Semantic Data Placement for Power Management in Archival Storage

Avani Wildani, Ethan L. Miller
Storage Systems Research Center
University of California, Santa Cruz
{avani,elm}@cs.ucsc.edu

***Abstract*—Power is the greatest lifetime cost in an archival system, and, as decreasing costs make disks more attractive than tapes, spinning disks account for the majority of power drawn. To reduce this cost, we propose reducing the number of times disks have to spin up by grouping together files such that a typical spin-up handles several file accesses. For a typical system, we show that if only 30% of total accesses occur while disks are still spinning, we can conserve 12% of the power cost. We classify files according to directory structure and see access hit rates of up to 66% for a power savings of up to 52% of the power cost of spinning up for every read in easily-separable workloads.**

I. Introduction

Archival systems are rapidly moving beyond the realm of corporate filing cabinets and into to the realm of personal, and hence cultural, memory banks. Pundits say that we are seeing the first generations that record their entire lives and expect this data to be digitally stored perpetually and inexpensively [18, 28]. According to Gantz *et al.*, 281 exabytes of digital information was created in 2007, and they expect this number to grow tenfold by 2011 [13]. While the majority of this personal data begins life as data in active use, over time people lose interest in vacation photos and scanned receipts and this data ends up unconsciously filed away yet living among actively used data. Although this data is never explicitly archived, accesses to this data gradually start to resemble an archival workload.

Research in archival systems has typically assumed a "write-once, read-maybe" workload. Real traces are hard to come by, so a significant body of work exists using simulators that assume that reads and overwrites are relatively rare. Real workloads, however, may have significant numbers of reads and overwrites in an area of the system that is "hot". Many workloads that are considered archival may be susceptible to a quick change in status from archival to active due to a variety of reasons such as periodic audits, a current event, or a renewed research interest. This bursty activity may be disproportionately costly in archival systems that assume most disks are idle most of the time.

We propose to approach this problem by taking a page from the mental model of memory and using semantic tagging to store similar data across the same sets of devices. Our goal is to group data to reduce the percentage of disk accesses that result in spin-ups and thus increase power efficiency and reliability in a more realistic archival system. We group files into *access groups*, defined as a set of files that are likely to be accessed within a short time of each other. We use both given and automated classifications to create our access groups and find that if the access group is left spinning for 50 seconds after a spin-up, enough subsequent reads are caught by the spinning access groups that the power cost is up to 52% lower than the alternatives of spinning up for every read or always leaving disks spinning.

We categorize workloads that gradually drift towards archival accesses as *archival-by-accident*. Archival-by-accident workloads differ from traditional workloads in that there may be changing subsets of the system that see "active" use while much of the remainder sees a more traditional archival workload with few reads or writes. A pressing example of an archival-by-accident system is the World Wide Web. Recent studies have shown that the top 10 websites account for 40% of web accesses, and the drop off is exponential instead of long tail [17]. The prevalence of archival-by-accident systems is a strong contemporary motivation for addressing this problem at a broader scale than previous work in hierarchical storage management [14]. We also know that typical storage systems can not cope with the scale of archival data [2]. Handling these heterogeneous and archival-by-accident systems is going to be the next challenge for designers of archival systems.

One of the primary features of an archival-by-accident workload is that a portion of the data could be in active use while most of it remains dormant. We see an example of this in Section V-A, which demonstrates how web automated search indexers can cause a cluster of reads that bombard an otherwise rarely accessed, archival-by-accident system. This paper examines a variety of different groupings to show how group size and makeup affects the power footprint. The remainder of this paper covers relevant background, our system design, and our preliminary experimental results accompanied by a discussion of current research directions.

II. Background

Power management for mobile systems and sensor networks relies on spinning the disk as little as possible to conserve scarce resources. A significant body of work shows that caching, and from there data arrangement, has a strong impact on the energy footprint of single-disk systems. [9, 11, 23]. Helmbold *et al.* point out that for mobile systems, an adaptive disk spin-down rate is superior to a fixed rate, and we intend to incorporate adaptive disk spin down into future work [15].

Conserving power in archival storage systems is another well researched problem. The pioneering project in this area was MAID, which leaves disks idle when they were not in use [5]. MAID, however, works best when accesses are few and far between, so the power saved offsets the increased power consumption and loss in disk lifetime from having more spin-ups. Localizing related data and metadata on the same cylinder group to reduce fragmentation was a part of original Berkeley Fast File System [19]. Essary and Amer provide a strong theoretical framework for power savings by dynamically grouping blocks nearby on a disk [11]. Other predictive methods have shown good results by offering the choice of "no prediction," allowing a predictor to signal uncertainty in the prediction [1]. We extend this work by providing a realistic prediction mechanism and semi-permanent groupings, reducing the need for constant prediction. Paris *et al.* show that files in most workloads have stable access patterns [21].

Though workloads such as HPC are known to be bad candidates for any technique that exploits idle time [3], Narayanan *et al.* show that significant idle periods exist in enterprise workloads, indicating that our technique may apply outside archival systems [20]. They go on to say that access stability could be exploited by grouping similar files together. Performance has been the primary focus for a number of systems that have looked at access history and frequency [7, 26].

Several projects incorporate caching into archival storage. These caches can be based on temporal locality [5] or file or block-level popularity [22]. Studies show that for systems with low request rates, caching reads and writes significantly improves the power efficiency of the system [20]. One major vulnerability of our methodology is that a small group of files in disparate access groups might start getting accessed repeatedly for a burst. In this case, we could follow the lead of PDC [22] and use a multi-queue caching scheme to keep popular files or access groups in a persistent, always-on cache. The impact of caching popular access groups could be especially interesting for workloads without regular access patterns, where we expect to not have many periods of low disk activity.

III. System Design

The only physical requirement for our system is that there exist enough devices for groups to be maintained and grown on separate devices. The disks are accompanied by an index server that handles the mapping of files to access groups. Multiple, hierarchical index servers may be required in a larger system. OSDs such as Pergamum are ideal because they have the added ability to use on-board NVRAM for read and write caching, allowing for a smaller main index and write-offloading [27].

After an initial setup period, we split our data into access groups. These groups could be formed around a variety of semantic or incidental labels such as timestamp, filesystem placement, writer, filetype, the authors in a LaTeX document, etc. These groups may be dynamic, so files must be able to move as access patterns change. Section IV goes into more detail of how these groups are determined. Access groups do not share hardware, and when any data is read from an access group the entire group is spun up and left on for a set amount of time. We currently set the time to 50 s [8].

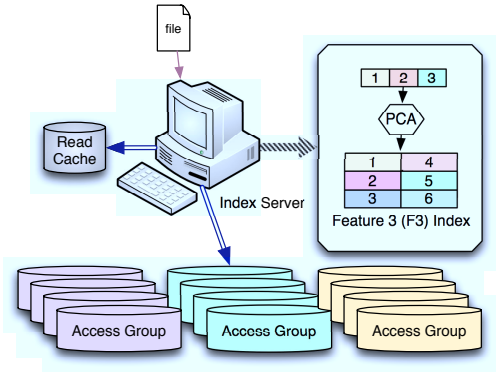


Fig. 1. Typical system components: disks, cache, and index server

There are a few additional, common storage system features that benefit this sort of power management. One obvious concern is that a flash crowd could focus on just one element of an access group, leaving the rest of the group spinning for no benefit. This issue is easily handled with a modest read cache. A simple LRU cache, such as PDC uses, handles this issue [22]. While this is less of an issue in a classic archival workload, a modern archival or archival-by-accident workload needs a level of caching to soften the impact of an influx of accesses to a data set caused by anything from a news event to a unexpected mention in a major blog.

Figure 1 shows the design of a basic system that uses access groupings. The index server stores the mapping between files and access groups and calculates the appropriate access group for an incoming file. If these access groups are stored on OSDs, the index server can treat groups as atomic entities and rely on the OSDs to index files within the group. When data is read, the data is first checked for in the cache. If there is not a cache hit, the index server identifies the access group that the indexed copy of the data is on and sends the read to the appropriate group of disks in addition to copying it into the cache.

Data to be written is sent to the index server. The index server classifies the incoming file or files using techniques such as support vector machines or component analysis to chose the appropriate access groups for the files. If a file group runs out of disk space, either disks are added or the file group is split. The standard approach is to assume that archival systems have enough excess capacity that empty disks will typically be available [16, 29]. However, we have found that smaller access groups have better power savings outcomes. Small groups also make better use of hardware. We intend to explore splitting groups in our future work.

IV. Access Group Classification

The amount of power saved depends on the classification scheme used to sort files into access groups. DGRAID groups similar blocks (defined as being blocks from the same file) on the same device to isolate faults [25]. A similar natural grouping exists between files in a directory or on a configuration path. Several types of similarity between files can lead to an increase in the probability of correlated accesses. For example, the setup files for a given application are likely to always be accessed in fast succession or a programmer working on a project is likely to open up files from the same group in his IDE for the duration of the project. We believe these types of correlations occur throughout real workloads.

We focus on archival-by-accident systems such as personal data or the Internet archive where accesses are more frequent but the overall workload still exhibits archival characteristics [24]. These interactive archival systems have the most potential for gain from any techniques that group accesses and reduce spin-ups, since if the data is effectively never accessed any improvements are very hard to justify after migration or implementation costs are considered.

Certain types of storage systems, such as those used for scientific data or public records, contain data with a relatively small number of natural groupings that correlate with how the data is later accessed. Other workloads contain data that is strongly tied to a particular application, leading to natural groupings of data that are accessed together.

Several schemes have been proposed to identify these groupings. In this paper, we concentrate on identifying potential power savings, so we use labels that are pre-defined in our data set. For demonstration, we also used principal component analysis on the full-text of the data and derived one group ("Site") automatically from one data set.

In future work, we intend to use support vector machine classification. Support vector machines (SVMs) are a common method to classify non-linear data by bisecting the dataset with a series of hyperplanes [10]. One of the primary draws of SVMs is that although they involve solving a quadratic optimization problem, the problem only has to be solved once to form the $O(labels)$ kernel matrix that is used for future classifications. Online SVMs have the ability to incrementally add or remove data and recalculate classifications in $O(1)$ [4, 12]. We expect the workload to change over time, making an algorithm that allows us to cheaply recalculate our classifications attractive. Additionally, since the size of the kernel matrix depends on the size of the label space instead of the size of the dataset, this is a promising technique for keeping the index manageable in a petascale or larger archival system.

V. Experiments

We analyzed two static file access traces to determine the magnitude of power savings to be gained from pursuing this idea. All analysis in this work is at the file level.

The first data set is from the California Department of Water Resources. Our data consists of 90,000 queries to a record store from 2007 through 2009. We make the assumption that queries correspond to record accesses. The data set is pre-grouped, and the grouping labels we consider for each access are "Timestamp", "Site" "Site Type", and "District." These groupings are analogous to directory structure. The dataset provided an additional grouping, "Year", that we chose not to use because it is inconsistently applied.

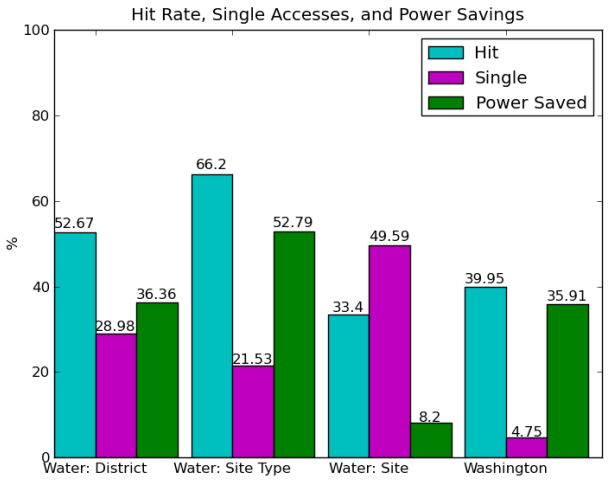


Fig. 2. Power Savings, Hit Rate, and Singleton Rate for Various Classifications

Though many access group classification schemes are valid for our data, we chose to use a pre-defined partitioning for our experiments to focus on the potential for power savings. Our other data set was a database of vital records from Washington state where records are labeled with one of many type identifiers (*e.g.* "Birth Records", "Marriage Records"). We examined 5,000,000 accesses from 2007 through 2010 to a 16.5 TB database. Our goal was to examine the power ramifications of a preliminary implementation of the design proposed in Section III. In particular, we wanted to see if semantic data placement increases power efficiency in a system where disks are in a low-power mode or powered off entirely much of the time.

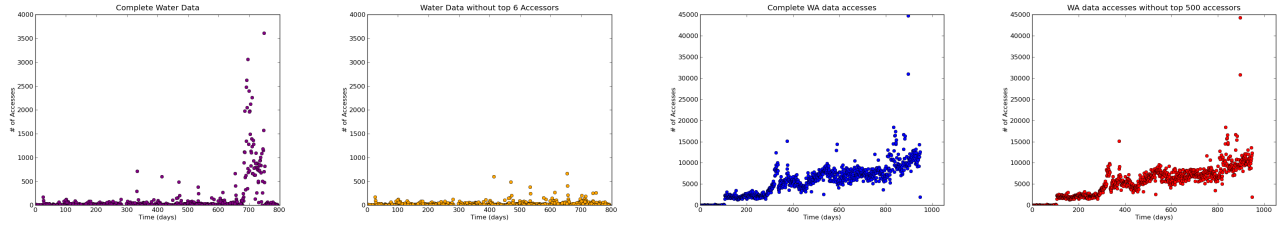
Outside of the minimal overhead of the index servers, we can do a quick calculation to show the value of working in this direction. Suppose a spin-up costs 100 J while leaving the disk on costs 3 J/sec ([6]) and disks are left on 50 s after an access in hopes of catching another access to the same group. In a system without any groupings, 100 accesses would cost:

$$p = 100 \times (100 + (3 \times 50)) = 25000\ J$$

In a system where even 30% of these accesses hit an already on filegroup, the power requirement falls to

$$p \leq [70 \times (100 + (3 \times 50)) + 30 \times (3 \times 50)] \leq 22000\ J$$

The inequality is necessary because we do not know how far into the 50 s period a subsequent access occurs. This shows that 30% hit rate, which we see in our data, results in at least a 12% power savings.



(a) Water (b) Water: Without top 6 (c) WA (d) WA: Without top 500

Fig. 3. Access patterns for the Water and Washington datasets with and without top accessors

Figure 2 shows the percentage of accesses that hit a spinning file group (Hit), the percent that caused a full spin period without a subsequent hit (Single), and the power savings compared to spinning up the disk on demand. While the Water Data Set is only 2.3 GB, the access patterns are still meaningful because they are independent of the data size. This data is also a good test because it attaches predefined groupings such as "Site Type" to each file.

If we break up the data into 7 disjoint groups based on site type and calculate the *hit rate*, defined as the number of accesses made within 50 s of each other, we find that leaving our disks spinning would have covered 63.5% of accesses, as we see in Figure 2. Figure 2 also shows the impact of singles on the power savings, especially on larger groups such as "Site". This is logical since each of those is a demonstration of our worst case, where a single access costs a spin-up and a full 50 s of additional active drive time for no benefit (though this time could be used for scrubbing and offloaded writes).

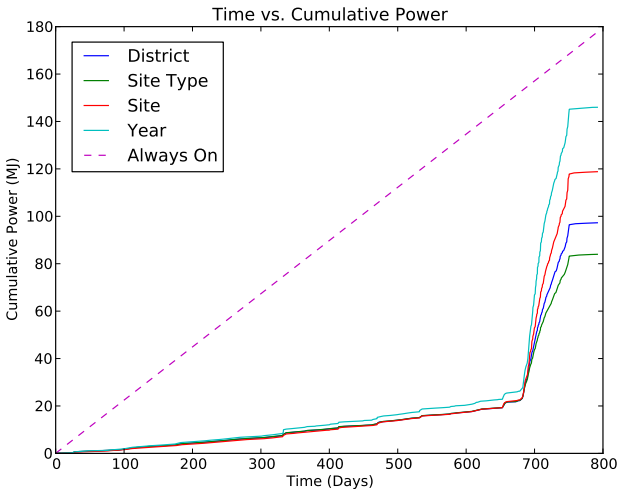


Fig. 4. Power Savings for various groupings versus leaving disks on. *ontime*=1 s, *spintime*=50 s

Figure 2 shows the power saved with our method versus spinning up disks as needed, leaving them on for a fixed period of time, and then powering them off. For a different perspective, Figure 4 shows the power savings over time compared to a system with always-on disks. Though these disks avoid spin-up costs, the batched nature of the accesses make leaving disks spinning a poor method of saving energy. The spike near Day 700 is caused by an influx of queries from a search indexer. The major drawback of this approach is that,

Label	*# of Groups*
District	9
Site Type	7
Site	680

TABLE I
WATER QUALITY GROUP SIZES

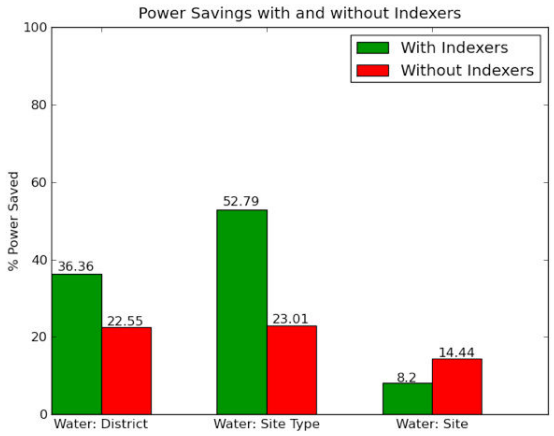


Fig. 5. Effect of Indexers on % of Power Saved.

assuming all disk groups are equal, disks that do not have subsequent accesses will be left on longer than they otherwise would have been.

A. Indexers

Both data sets had strong spikes in their access patterns (Figure 3). Figure 5 shows the effect of removing a small percentage of the highest accessors in the water dataset by selecting the most active IP addresses. Based on `whois` data, these IPs all belonged to a search engine. As we develop this project further, we intend to take into account I/O activity needed for indexing. The Washington dataset's spike is a good example of a case where our method would save a significant amount of resources.

VI. Discussion

Based on this preliminary work, we believe that certain workloads could show significant power savings if files are grouped on disk based on semantic similarities. In addition to archival-by-accident systems, we expect that both disk and tape-based archival systems will also see reduced spin-ups or tape-mounts using access groups, though the magnitude of savings will likely be lower.

Though we do not explicitly model writes, we presume that writes in our archival-by-accident workloads can be offset with the cache located either on the index server or a access local group controller [20]. Writes will then be distributed among the disks in the access group when the access group is spinning for a read. We also recommend a read cache to handle repeated, consecutive accesses to a single file, which reduces the power footprint of a flash crowd. We expect that the overall performance impact will be limited after groups are determined. Also, any computation for classification could be distributed across the nodes in an OSD cluster.

In this paper, we described archival-by-accident workloads and provided a case for why they are important to the study of archival systems. We describe a technique to group data such that fast, consecutive accesses to the same access group do not need an extra disk spin-up, and we showed that the potential power savings of this method is significant, especially in data sets that are automatically indexed. We are currently exploring algorithms for group identification and looking at the reliability implications of access grouped systems.

VII. Acknowledgments

Ian Adams provided much help in obtaining the data and pointing out access patterns. This research was supported in part by the National Science Foundation under awards CNS-0917396 (part of the American Recovery and Reinvestment Act of 2009 [Public Law 111-5]) and IIP-0934401, and by the Department of Energy's Petascale Data Storage Institute under award DE-FC02-06ER25768. We also thank the industrial sponsors of the Storage Systems Research Center and the Center for Research in Intelligent Storage for their generous support.

References

[1] A. Amer, D.D.E. Long, J.F. Paris, and R.C. Burns. File access prediction with adjustable accuracy. In *IPCCC 2002*, pages 131–140. IEEE Computer Society, 2002.

[2] M. Baker, K. Keeton, and S. Martin. Why traditional storage systems dont help us save stuff forever. In *HotDep 2005*, pages 2005–120, 2005.

[3] Enrique V. Carrera, Eduardo Pinheiro, and Ricardo Bianchini. Conserving disk energy in network servers. In *ICS '03*, pages 86–97, 2003.

[4] G. Cauwenberghs and T. Poggio. Incremental and decremental support vector machine learning. In *Advances in neural information processing systems 13: proceedings of the 2000 conference*, page 409. The MIT Press, 2001.

[5] D. Colarelli and D. Grunwald. Massive arrays of idle disks for storage archives. In *Proceedings of the 2002 ACM/IEEE conference on Supercomputing*, page 11. IEEE Computer Society Press, 2002.

[6] Western Digital. Western digital caviar green specification sheet. Western Digital, 2010. http://www.wdc.com/wdproducts/library/?id=132&type=8.

[7] X. Ding, S. Jiang, F. Chen, K. Davis, and X. Zhang. DiskSeen: exploiting disk layout and access history to enhance I/O prefetch. In *2007 USENIX ATC*, pages 1–14. USENIX Association, 2007.

[8] F. Douglis, P. Krishnan, and B. Bershad. Adaptive disk spin-down policies for mobile computers. *Computing Systems*, 8(4):381–413, 1995.

[9] F. Douglis, P. Krishnan, and B. Marsh. Thwarting the power-hungry disk. In *1994 USENIX ATC*, page 23. USENIX Association, 1994.

[10] R.O. Duda, P.E. Hart, and D.G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2 edition, 2000.

[11] D. Essary and A. Amer. Predictive data grouping: Defining the bounds of energy and latency reduction through predictive data grouping and replication. *Trans. Storage*, 4(1):1–23, 2008.

[12] G. Fung and O.L. Mangasarian. Incremental support vector machine classification. In *Proceedings of the Second SIAM International Conference on Data Mining, Arlington, Virginia*, pages 247–260, 2002.

[13] J.F. Gantz, C. Chute, A. Manfrediz, S. Minton, D. Reinsel, W. Schlichting, and A. Toncheva. The diverse and exploding digital universe: An updated forecast of worldwide information growth through 2011. *IDC white paper*, 2008.

[14] Timothy Gibson and Ethan Miller. An improved long-term file usage prediction algorithm. In *Proceedings of the 25th International Conference for the Resource Management and Performance and Performance Evaluation of Enterprise Computing Systems (CMG99)*, pages 639–648, Reno, NV, December 1999.

[15] D.P. Helmbold, D.D.E. Long, T.L. Sconyers, and B. Sherrod. Adaptive disk spin-down for mobile computers. *Mobile Networks and Applications*, 5(4):285–297, 2000.

[16] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, C. Wells, et al. Oceanstore: An architecture for global-scale persistent storage. *ACM SIGARCH Computer Architecture News*, 28(5):190–201, 2000.

[17] T.J. Mahoney. The long tail internet myth: Top 10 domains arent shrinking. Compete Inc., 2006. http://blog.compete.com/2006/12/19/long-%20tail-chris-anderson-top-10-domains/.

[18] D. McCullagh. Why no one cares about privacy anymore. 2010.

[19] Marshall Kirk McKusick, William N. Joy, Samuel J. Leffler, and Robert S. Fabry. A fast file system for UNIX. *ACM Transactions on Computer Systems*, 2(3):181–197, August 1984.

[20] D. Narayanan, A. Donnelly, and A. Rowstron. Write off-loading: Practical power management for enterprise storage. *ACM TOS '08*, 4(3):1–23, 2008.

[21] Jehan-François Pâris, Ahmed Amer, and Darrell D. E. Long. A stochastic approach to file access prediction. In *The International Workshop on Storage Network Architecture and Parall I/Os (SNAPI '03)*, sep 2003.

[22] E. Pinheiro and R. Bianchini. Energy conservation techniques for disk array-based servers. In *ICS '04*, pages 68–78. ACM, 2004.

[23] J.P. Rybczynski, D.D.E. Long, and A. Amer. Adapting predictions and workloads for power management. In *MASCOTS 2006*, pages 3–12, 2006.

[24] T. Schwarz, M. Baker, S. Bassi, B. Baumgart, W. Flagg, C. van Ingen, K. Joste, M. Manasse, and M. Shah. Disk failure investigations at the internet archive. In *MSST 2006*, 2006.

[25] M. Sivathanu, V. Prabhakaran, A.C. Arpaci-Dusseau, and R.H. Arpaci-Dusseau. Improving storage system availability with D-GRAID. *ACM TOS*, 1(2):133–170, 2005.

[26] C. Staelin and H. Garcia-Molina. Clustering active disk data to improve disk performance. *Princeton, NJ, USA, Tech. Rep. CS–TR–298–90*, 1990.

[27] M.W. Storer, K.M. Greenan, E.L. Miller, and K. Voruganti. Pergamum: Replacing tape with energy efficient, reliable, disk-based archival storage. In *6th USENIX FAST*, pages 1–16. USENIX Association, 2008.

[28] J. Urquhart. Does the fourth amendment cover 'the cloud'? 2008.

[29] J.J. Wylie, M. Bakkaloglu, V. Pandurangan, M.W. Bigrigg, S. Oguz, K. Tew, C. Williams, G.R. Ganger, and P.K. Khosla. *Selecting the right data distribution scheme for a survivable storage system*. School of Computer Science, Carnegie Mellon University, 2001.