has at most one open tape, and each tape will hold exactly R regions. A random number between 1 and r_f regions of data are migrated to the selected family f. We execute the immediate policy for selecting the protection group for every newly migrated region. The last region migrated to a family is open, and becomes closed on the next migration of data to the family.

We ran an experiment in which each tape family received data at the same rate, and in 1-region chunks. We varied the protection width N and the number of regions per tape R. In Figure 4, we plot the number of tape mounts required to rebuild a damaged tape N_R against the protection width N for varying numbers of regions per tape R and 40 tape family. The chart shows that $N_R \approx N * R$ for moderate values of R, but that N_R approaches an asymptote as R becomes large. The asymptote that N_R approaches is approximately $2F$, as is shown in Figure 5.

In another experiment we measured the number of open protection groups as we varied the number of tape groups, the number of regions per tape, and the protection width. We found that the average number of open protection groups varied little with R, but did depend on F and N. In Figure 6, we plot the number of open protection groups as we vary the protection width for different values of F and $R = 10$. As expected N_P increases with F. The number of open protection groups decreases with N, because each protection group can admit more regions.

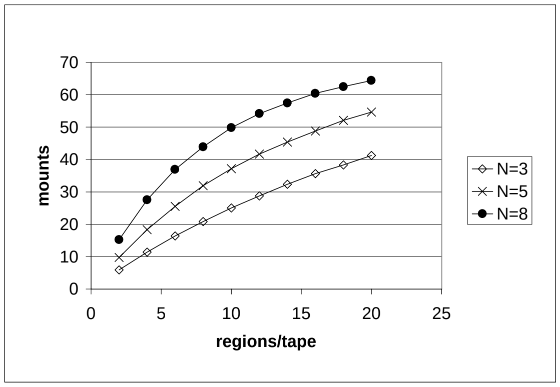

Figure 4: Number of mounts to rebuild a damaged tape vs. R, 40 tape groups.

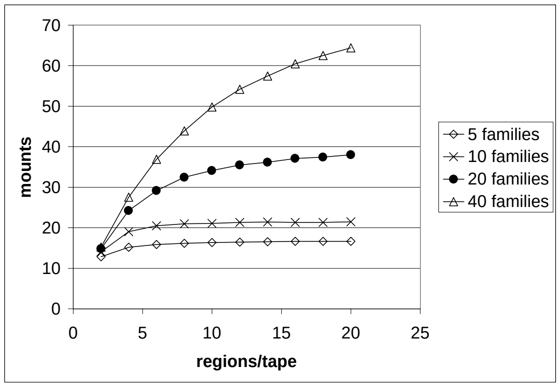

Figure 5: Number of mounts to rebuild a damaged tape vs. R, protection width = 8.

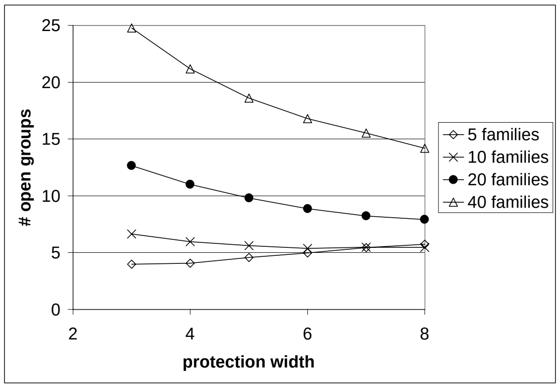

Figure 6: Number of open protection groups vs. N, R = 10.

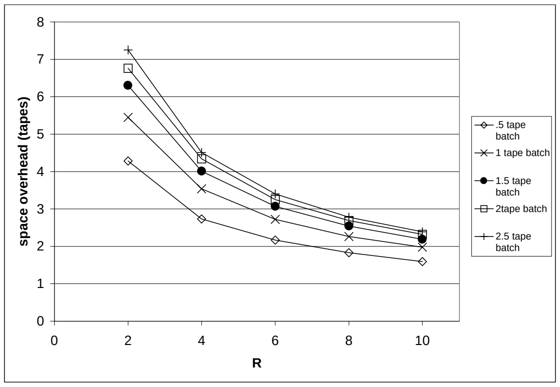

Figure 7: Space overhead (in units of 1 tapes worth of data) vs. R for different data batch arrival sizes.

In the previous experiments, data arrives at the tape families in small batches (i.e., one region or less). We ran a set of experiments to investigate the interaction of the batch arrival size and the and number of regions per tape. The migrated data arrives in batches of k regions at a time. We found that N_R increased slightly, but that N_P increases significantly. In Figure 7, we plot the space overhead of TGPP (in units of tape sizes) versus R for different batch sizes (again expressed in units of tape sizes). The space overhead increases with the batch size, but the increase decelerates rapidly as the batch size becomes larger than one tape worth of data. The space overhead decreases as R increases (as is expected), but not quite linearly anymore.

We ran another set of experiments in which 60% of the data was migrated into 20% of the tape groups. However, the results do not change significantly.

3.2 Protection Set

The time to rebuild a damaged tape can be minimized by minimizing the number of tapes that must be accessed during the rebuild process. One way to accomplish this goal is to use a *protection set*. Regions from the N tapes in a protection set participate in protection groups only with other tapes in the protection set. The parity regions are written sequentially to tape as contiguously as possible. This policy ensures that to rebuild a damaged tape, only the other $N-1$ data tapes and a few parity tapes must be read.

Because of compression and unusable areas on tape, each tape in a protection set is likely to have a different capacity. Unfilled regions on a tape are considered to be zero-filled for the purpose of computing the parity region. The parity set algorithms will create protection regions as long as at least one tape contains a region of data. However, system management might decide to close a non-full tape when all other tapes in the protection set are closed.

3.2.1 Single Tape Family

The single tape family oriented strategy attempts to simplify tape management. Every protection set contains tapes from a single tape family. Furthermore, the tape family is written to a *slice* of tapes of width s, where s divides the protection width, N evenly and $w = N/s$. Tapes in a slice should be filled at about the same rate, whether by tape striping techniques, or by writing files to tapes in the slice in parallel streams and allocating new files to the least-filled tape.

The advantages of the single tape group strategy are to minimize the rebuild time (since $N_R = N$) and to simplify tape management. A protection set contains data from a single tape family, written during a particular period of time. It is likely that all of the tapes in the tape set will be removed from the robotic storage library, or exported to another system.

If the slice width equal the protection width ($s = N$), then each tape in the protection set should be written to at about the same rate. Therefore, protection groups will close shortly after being opened, and can be quickly migrated to tape. Setting the slice width to the protection width increases the likelihood that all tapes in the protection set will be exported or retired at the same time.

If $s = N$, the protection groups will tend to close quickly, and only one or two protection groups will be open per tape family. That it, $N_P \approx 2F$. However, each tape group must have $s+1 = N+1$ open tapes at any time. If $s < N$, then each tape group has fewer open tapes, but in general there will be R open protection groups per tape group, so $N_P \approx 2RF$.

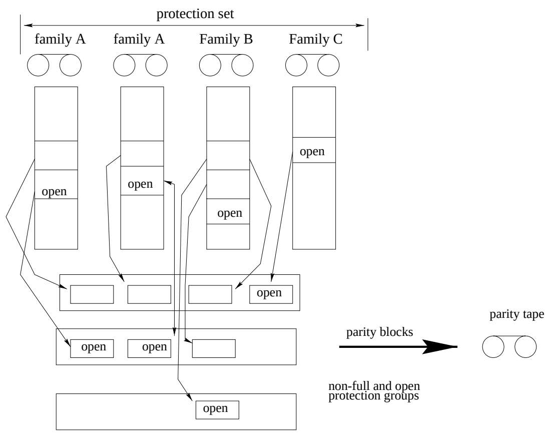

Figure 8: The Multiple Tape Group protection set TGPP management strategy.

3.2.2 Multiple Tape Families

The disadvantage of the single tape family strategy is that some tape groups might not experience generate enough migratable data to justify the cost of keeping $s+1$ open tapes for the tape family. In this case, slices of several tape family can be combined into a protection set. See Figure 8. Whenever a new tape (or stripe) is written to, the tapes are assigned to protection sets that do not yet have k tapes in them. Some tape family might be constrained to enter protection family only with each other (for example, to avoid combining in a protection set a tape with highly compressible data and a tape with uncompressible data).

The Multiple Tape Groups strategy ensures that only N tapes must be mounted to rebuild a damaged tape, and does not require a large number of open tapes per tape group. However, some tape groups might receive few updates and cause some protection sets to store a large number of parity regions on-line. To analyze the expected number of open protection groups, we can observe that the Multiple Tape Groups is similar to the immediate strategy in which $R = 1$, with open protection groups replaced by open protection sets. The expected number of open protection groups when $R = 1$ is similar to that shown for $R = 10$ in Figure 6, except that the number of N_P ranges from 3.4 to 2 when $G = 5$ and from 6.3 to 4.9 when $G = 10$. We note that parity regions of closed protection groups of open protection sets can be written to tape, so each open protection set represents substantially less than R open regions.

3.3 Combined Strategies

The relative performance of the TGPP management strategies suggest that a combined approach is best. The Single Tape Group strategy is best applied to the tape groups with a high ingest volume. Tape groups with a lower ingest rate are best managed with the Multiple Tape Groups strategy. The tape groups with the lowest ingest rate are best managed using the Immediate strategy. Although the tapes managed by the immediate strategy might require a large number of tape mounts for rebuilding, only a few such tapes will be produced because of the low ingest rate.

4 Implementation Considerations

TGPP is designed to work with the existing HSM to provide the parity protection. Therefore, the details of the HSM implementation affect how TGPP can be implemented. One example is the determination of region boundaries. Some HSMs keep track of individual blocks (or sequences of blocks) that are written to tape. If fixed size blocks are used, determining the region boundary is done by counting the blocks written since the last region boundary. Other HSMs write files to tape (e.g., using a tar format). In this case, region boundaries are best determined before the file write begins.

A tight integration between TGPP and the HSM implementation can yield significant performance improvements. One example is the computation of parity regions. Ideally, this would be done as the data is written to tape so that the in-memory image of the data can be used. In the best case, the single tape family strategy is used with a slice equal to the protection width. If the tape writes are kept reasonable in synch, the parity regions can be generated without additional disk reads.

The method in which files are chosen for migration can interact well with the policy for forming protection groups. A typical HSM migration strategy is to wait until on-line storage utilization reaches a high-water mark before starting migration. On-line files are ranked in order of their migratability (See [4] for a discussion of algorithms). We can take advantage of the bulk migration is several ways. First, we can prefer to migrate data that will tend to close open protection groups and open protection sets. Second, we can identify collections of files that will form a closed protection set when migrated. In this case we can again avoid additional disk reads to generate the parity region.

The number of open protection groups in a protection set managed by the Multiple Tape Groups strategy can become large if one or more of tape groups in the protection set does not receive any migrated data for a long period of time. In this case, these tapes can be considered closed for the protection set (and the unwritten data is zero-filled) so that open protection groups can be closed and migrated to tape. Subsequently the low-volume tape groups should be put into the Immediate pool.

5 Conclusions

We have presented a flexible and inexpensive method for protecting tape archive resident data, tape group parity protection (TGPP). The advantages of TGPP over writing one copy of the data to a non-RAIT tertiary storage device is the greatly enhanced protection of the tape-resident data (e.g., two tapes must be damaged before data is lost). The advantage of tape group parity protection over writing two copies of data to a tertiary storage is the significant reduction in the space overhead required for ensuring data reliability. For example, a tape group with 4 open tapes requires a 25% space overhead for reliability, as opposed to a 100% space overhead if two copies are written.

The advantages of tape group parity protection (TGPP) over RAIT are:

1. performance – RAIT files must be read and written in a synchronous manner, tying performance to the slowest device and incurring synchronization delays. Files can be read from a TGPP-protected tape by loading and reading the single tape that contains the file. Similarly, writes are asynchronous. Parallel I/O is obtained by concurrent reads and writes of different files. Furthermore, tape striping coexists well with TGPP.

2. flexibility – Tapes in a tape group protected by TGPP can be read (or written) individually. There is no need to wait until N drives are available before data can be read or written. Exporting tapes in a TGPP protected tape group is easy, one needs only to move the tape group to the destination and read them on compatible drives. Typical RAIT tapes require special hardware to read the tape-resident files, or a complex software emulation. A subset of the tapes in a TGPP can be exported, although the remaining tapes lose their parity protection.

3. Enhanced reliability – If two tapes in a RAIT group are damaged, all data in the set of RAIT tapes is lost. In a TGPP group, only the files on the damaged taps are lost, the other files can still be read through conventional means.

We analyzed several TGPP management techniques. The single tape family provides good performance for high-volume tape families, while other TGPP management strategies are appropriate for low-volume data.

Acknowledgements

We'd like to thank Jean-Jacques Bedet for his careful reviews useful suggestions.

References

[1] M. Chen, E. Lee, G. Gibson, R. Katz, and D. Patterson. RAID: High-performance reliable secondary memory. *Computing Surveys*, 26(2):145–185, 1994.

[2] A. Drapeau and R. Katz. Striped tape arrays. In *Proc. 12th IEEE Mass Storage Systems Symposium*, pages 257–265, 1993.

[3] L. Golubchik, R. Muntz, and R. Watson. Analysis of striping techniques in robotic storage libraries. In *Proc. 14th Conf. IEEE Mass Storage Systems*, pages 225–238, 1995.

[4] T. Johnson and J. Bedet. Analysis of the access patterns at the GSFC Distributed Active Archive Center. In *NASA Goddard Conf. on Mass Storage Systems and Technologies*, pages 153–178, 1996.

[5] R. Van Meter. comp.arch.storage faq. http://alumni.caltech.edu/ rdv/comp-arch-storage/FAQ-1.html.

[6] Sun enterprise hsm system administrator's guide, 1997. Sun Microsystems Computer Company, Part No. 805-1625-10, Revision A.