# Analyzing Long-Term Access Locality to Find Ways to Improve Distributed Storage Systems

Alberto Miranda[1]
Toni Cortes[1,2]
Barcelona Supercomputing Center (BSC-CNS)[1]
Technical University of Catalonia (UPC)[2]
Jordi Girona 1-3 08034, Barcelona, Spain
{alberto.miranda, toni.cortes}@bsc.es

## Abstract

*An efficient design for a distributed filesystem originates from a deep understanding of common access patterns and user behavior which is obtained through a deep analysis of traces and snapshots. In this paper we analyze traces for eight distributed filesystems that represent a mix of workloads taken from educational, research and commercial environments. We focused on characterizing block access patterns, amount of block sharing and working set size over long periods of time, and we tried to find common behaviors for all workloads that can be generalized to other storage systems. We found that most environments shared large amounts of blocks over time, and that block sharing was significantly affected by repetitive human behavior. We also found that block lifetimes tended to be short, but there were significant amounts of blocks with long lifetimes that were accessed over many consecutive days. Lastly, we determined that most daily accesses were made to a reduced set of blocks. We strongly believe that these findings can be used to improve long-term caching policies as well as data placement algorithms, thus increasing the performance of distributed storage systems.*

## 1. Introduction

When designing a distributed filesystem, and any other computer system, the key to providing good performance relies on an effective optimization of common access patterns. This can be done only by obtaining a deep understanding of the behavior of the intended data workloads and users, which originates by analyzing filesystem traces or snapshots searching for data access patterns.

There are already several studies about data access locality [4, 7, 14, 17] though most of them focus on short-term access patterns and are normally performed from a process or file perspective. This is fine when considering data that is heavily used only for a few seconds, since identifying access peaks beforehand allows it to be cached in memory when needed.

Nevertheless, for data that accessed for long periods of time, it might be better to optimize physical allocation by clustering related data blocks or by arranging a multi-tier storage organization that allocated commonly accessed data to the fastest storage systems.

For this reason, we focused our study on long-term block access patterns, with a minimum granularity of 24 hours. We examined eight sets of traces collected at distributed storage systems, each one with different peculiarities and workloads. The traces used, though relatively old, correspond to well-known datasets that cover a wide range of workloads and have not been studied for long-term access locality. By studying the block temporal locality and sharing, we expect to find common access patterns that may lead to new data placement strategies or prediction algorithms, which might improve distributed storage systems. For instance, an accurate prediction of ”hot“ data blocks would allow to migrate them to faster storage nodes before they are needed.

The remainder of the paper is organized as follows: Section 2 provides background on previous trace studies, while Section 3 gives a description of the trace data used in the study. In Section 4 we explain the methodology used to perform our analysis, and we describe the results and findings we arrived at. Finally, Section 5 exposes conclusions extracted from this work.

## 2. Previous Work

Effectively characterizing filesystem behavior is difficult because of several problems. First, there is a wide range of

workloads to take on, each with its particular type of access patterns, data and intended users. Second, there are technical difficulties associated with the generation of traces, such as the performance impact incurred by enabling the tracing framework or the large amounts of data generated by it. Third, the usage semantics of filesystems change over time which can render current models invalid for future workloads. Nevertheless, over time several studies have been conducted that have provided a global view on common filesystem behaviors.

Early trace-based filesystem studies like those of Smith [18] and Ousterhout [14] provided useful observations on filesystem behavior that, though useful, have been slowly losing relevance due to changes in storage semantics. Smith studied text-based user files for thirteen months which are very different from the large multimedia files of current user workloads. Ousterhoust's traced three servers running BSD over a period of three to four days which is insufficient to predict long-term access trends.

Ramakrishnan *et al.* were the first to examine traces collected from commercial customer sites over several environments. They observed that only a relatively small percentage of all data was active at a time and that it received a considerable amount of accesses. They also found that a large portion of active data (23%-37%) was shared by multiple processes over time [16]. Nevertheless, they analyzed accesses to files with a resolution of hundreds of nanoseconds and their results may not apply to access patterns seen over several days or weeks.

In 1996, Gibson, Miller and Long [8] conducted a long-term study on filesystem activity on different UNIX environments. In particular, they were interested in the long-term behavior of files over a distributed filesystem to find common activity trends and access patterns. Consistently with previous works, they found out that 90% of all files were not used after they were created, and that approximately 1% of all files were used daily. Furthermore, they determined that files were usually short-lived and if they were not used immediately after being created, they'd never be. However, their study was file-based which limits its usefulness when considering block access behavior. Furthermore, after 15 years a new reevaluation of their results can be interesting.

In an attempt to track how filesystem behavior changed over time, Roselli *et al.* [17] measured a wide range of filesystems and compared their results with those from the Sprite study, conducted almost a decade earlier [4]. They noted that I/O load varied greatly depending on the environment study, but that file access patterns were bimodal in all environments (files were mostly read or mostly written). Most interestingly, they found that block lifetimes had increased since past studies, as well as maximum file sizes. They were primarily interested in determining how disk behavior was affected by caching, memory mapping and filesystem parameters. Therefore, even though their study was made at the block level, they didn't consider block sharing semantics.

Ellard *et al.* [7] analyzed NFS traffic for research and email environments and found out that blocks died quickly in both, and that many read access patterns classified as "random" by NFS servers were in fact long reads composed of sequential sub-runs. Though they focused on determining access patterns for individual files and blocks, they didn't analyze block or file sharing.

The most recent study that we are aware of, in 2008, examined CIFS traffic for two enterprise servers during three months [11]. They observed that read-write and random access patterns were more common than previously thought, that file sharing by clients was rarely concurrent and that a small fraction of clients accounted for most file activity. Again, this study used files as its basis and even though it analyzed file sharing by multiple clients, it only considered concurrent sharing, while we are interested in determining sharing over time.

## 3. Trace Data

In order to make the study as complete as possible, we chose a set of eight different traces representing a variety of computing environments. We studied traces collected at HP Labs (CELLO99), the University of Harvard at Cambridge, Massachussets (DEASNA and HOME02) and at a feature animation company (RENDER, VCS, DBS, NFSS and NFSC). Note that whenever we need to address this last set of traces as a group, we will refer to them as the ANIMATION dataset. When available, we include information about the storage architecture and the dataset size. Table 1 summarizes these traces which we describe in detail below:

- The CELLO99 traces are a set of I/O traces used in many I/O-related studies [10, 15, 22, 23]. Collected at HP Labs in 1999, CELLO99 capture I/O workloads from a typical research computer cluster. These traces are particularly interesting as they run for almost a year which makes them suitable for searching long-term locality patterns. Trace data is missing for two days and is incomplete for nine days. However, since they are such a small fraction of the totality of the data, we believe that this fact doesn't affect our analysis.

- The ANIMATION traces were collected from the NFS filesystem of a feature animation company in fall 2003-spring 2004. The traces were taken from several network locations, and include a pair of render racks (RENDER), a version control server (VCS), a commercial database server (DBS), various NFS servers and an

NFS cache (NFSS and NFSC). We considered each one as an individual environment to study.

Trace data seems to be incomplete for two days in the RENDER and nfs server datasets. Also, even though the traces were collected daily over several weeks, the RENDER dataset has a large gap of fifteen days just after the first three days of traces. After that, it continues normally.

- The DEASNA traces [7] were taken from the NFS system at Harvard's Department of Engineering and Applied Sciences over the course of 6 weeks, in mid-fall 2002. This system's workload is a mix of research and email.
- The HOME02 traces [7] were collected from one of the fourteen disk arrays in the Harvard CAMPUS system over 16 weeks. The CAMPUS NFS system served the majority of the school and administration at Harvard, with over 10,000 accounts. It consisted of three NFS servers, all connected to fourteen 53GB disk arrays. Traces were collected between August 2001 and December 2001. Trace data is missing for two days.

# 4. Trace Analysis

In this section, we describe our findings when analyzing block sharing for all workloads. We studied data and metadata accesses separately and, for each, we tried to determine relevant access patterns.

Concerning metadata, our results showed important amounts of block sharing for a large amount of blocks, which concurs with previous studies [1, 5, 11, 17]. Nevertheless, since this behavior was already described and we didn't make new findings, we will not go into further details in the paper.

For data blocks, we must distinguish between read and write operations: while the former showed important amounts of long-term sharing, the latter displayed very little access locality (most data blocks tended to be written and seldomly accessed) which didn't make them suitable for data placement optimizations.

For all these reasons, we decided to keep write operations out of the study and focus on analyzing usage patterns for read operations, since they showed more opportunities for improvement.

We analyzed sharing proportions for individual blocks and accesses, as well as the usage history for each block in the system. We were particularly interested in understanding how sharing and block usage changed over time, therefore we studied sharing from a distance point of view, and we analyzed the typical lifespan of shared blocks. We also studied relevant access patterns, access sequentiality and working set density.

## 4.1. Block Sharing

Histograms in Figure 1 show the distribution of shared blocks by the number of days of each dataset. We plot the normalized day count for each individual percentage of shared blocks. Besides, in order to give a better perspective on the distribution of sharing across days, we aggregate the results in granularities of 20% and also plot the number of days belonging to each sharing interval. For instance, for CELLO99 the graph shows that around 5% traced days shared *exactly* 40% of their blocks. It also shows that *in aggregate* around 45% days shared 40-60% of their blocks.

Figure 1 shows, unsurprisingly, that very different workloads have very different sharing profiles. Most days in traces CELLO99, DEASNA and HOME02 share about 20-80% of their blocks, with distributions mostly centered around 40-60%, 30-50% and 55-75% ranges respectively.

As expected, ANIMATION workloads vary significantly, with RENDER, VCS and DBS days rarely sharing over 60%, 35% and 30% blocks. The sharing distribution for NFSS and NFSC, however, is more similar to CELLO99, DEASNA and HOME02. This is interesting because all these environments are more interactive in nature, either because they are directly accessed by students/researchers (HOME02, CELLO99, DEASNA) or because they react directly to user requests (NFSS, NFSC). This suggests that interactive environments might have an increased sharing profile due to human repetitive access patterns.

Rendering processes, on the other hand, read a set of 3D models to write a scene to disk. Thus, each process reuses some file blocks (corresponding to models common to several scenes) and writes new file blocks that might or might not be used in following days. This might explain the reduced sharing percentage when compared to interactive environments.

Note that, predictably, for each dataset there is a large proportion of days that share very few blocks. However, there is also a noticeable amount of days (5-10%) sharing over 80% blocks.

**Observation 1** *In general, up to 60% days share more than 40% blocks.*

### 4.1.1 Sharing by Distance

While Figure 1 showed the overall distribution of block sharing for each workload, it says nothing about the temporal locality of this sharing. Figures 2 and 3 show the evolution of sharing profiles in function of the distance between days. For each point $(x, y)$ in the plot, $y$ is computed as the mean of the *percentage of shared blocks* between every pair of days whose distance is $x$. We also compute the standard error $\epsilon$ with 95% confidence in order to show the variability of those percentages. For instance, for CELLO99

| | Environment | Date of Traces | I/O Level | Trace Length | Read Accesses | Blocks Read | Avg. Reads/Block |
|---|---|---|---|---|---|---|---|
| CELLO99 | research | 1999 | block | 352 days | 734,239,483 | 377,484,947 | 1.945 |
| RENDER | rendering | 2003 | NFS | 25 days | 1,941,929,527 | 170,068,738 | 11.418 |
| VCS | versioning server | 2003 | NFS | 8 days | 25,799,219 | 9,202,048 | 2.804 |
| DBS | database | 2004 | NFS | 12 days | 172,851 | 146,174 | 1.183 |
| NFSS | file server | 2003 | NFS | 20 days | 1,159,197,936 | 144,597,245 | 8.017 |
| NFSC | caching server | 2004 | NFS | 7 days | 621,741,766 | 31,451,864 | 19.768 |
| DEASNA | research & email | 2002 | NFS | 38 days | 1,356,254,187 | 49,156,529 | 27.591 |
| HOME02 | home share | 2001 | NFS | 110 days | 2,755,799,037 | 197,645,388 | 13.943 |

**Table 1. Summary of traces examined**

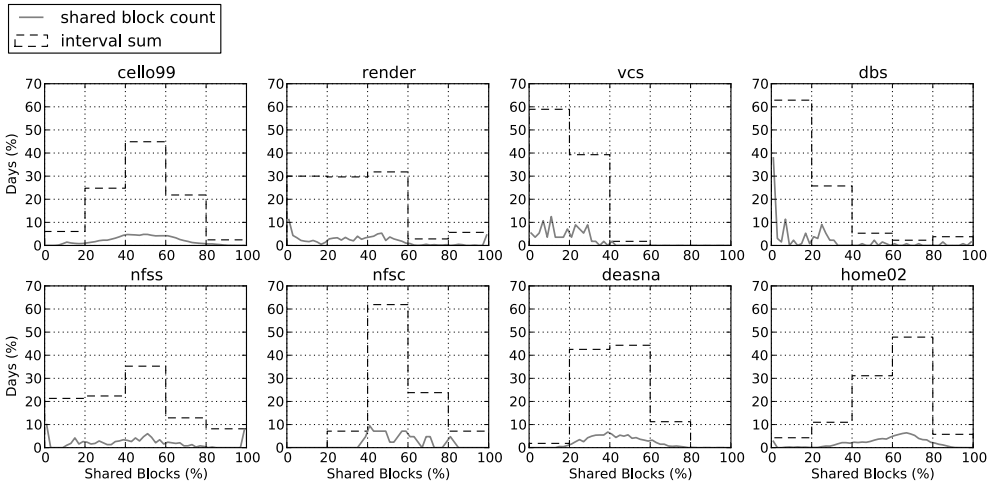


**Figure 1. Individual and aggregate percentage of shared blocks by day count**

the point $(100, 58 \pm 0.02)$ in the graph shows that all days at distance $+100$ of one another shared, on average, $58\%$ of their blocks with $0.02\%$ error. Notice that positive distances account for block sharing with future days while negative distances represent the amount of sharing with past days. Figure 2 includes long-term mostly general purpose workloads and Figure 3 includes mid-term specific workloads.

Figure 2 shows that sharing 40-60% of a day's blocks is a fairly common situation for all environments plotted, and that this sharing is done with days all over the year. It is fairly apparent, however, that there is much more sharing with future days than with past days, which might be explained by the creation of new blocks that can be shared with future days but not past days. Sharing over 60% of a day's blocks only happens with extremely close distances, around 15 days for CELLO99, 1-3 days for DEASNA and 25-50 days for HOME02. This makes sense, as it is very likely that blocks being used today were also used yesterday and will also be used tomorrow. Sharing beyond an 80% is very rare.

It is worth mentioning that there seems to be a periodical pattern where sharing is above normal behavior for all three workloads. This can be seen in all datasets in Figure 2 as a series of peaks that are almost evenly spaced, though it is more noticeable for CELLO99. Careful examination of the traces shows that this increase in sharing tends to repeat every 6 or 7 days, which strongly implies that it might be related to some cyclical situation associated to work weeks and repetitive human behavior.

Figure 3 shows, again, that workloads contained in the ANIMATION dataset are very diverse. NFSS, NFSC and RENDER workloads have a similar behavior to those of Figure 2 but with a lower amount of sharing: 30-50% for NFSS, 40-60% for NFSC and 30-40% for render. RENDER in particular seems to favor sharing with future days which supports our hypothesis that an important proportion of newly created blocks will be reused. VCS and DBS show the least amount of sharing (0-10% and 10-20%, respectively) that only grows for extremely close days. This is a typical behavior of source control systems, where writes (source commits) tend to dominate over reads (source checkouts), and would be expected of databases used mostly to keep information, rather than accessing it. Interestingly enough, the overall sharing for the DBS workload increases by the end

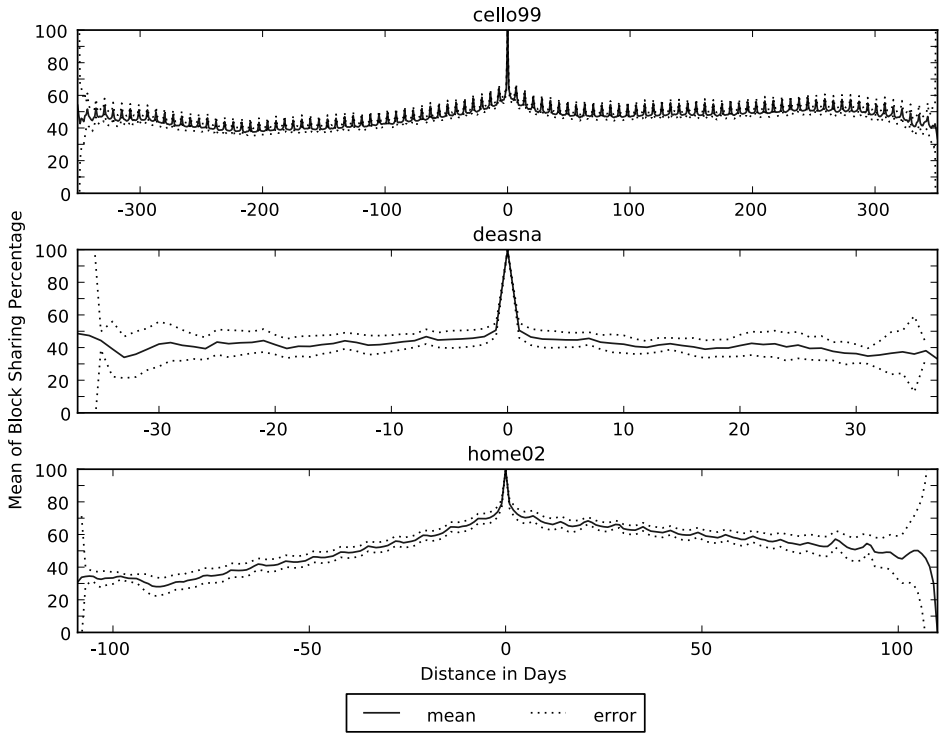


**Figure 2. Percentage of shared blocks by distance for general purpose workloads**

of the tracing period. Lack of further data prevents us from attempting to explain this behavior.

**Observation 2** *Block sharing, though decreasing as distance between blocks grows, remains stable at 30-50%.*

**Observation 3** *Block sharing is heavily influenced by repetitive human behavior.*

#### 4.1.2 Accesses to Shared Blocks

In the previous section, we saw that most workloads shared an important amount of blocks. However, it would be interesting to determine how often these blocks are accessed, since shared data is of no interest if it doesn't receive a significant amount of activity. Histograms in Figure 4 show the distribution of accesses to shared blocks normalized by the number of days of each dataset. Like in Figure 1, we plot the normalized day count for each percentage of accesses to shared blocks, as well as the aggregate.

Once again, we find the duality between general purpose, mostly interactive environments and environments with specialized workloads. Accesses to shared blocks are predominant for the former, with CELLO99 being in the 50-70% range, DEASNA in the 70-95% and HOME02 in the 60-80% range.

Workloads in the ANIMATION dataset are particularly interesting, because all but DBS have an important percentage of days with 40-80% accesses to shared blocks: 55%, 85%, 51% and 80% for RENDER, VCS, NFSS and NFSC, respectively. Note also that RENDER, NFSS and NFSC also have 10-30% days accessing over 80% shared blocks, implying that working sets for these environments might have been nearly identical for some days. This would make sense for processes continuously accessing the same sets of files over several days.

Note as well that there is also a large proportion of days accessing very few shared blocks for RENDER and NFSS, with DBS being the best representative for this tendency. This suggests that these environments accessed an important amount of blocks that had not been seen before. This would make sense for short-lived, temporary files like those created as part of a rendering process. This isn't conclusive, however, since there's no more data available for these environments.

**Observation 4** *In general, shared blocks are noticeably more accessed than non-shared blocks.*

### 4.2. Block Usage

In previous sections we have determined that there are considerable amounts of shared blocks in most of the work-

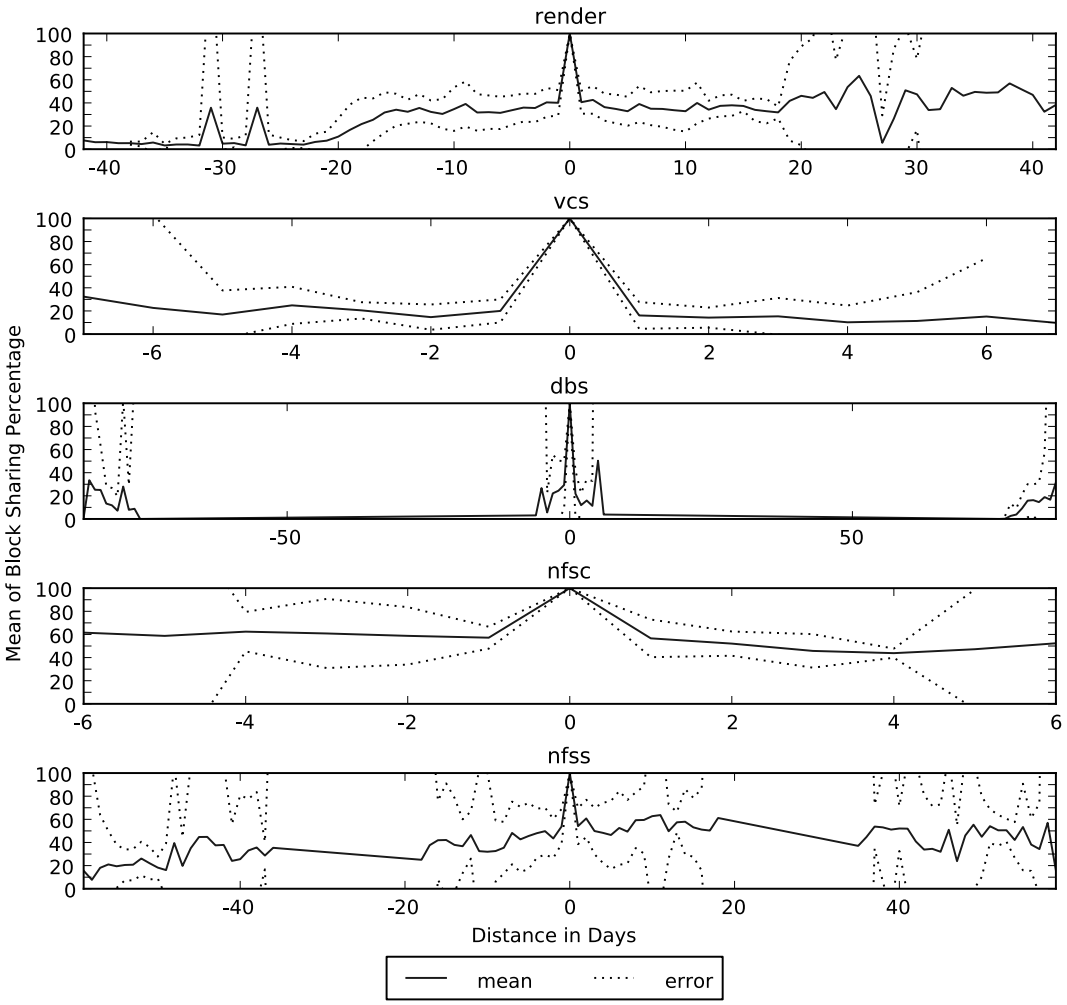


**Figure 3. Percentage of shared blocks by distance for specialized workloads**

loads studied and that these blocks represent a large portion of daily accesses. Nevertheless, it is still unclear if this set of shared blocks changes over time or the same blocks are being accessed over and over again.

#### 4.2.1 Block Lifespan

We define "lifespan" as the number of days between the first and last accesses recorded for each block. Figure 5 depicts the cumulative distribution of the blocks lifespan by number of blocks, grouped by lifespan length for the sake of legibility. Notice that due to the existence of gaps in some traces, lifespan length can be larger than trace length.

All traces show different but relevant amounts of short-lived blocks. This is to be expected as there should be many blocks that are accessed only once or for a few days (e.g. those related to short-lived files that are created and accessed over a week). For instance, blocks accessed in the DEASNA,DBS and VCS environments are extremely short-lived, with over 75% not being accessed again after 1 or 2 days. RENDER, NFSS, NFSC and HOME02 show that 40-50% blocks are not accessed for more than 4, 3, 1 and 16 days, respectively. The CELLO99 case stands out as short-lived blocks only represent a 13% of all blocks.

The relevance of medium to long-lived blocks (seen between 1-7 days) varies wildly for each environment. For the CELLO99, DBS, VCS, RENDER and DEASNA workloads they represent but a small fraction of all blocks (5-8%), whereas 15-20% of all NFSS and HOME02 blocks as well as 20-25% of NFSC blocks fall into this category.

Long-lived blocks (seen for more than a week) represent 10-12% of all blocks for DEASNA and DBS workloads. For RENDER, NFSS and HOME02 they add up to 40%, while 80% of CELLO99's blocks are long-lived. It is apparent

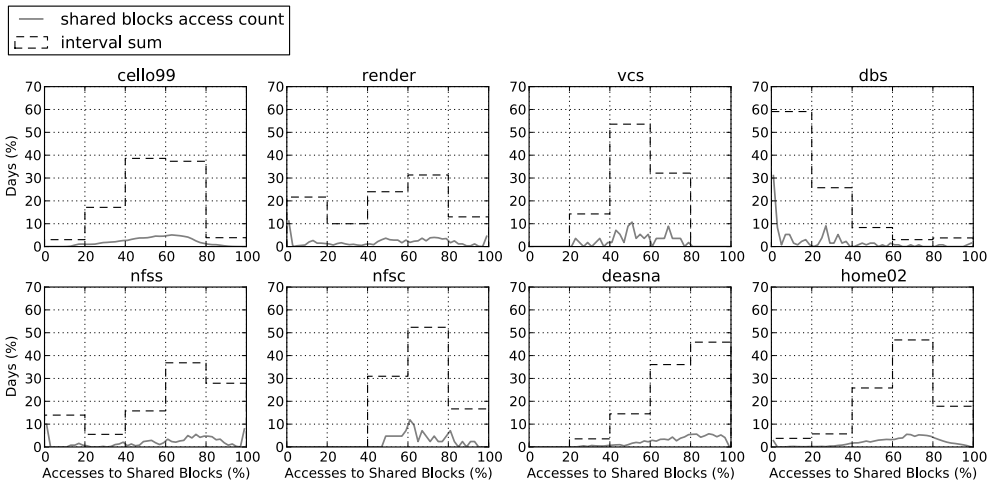


**Figure 4. Individual and aggregate percentage of accesses to shared blocks by day count**

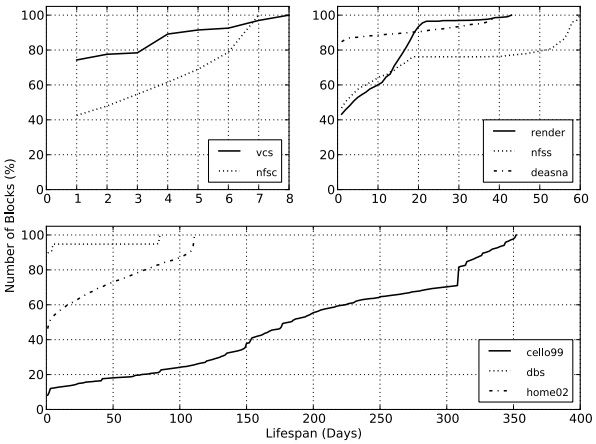


**Figure 5. Cumulative distribution of block lifespan by number of blocks**

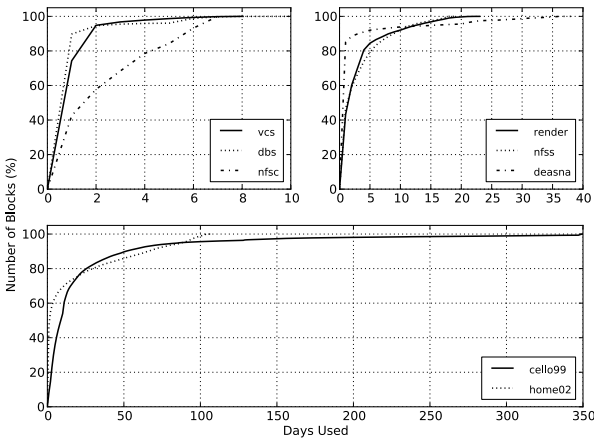


**Figure 6. Cumulative distribution of block usage by number of blocks**

that, the longer the trace, the longer the period of time where blocks were accessed repeatedly. It seems that blocks living over a day are very likely to live a relatively long time, which agrees with previous observations [7, 17]. We have not considered VCS and NFSC due to their short duration.

**Observation 5** *All workloads show important amounts of short-lived blocks (1-day lifespan).*

**Observation 6** *Lifespans greater than 1 day follow a uniform distribution.*

#### 4.2.2 Block Access Patterns

The previous section showed how long blocks were used, but it said nothing about how often they were accessed during this period of time. For instance, a block only accessed once on day 1 and once on day 200 would have a lifespan of 200 days, but this wouldn't reflect its real usage. Figure 6 shows the cumulative distribution for blocks actual usage in days, grouped by trace length this time.

We can see that although there is a lot of diversity concerning block lifespan, there are evident similarities when considering the actual amount of days each block is used. Around 80% blocks in research oriented workloads like

CELLO99 and DEASNA have been used for very few days (25 days for CELLO99 and 2-3 days for DEASNA), and the same happens with DBS. This means that those blocks were either accessed very frequently for a short amount of time or accessed intermittently over the trace duration. VCS and RENDER also show this behavior predominantly (around 5 days for 80% blocks).

In the case of NFSS and HOME02, a 25% of blocks were accessed for more than 55 and 75 days, respectively. Given that both environments are not computation-intensive, and the inherent randomness and unpredictability of their workloads, it seems likely that this represents large amounts of data accessed periodically.

The NFSC behavior is interesting, with up to 40% blocks being accessed for more than 4 days. Given the trace's length (7 days) and the caching behavior of this environment, this might suggest clients accessing the same blocks for several days.

**Observation 7** *Most blocks are accessed for short periods of time, even if they remain alive for a long time.*

#### 4.2.3 Long-Term Access Locality

The previous analysis were useful to determine the proportions of short, medium and long-lived blocks that characterize each environment, and also if blocks were used often during their lifetime or not. Now it would be interesting to find out whether blocks are accessed in successive day bursts or, on the contrary, they are accessed randomly. Figure 7 shows the cumulative distribution of the longest periods of time when blocks were used. We consider a period of time as a series of "consecutive" days when a particular block was used.

It is worth noting that this notion of consecutiveness isn't strict, as we have included a tolerance factor $tf$ in order to ignore small-sized gaps. Hence, a block used during $N$ consecutive days, that stops being used during $d <= tf$ days but is used again for another $M$ consecutive days, will be considered as used $N + d + M$ consecutive days. The reason for this is that, due to weekends and holidays, there might be intermittent gaps in the blocks usage. This way, we can filter out with high probability the usage gaps due to non-working days.

Figure 7 has been generated using tolerance factors of 0, 1, 2, and 3 in order to see the differences between them. Notice that lower tolerance factors seem to "contain" higher tolerance factors. This is to be expected since, for instance, a 3-day gap tolerance factor will ignore 3-day gaps as well as 2-day gaps and 1-day gaps.

When considering 0-gap tolerance, ANIMATION datasets show that about 90% of their blocks have been used consecutively for 5 days or less, though most curiously RENDER, NFSS and NFSC settle around the 5-day mark and VCS and DBS around the 1-day mark. For general purpose environments, 90% blocks for CELLO99 and DEASNA are used consecutively for 5 days or less. HOME02 doesn't follow this pattern, however, with 90% blocks being accessed up to 25 consecutive days. This might be explained by workload differences between research and user workloads.

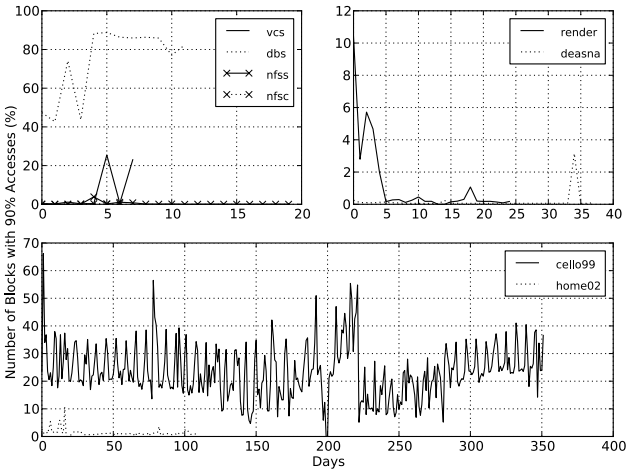


**Figure 8. Daily count of blocks with 90% accesses**

Increasing gap tolerance to 1 day only increases sequentiality counts significantly for NFSC (5-7 days for 90% blocks), whereas other workloads only exhibit slight increases.

For all workloads but DBS and NFSC, gap tolerances of 2-3 days increase sequential usage for 90% blocks by 2 and 2.5 times, respectively. Notice also that CELLO99, RENDER and HOME02 show around 5% blocks with very long runs (300-350, 18-25 and 96-120 days, respectively) that were not visible with smaller tolerance factors. Since the 2-day gap curves should be most "realistic" (as they include both weekends and 1-day holidays) this information could be very important when designing long-term caching algorithms.

**Observation 8** *Most blocks are accessed for a few consecutive days.*

**Observation 9** *Consecutive usage for blocks increases when adapting for human behavior.*

#### 4.2.4 Working Set Density

Finally, it would be interesting to determine if daily working sets are sparse and include a large number of different blocks, or if they are dense and a small number of blocks concentrate a large amount of accesses. Figure 8 plots, for each day, the number of blocks that concentrate up to 90%

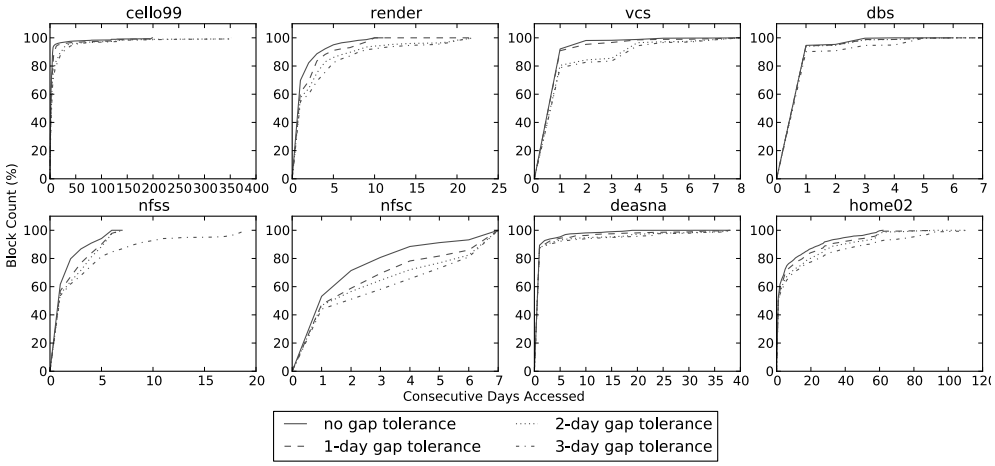


**Figure 7. Consecutive usage by block count**

daily accesses by daily count of blocks, grouped again by trace length.

Figure 8 shows that for all environments except CELLO99 and DBS 90% accesses concentrate on 1-6% blocks with low variance. All these systems are fairly stable, occasionally showing peaks or valleys where the block count increases or decreases significantly. This is important because a careful inspection of the data contained by the ANIMATION dataset shows several days where significantly less blocks were accessed. The number of blocks accessed these days was so low when compared to usual behavior for the rest of the trace, that we believe the difference might be due to tracing collection errors or anomalous situations. The case for the DBS is special since considerably more blocks were accessed at the beginning of the tracing period, thus lowering the results when compared to the rest of the trace. Nevertheless, the data provided is sensible enough to consider it as normal behavior.

General purpose environments like CELLO99, DEASNA and HOME02 show the variable behavior typical in interactive systems, though 90% accesses are usually made to the same block percentages: 20-40% for CELLO99, 0.05%-0.15% for DEASNA and 1-2% for HOME02. Notice also a peak around day 33 in DEASNA and a valley around day 200 in CELLO99. Both anomalies correspond to trace collection errors.

**Observation 10** *A small percentage of blocks receives up to 90% of daily accesses.*

## 5. Conclusions

In this paper we presented an analysis of eight different network filesystem traces, three of which were collected over long periods of time. Traces included two research workloads, an NFS home share workload, a rendering workload, three server workloads and a database workload. We analyzed daily block usage and read access patterns for each environment and found similarities between those with direct human influence, even though they were collected in different years.

We studied the variation of block sharing across filesystems and found that in general purpose workloads 50% of the blocks used in a day were also used some other day. Specialized workloads varied depending on the work performed, though we found a noticeably high amount of block sharing in those systems where human interaction was normal.

We also analyzed the correlation between block sharing and distance between days and found similarities in systems with direct and indirect human interaction. Unsurprisingly, block sharing was higher for closer days and tended to propagate to the future rather than the past, decreasing as the distance between days increased. Interestingly, for general purpose environments we found an important amount of sharing even for extremely distant days. Specialized workloads without human interaction, on the other hand, were likely to have less sharing. A most interesting result is that week periods had a strong influence in block sharing. A surprising but important result is that more than half daily accesses were made to shared blocks for all environments.

Block lifespan was very variable for different environments though similarly to previous results [7, 17] we found that blocks living over a day were very likely to live a relatively long time for the interactive long-term traces studied. Furthermore, we found that for all environments blocks were more likely to be accessed over a few consecutive days. Finally, we determined that 90% of all daily accesses went to relatively few blocks.

**Summarizing, we found that most environments have a high amount of blocks shared over time, that most accesses are made to shared blocks, and that the majority of those accesses is concentrated in very few blocks.** We believe these findings can be used to design long-term caching policies or prediction-based data placement strategies that determined future working sets with high probability: data could be placed in faster storage nodes or caches before it was actually needed, thus improving the response-time of the distributed filesystem.

## References


[1] N. Agrawal, W. Bolosky, J. Douceur, and J. Lorch. A five-year study of file-system metadata. *ACM Transactions on Storage (TOS)*, 3(3):9, 2007.

[2] E. Anderson, M. Arlitt, C. Morrey III, and A. Veitch. DataSeries: an efficient, flexible data format for structured serial data. *ACM SIGOPS Operating Systems Review*, 43(1):70–75, 2009.

[3] E. Anderson, S. Spence, R. Swaminathan, M. Kallahalla, and Q. Wang. Quickly finding near-optimal storage designs. *ACM Transactions on Computer Systems (TOCS)*, 23(4):337–374, 2005.

[4] M. Baker, J. Hartman, M. Kupfer, K. Shirriff, and J. Ousterhout. Measurements of a distributed file system. In *Proceedings of the thirteenth ACM symposium on Operating systems principles*, pages 198–212. ACM, 1991.

[5] J. Douceur and W. Bolosky. A large-scale study of file-system contents. In *Proceedings of the 1999 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 59–70. ACM, 1999.

[6] A. Downey. The structural cause of file size distributions. In *Modeling, Analysis and Simulation of Computer and Telecommunication Systems, 2001. Proceedings. Ninth International Symposium on*, pages 361–370. IEEE, 2002.

[7] D. Ellard, J. Ledlie, P. Malkani, and M. Seltzer. Passive NFS tracing of email and research workloads. In *Proceedings of the 2nd USENIX Conference on File and Storage Technologies*, pages 203–216. USENIX Association, 2003.

[8] T. Gibson and E. Miller. Long-term file activity and inter-reference patterns. *Computer Measurement Group (CMG 98) Proceedings*, 1998.

[9] S. Kavalanekar, B. Worthington, Q. Zhang, and V. Sharda. Characterization of storage workload traces from production Windows servers. In *Workload Characterization, 2008. IISWC 2008. IEEE International Symposium on*, pages 119–128. IEEE, 2008.

[10] S. Lee and H. Bahn. Data allocation in MEMS-based mobile storage devices. *Consumer Electronics, IEEE Transactions on*, 52(2):472–476, 2006.

[11] A. Leung, S. Pasupathy, G. Goodson, and E. Miller. Measurement and analysis of large-scale network file system workloads. In *USENIX 2008 Annual Technical Conference on Annual Technical Conference*, pages 213–226. USENIX Association, 2008.

[12] D. Muntz, P. Honeyman, and C. Antonelli. Evaluating delayed write in a multilevel caching file system. In *Distributed Platforms: Client/Server and Beyond: DCE, CORBA, ODP and Advanced Distributed Applications, Proceedings of the IFIP/IEEE International Conference on*, pages 415–429. IEEE, 2002.

[13] D. Narayanan, A. Donnelly, E. Thereska, S. Elnikety, and A. Rowstron. Everest: Scaling down peak loads through I/O off-loading. In *Proceedings of the 8th USENIX conference on Operating systems design and implementation*, pages 15–28. USENIX Association, 2008.

[14] J. Ousterhout, H. Da Costa, D. Harrison, J. Kunze, M. Kupfer, and J. Thompson. A trace-driven analysis of the UNIX 4.2 BSD file system. In *Proceedings of the tenth ACM symposium on Operating systems principles*, page 24. ACM, 1985.

[15] J. Park, H. Chun, H. Bahn, and K. Koh. G-MST: A dynamic group-based scheduling algorithm for MEMS-based mobile storage devices. *Consumer Electronics, IEEE Transactions on*, 55(2):570–575, 2009.

[16] K. Ramakrishnan, P. Biswas, and R. Karedla. Analysis of file I/O traces in commercial computing environments. In *Proceedings of the 1992 ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*, pages 78–90. ACM, 1992.

[17] D. Roselli, J. Lorch, and T. Anderson. A comparison of file system workloads. In *Proceedings of the annual conference on USENIX Annual Technical Conference*, page 4. USENIX Association, 2000.

[18] A. Smith. Analysis of Long Term File Reference Patterns for Application to File Migration Algorithms. *IEEE Transactions on Software Engineering*, 7(4):403–417, 1981.

[19] A. Traeger, E. Zadok, N. Joukov, and C. Wright. A nine year study of file system and storage benchmarking. *ACM Transactions on Storage (TOS)*, 4(2):1–56, 2008.

[20] W. Vogels. File system usage in Windows NT 4.0. In *Proceedings of the seventeenth ACM symposium on Operating systems principles*, pages 93–109. ACM, 1999.

[21] J. Wilkes. Traveling to Rome: QoS specifications for automated storage system management. *Quality of ServiceI-WQoS 2001*, pages 75–91, 2001.

[22] T. Wong, G. Ganger, J. Wilkes, and C.-M. U. P. P. S. O. C. SCIENCE. *My Cache Or Yours?: Making Storage More Exclusive*. School of Computer Science, Carnegie Mellon University, 2000.

[23] Q. Zhu, Z. Chen, L. Tan, Y. Zhou, K. Keeton, and J. Wilkes. Hibernator: helping disk arrays sleep through the winter. In *Proceedings of the twentieth ACM symposium on Operating systems principles*, pages 177–190. ACM, 2005.