File Migration on the Cray Y-MP at the National Center for Atmospheric Research*

Ethan L. Miller
Computer Science Division
Department of Electrical Engineering and Computer Science
University of California, Berkeley
Berkeley, CA 94720
elm@cs.berkeley.edu

June 25, 1991

Abstract

The supercomputer center at the National Center for Atmospheric Research (NCAR) migrates large numbers of files to and from its mass storage system (MSS) because there is insufficient space to store them on the Cray supercomputer's local disks. This paper presents file migration data collected over a 10 month period and some analysis of the data. The analysis shows that requests to the MSS are periodic, with one day and one week periods. Read requests to the MSS account for the majority of the periodicity; write requests are relatively constant over the course of a week. The intervals between requests to the MSS appear to be bunched, since well over half of the intervals are shorter than half the mean interval length. The latencies to access the first byte from disk, automated tape library, and manually-mounted tape are examined. In addition, information about file size distribution is given. We found that the NCAR system was different from systems studied earlier in several ways, including the ratio of local disk to tertiary storage. These differences affect the way that file migration can be done, so previous migration algorithms may not be effective.

1 Introduction

Over the last decade, computers have made incredible gains in speed. This speedup has encouraged the processing of larger and larger amounts of data; however, storing this data on magnetic disk is not feasible. Instead, most data centers with large data sets use tertiary storage devices such as tapes and optical disks to store much of their data. These devices provide a lower cost per megabyte

*This research was supported by contract S9128 with the University Corporation for Atmospheric Research (UCAR) which is sponsored by the National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this publication are those of the author and do not necessarily reflect the views of UCAR nor the National Science Foundation.

of storage, but they have longer access times than magnetic disk. By studying the tradeoffs between cheaper and slower tertiary storage and more expensive and faster disk storage, response time can be improved without increasing storage costs.

This paper will first cover the history and current solutions to the problem of mass storage. In the second section of the paper, we will present file migration data gathered on the Cray Y-MP used by the National Center for Atmospheric Research (NCAR). We will present the data and do some elementary analysis of it, particularly stressing its similarities to and differences from previous file migration studies. Finally, we will summarize current file migration policies and how they would perform on the system at NCAR.

2 Background

2.1 History

File migration systems are used by most large computer installations to store more data than that which would fit on magnetic disk. Tertiary storage, which usually consists of tape and optical disk, lies at the bottom of the "storage pyramid," which is shown in Figure 1. Cost and speed increase going up the pyramid, while the size of the memory level increases towards the bottom of the period. CPU cache, or perhaps even CPU registers, are at the top of the pyramid; they have the highest cost per byte and are the smallest and fastest of the levels. At the bottom of the pyramid are tape and optical disk, which have slow access speeds, on the order of seconds or minutes, and very low cost, under \$10/GB.

Early mass storage systems used manual tape mounting, since it was cheaper to hire system operators than it was to have a robot manage tape mounts. However, by 1978, several companies had introduced automated tape systems [Boy78], and automated tape storage became part of the mass storage systems in most major systems. Several systems were studied in the early 1980s; these included Brookhaven National Laboratory [EP82], the University of Illinois [LRB82], and the Stanford Linear Accelerator Center [Smi81b,Smi81a]. These will be discussed in a later section.

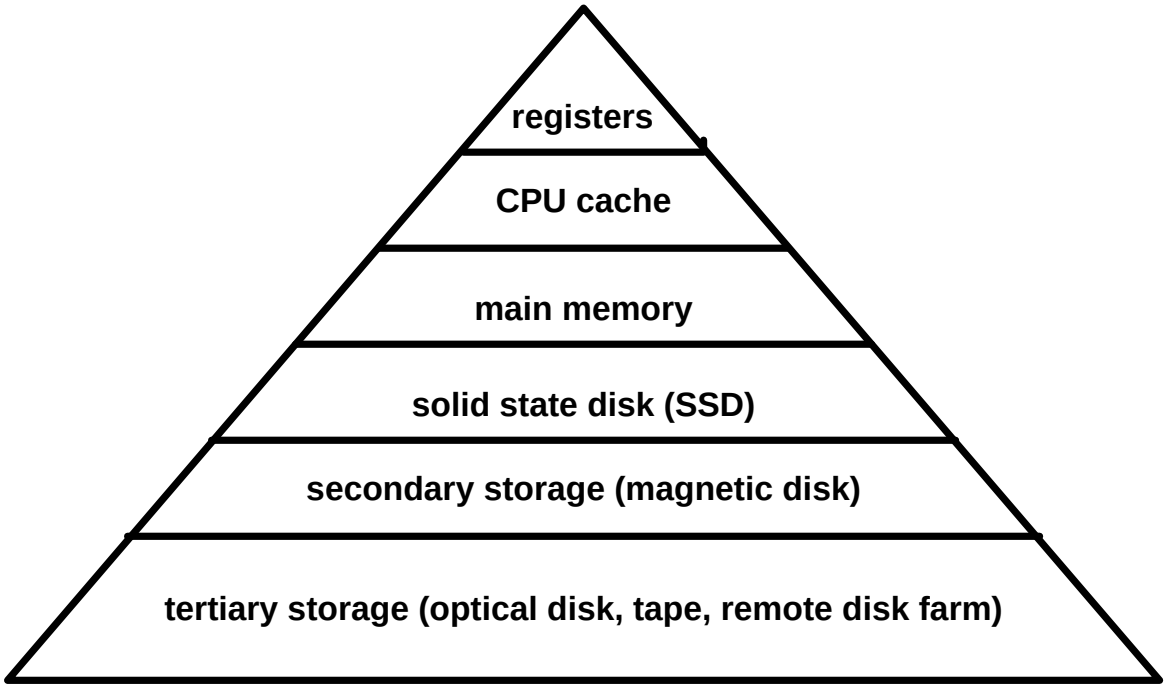

Figure 1: Memory hierarchy in large systems.

Since these studies, many complex mass storage systems have been implemented [McC87, NKMH87,HP89]. However, no studies on these systems have been published. Instead, the data management staff at these sites collect huge amounts of data to justify new equipment purchases and tune their systems. While this guarantees good performance for each system, it does not provide any guidelines for building future systems.

2.2 Mass Storage Devices

Currently, there are two major types of tertiary storage devices in common use—tape and optical disk. Both of these are high-density removable media. The tradeoffs between the two media are presented in Table 1. Two types of magnetic tape, helical scan and longitudinal scan, are presented. The numbers for the tapes come from [Woo88], while the optical disk statistics come from [Spe88]. The only change from the published numbers are for the IBM 3480, which is a longitudinal scan tape. This tape has now come out with a double density version, so the numbers for capacity per tape and cost per megabyte will be adjusted accordingly.

Category	Optical Disk Jukebox	Linear Tape	Helical-Scan tape
Media capacity (GB)	1.2	0.4	2.0
Random access speed	7 sec	13 sec	60 sec
Transfer rate (MB/sec)	0.25	3.0	0.25
Media cost/GB	$250	$35	$7

Table 1: A brief comparison of optical disk and tape.

The major tradeoffs among the three media are access latency and transfer bandwidth. Optical disks have a much lower access latency than either type of magnetic tape, but their bandwidth is also considerably lower. Thus, a system which performs many small I/Os to tertiary storage, such as a database system, would be best served by optical disk, since the dominating factor in calculating time per byte is access time to the first byte. For supercomputing installations, however, magnetic tape is better. While the time to get the first byte of data is longer for tape than for optical disk, the time to get *all* of the data is often lower for tape. Files on supercomputing installations tend to be large [Wal91], so the difference in transfer time between optical disk and tape is substantial. The two types of tape differ mainly in cost per MB and in transfer rate. Helical scan is a newer technology, and transfer rates are expected to rise. Another new technology, optical tape [Spe88], also looks promising because of its high density storage and high transfer rate.

Another primary consideration is price per gigabyte. As can be seen in Table 1, magnetic tape has a lower cost per gigabyte stored than optical disk. For systems with terabytes of data stored on tertiary storage, such as NCAR, this cost difference alone is enough to favor using tape exclusively as the tertiary store. The lower cost and higher transfer rate make magnetic tape the obvious choice for supercomputer centers which deal with sequentially-read large files.

Currently, the IBM 3480 tape format is standard at most supercomputer installations, though some are beginning to move to higher-density tapes such as Exabyte. The IBM 3480 uses linear recording, which provides high speed at the expense of recording density. The Exabyte drive, on the other hand, uses helical scan techniques (similar to conventional VCR recording) to greatly increase recording density; however, transfer rate is currently quite low on such tape drives.

Most installations today have one or more cartridge tape robots to automatically mount some of their tape libraries. An example of a tape robot, or automated cartridge system (ACS) is the StorageTek 4400 [LYSK87]. This system can provide access to 1.2 TB of data (6000 IBM 3480 cartridges, holding 200 MB each) without human intervention. Loading a cartridge takes approximately 6 seconds; from there, tape characteristics are identical to the IBM 3480.

2.3 Previous Work

There have been several studies of actual file migration systems, but they are quite old and deal with different computing environments. We will summarize them here, and in a later section will compare the results of studying the current NCAR environment with the results of the earlier studies.

In [Smi81b] and [Smi81a], Smith studied the file system at the Stanford Linear Accelerator Center. His data dealt with Wylbur text editor data sets, and tracked the references to those data sets. He found that the best algorithms had access to the entire reference string for a file. Since this is often not feasible, the migration criterion he suggested was to migrate off disk the files with the highest value of $last\ reference\ time^{1.4} \times file\ size$. This algorithm, called Space-Time Product (STP**1.4), was the best of the algorithms examined which did not make use of any file history other than the last reference time. The analysis in the paper also did not consider the possible effects of transfer time and access latency in minimizing average file reference time; instead, the analysis attempted to minimize file miss rate.

Smith also made several observations about file system activity. He noted that usage followed a weekly pattern, with activity highest on weekdays and lower on weekends and holidays. He also has extensive data on file sizes and interreference intervals; because of the size of the data set in the NCAR study (over 600,000 files), it would be very difficult to perform the same computations over the entire file set. The data set in the paper has a granularity of one day and does not distinguish between reads and writes.

For the data set in the paper, none of the acceptable migration algorithms would have had

much effect on average file access time. As noted in the paper, a miss ratio of 1% would mean a loss of 6.26 man/minutes per day, given the file usage rates and the number of users on the system. For STP, this miss ratio would require a disk system that held 1.5% of the total tertiary storage, and would require 300 tracks, or about 1 MB, of data to be transferred each day.

Lawrie, et. al., in [LRB82], considered the file migration patterns on the University of Illinois Cyber 175. Again, the system examined is quite different from the NCAR system studied in this paper. Interestingly, Lawrie reported that, though his system was quite different from SLAC, his results matched Smith's closely. This paper also examined several migration algorithms, and compared them against Smith's STP algorithm on their data. They found that STP was better than the algorithms they tried, which included pure LRU, pure length (migrate large files first), and SAAC, which migrated files that became less active. In all cases, STP outperformed these algorithms, though only by a slim margin.

Other papers have simply presented data gathered from existing mass storage systems without analyzing the data and suggesting possible algorithm changes. Systems analyzed include Brookhaven [EP82], NCAR [TH88,AN88], and NASA [HP89]. In addition, many large sites internally publish a summary of statistics gathered from their machines. They use these statistics for two purposes: to better tune their systems, and to justify new equipment purchases.

3 NCAR System Configuration

In this section, we describe the system on which the file migration traces were gathered. Rather than describe the entire NCAR network, we will focus on the parts which are relevant to the study. However, the rest of the network will be briefly described, since the mass storage system is shared by all of the systems at NCAR, so their presence might have an effect on mass storage systems performance.

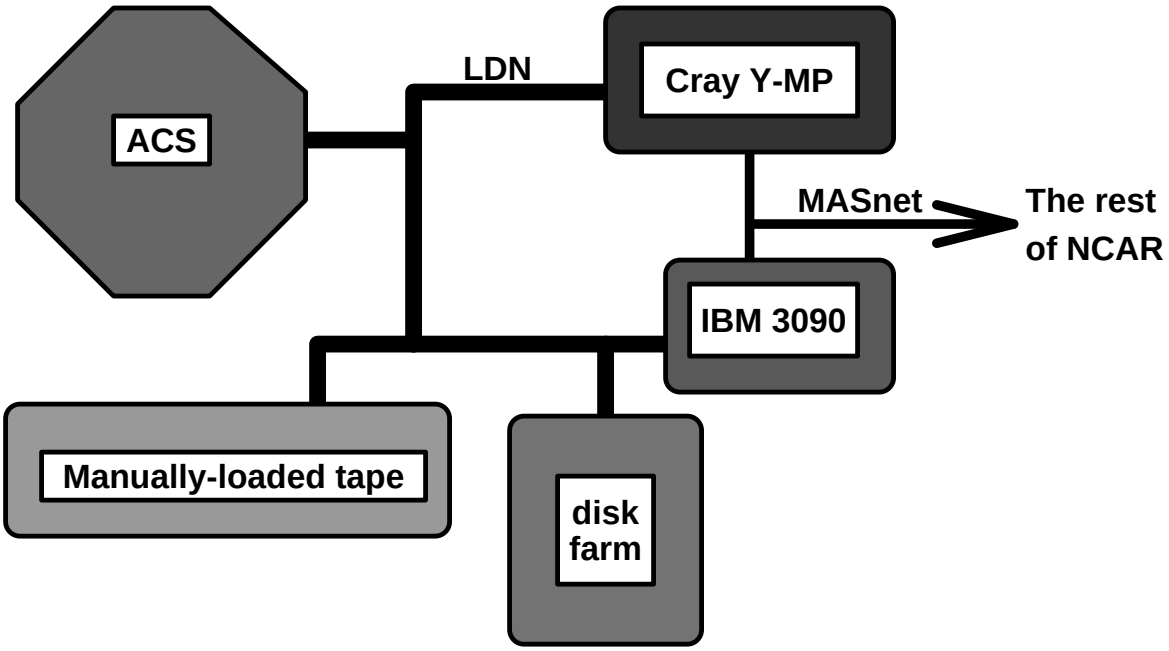

Figure 2: Network connections to the MSS at NCAR.

3.1 Hardware Configuration

The CPU in the study was a Cray Y-MP 8/864 (`shavano.ucar.edu`), with 8 CPUs and 64 MWords* of main memory. Each CPU has a 6 ns cycle time. `Shavano`, like other Cray Y-MPs, has several 100 MB/sec connections to its local disks and two 1 GB/sec connections to a solid state disk (SSD). There are about 56 GB of disks attached directly to the Cray; 47 GB of this space is reserved for application scratch space and files over a few days old are purged from it regularly.

The mass storage system (MSS) at NCAR is composed of an IBM 3090—used as a bitfile server—with 100 GB of online disk on IBM 3380s, a StorageTek 4400 Automated Cartridge System with 6000 200 MB IBM 3480-style cartridges, and approximately 15 TB of data in shelved tape. The MSS tries to keep all files under 30 MB on the 3090 disks, and immediately sends all files over 30 MB to tape. Usually, the tapes written are those in the cartridge silo. Files on the MSS are limited to 200 MB in length, since a file cannot span multiple tapes. While the Cray supports much larger files on its local disks, they must be broken up before they can be written to the MSS.

*Each Cray word is 8 bytes long.

The MSS at NCAR is shared by the entire NCAR computing environment, which includes the Cray Y-MP, an IBM 3090 which runs the MSS, several VAXen, and many Sun-type workstations. Figure 2 shows the network connections between the various machines at NCAR. The disks and tape drives attached to the MSS processor have direct connections to the Crays, providing a high-speed data path. All machines connected to the MSS (including the Crays) are connected to the 3090 by a hyperchannel-based network called the MASnet. Data going out over the MASnet must pass through the 3090's main memory, so it is a slower path than the direct connection the Crays have. All of the mainframes, and a few workstations also have connections to the MASnet, which is a custom high-speed network. The few workstations with connections act as gateways to the networks which connect the rest of the workstations at NCAR. These gateways are also usually the fileservers for the local networks. Many of these smaller machines have their own local lower-speed disks, about 5.5 GB of which are mounted by the Cray via NFS (Network File System). According to the monthly report published by the NCAR systems group [Wal91], `shavano` puts more data on the network than any other node, but several other nodes receive more data. In particular, several of the Sun workstations receive comparable amounts of data. It is likely that these workstations, which are the gateways to internal networks of desktop workstations, are receiving a large amount of image traffic.

3.2 System Software

The Cray Y-MP is primarily used for climate simulations— both the extensive number crunching necessary to generate the data, and the less computationally-intensive processing used in visualizing it. The Cray has two primary modes of operation; it can either run in primarily interactive mode, where programs are short and run as the user requests them, or in batch mode, where jobs are queued up and run when space and CPU time are available. There is no explicit switch between operating modes, but short interactive jobs typically have higher priority. During the day, scientists usually look at results from batch jobs submitted the previous day, so there is less CPU time for running batch jobs then. At night, however, the CPU is mainly used to run large jobs which require

hours of CPU time. The MSS request patterns will reflect these two different uses of the CPU, as will be shown in a later section.

The software which runs the MSS is based on concepts in the Mass Storage Systems Reference Model [CM90]. It consists of software on the mass storage control processor (MSCP), which is the IBM 3090, and one or more bitfile mover processes on the Cray. Users on the Cray make explicit requests (via the UNICOS commands `lread` and `lwrite`) to read or write the MSS. These commands send messages to the MSCP, which locates the file and arranges for any necessary media mounts. The MSCP then configures the devices to transfer directly to the Cray. For disk and tape silo requests, these mounts are handled without operator intervention, but an operator must intervene to mount any non-silo tapes which are requested. After the data is ready to be transferred, the MSCP sends a message to a bitfile mover, which manages the actual data movement. When transfer is complete, the bitfile mover returns a completion status to the user.

3.3 Applications

The Cray at NCAR runs two types of jobs—interactive jobs, which finish quickly and require a short turnaround time, and batch jobs, which may require hours of CPU time but have no specific response time requirements.

A typical climate simulation, such as the Community Climate Model [WKR+87], might take 1 hour and produce 500 MB of data which would be stored on a tertiary store. This is an example of a batch job, since a researcher would submit the job and allow it to run overnight or longer. These jobs use a large amount of temporary disk storage as well as CPU time. The Y-MP at NCAR is configured with small, 300 MB user partitions. Each user is allocated a few megabytes on one partition, which would be insufficient for storing the output of even one run of a climate model. Thus, the initial input to a climate model must come from the MSS, and any results must go back to the MSS. If the results are needed later, they must be retrieved from the MSS.

Interactive jobs, such as a "movie" of the results of a climate simulation, have much more stringent turnaround time requirements. Typically, a user will initiate a command and expect

a response quickly. According to [Twe90], an interactive request must be satisfied in just a few seconds, or interactive behavior is lost. Nevertheless, the average response time to satisfy MSS requests is over 60 seconds; possible solutions to this problem will be discussed later.

4 Tracing Methods

4.1 Trace Collection

The data used in this study was gathered from system logs generated by the mass storage controller process and the bitfile mover processes. Approximately 50 MB of data was written to these logs per month. The system managers at NCAR use the data to plan future equipment acquisitions and improve performance on the current system. The logs also serve as proof that a requested transaction took place. The system managers occasionally use them to refute users who claim their files were written to the MSS and then disappeared.

The system log, as written by the mass storage management processes, contains a wealth of information. Much of it is either redundant or unnecessary for migration tracing. Information such as project number and user name are not needed for migration studies, since the user identifier is also reported. The information is written to be easily human-readable, so fields are always identified and dates and times are in human-readable form. In addition, each MSS request is assigned a sequence number, since there are several records in the system log which correspond to the same I/O. This sequence number is useful for assembling a single record for a migration trace, but provides no additional utility. By processing the traces to remove redundant information and transform the rest of the information into a form more easily machine-readable, the traces were cut from 50 MB per month to 10-11 MB per month. The reason they were not reduced further was that file names are long, and they could not easily be compressed without losing information.

Field	Meaning
source	Device the data came from
destination	Device the data is going to
flags	Read/write, error information, compression information
start time	time in seconds since the previous start time
startup latency	time in seconds to start the transfer
transfer time	time in milliseconds to transfer the data
file size	file size in bytes
MSS file name	file name on the MSS
local file name	file name on the computer
user ID	user who made the request

Figure 3: Information in a single trace record.

4.2 Trace Format

Once the system logs were copied to a local host, they were processed into a trace in a format that is easy for a trace simulator or analysis program to read. The traces were kept in ASCII text so they would be easy to read on different machines with different byte orderings. A list of the fields in the trace is in Figure 3.

Very little information is common between two consecutive records except temporal information. Even so, the trace can be compressed by recording times as differences from some previous time [Sam88]. The start time for a MSS request is recorded as the elapsed time since the start time of the previous request. The latency until the first byte is transferred (the startup latency) and the transfer time are recorded as durations. Start time and startup latency are measured in seconds, while transfer time is measured in milliseconds. These were the precisions available from the original system logs. The only other commonality between consecutive requests might be the requesting user, so there is a bit in the flag field which indicates that the request was made by the same user who made the previous request. Directories, too, might be common between consecutive requests, but they would be harder to match. Future versions of the trace format may allow for full or partial paths to be obtained from previous records.

5 Observations

5.1 Trace Statistics

The traces for this study were collected over a period of 10 months, from June, 1990 through March, 1991. From the data collected, it appears that the MSS was very lightly used from June through August, perhaps because the Cray Y-MP was being brought up under UNICOS for the first time. The basic statistics from the trace are in Table 2. The traces actually contain 811,879 references; the references not counted in the table all contained errors. Approximately 4% of all the references traced had errors. By far the most common error was the non-existence of a file. In such cases, it was impossible to include the reference in our analysis, since the file never actually existed and thus couldn't be fetched or stored. It might have been possible to include references which encountered other errors, such as media errors and user termination errors (the user stopped the migration before it finished), but there were few enough that we believed it would not affect the results significantly.

One surprising statistic from the overall MSS statistics is that reads are more common than writes. Conventional supercomputer wisdom is that most files are written to disk and forgotten about. The data in Table 2 suggests otherwise. The data read/write ratio is 1.35 for the tape silo and 1.69 for disk. However, when manual tape I/O is considered, the read/write ratio for tapes jumps to 1.89. These numbers are relatively close together, and there is no obvious reason for the difference.

Clearly, the high read/write ratio for the manual tape is due to the migration policy. Usually, any data that needs to be written to tape will be written to a tape in the automated cartridge system (ACS). When the ACS runs out of free space, files which have not accessed recently are copied from the ACS to a manually archived tape, freeing up space in the ACS for more frequently-used files. In this way, tape writes can always be fast and never need wait for an operator.

The distribution of file sizes written to the MSS is shown in Figure 4. The distribution for files read and files written is similar, though there are fewer small files written and more large files

	Reads	Writes	Total
References	494394	284373	778767
Disk	342079	216550	558629
Tape (silo)	85176	61514	146690
Tape (manual)	67139	6309	73448
GB transferred	11163.7	6039.1	17202.9
Disk	1489.3	876.2	2365.6
Tape (silo)	6817.3	5040.3	11857.6
Tape (manual)	2857.0	122.6	2979.7
Avg. file size (MB)	22.58	21.24	22.09
Disk	4.35	4.05	4.23
Tape (silo)	80.04	81.94	80.83
Tape (manual)	42.55	19.44	40.57
Seconds to first byte	62.14	32.47	51.31
Disk	26.06	22.07	24.52
Tape (silo)	99.13	63.89	84.35
Tape (manual)	119.04	82.93	189.07

Table 2: Overall trace statistics.

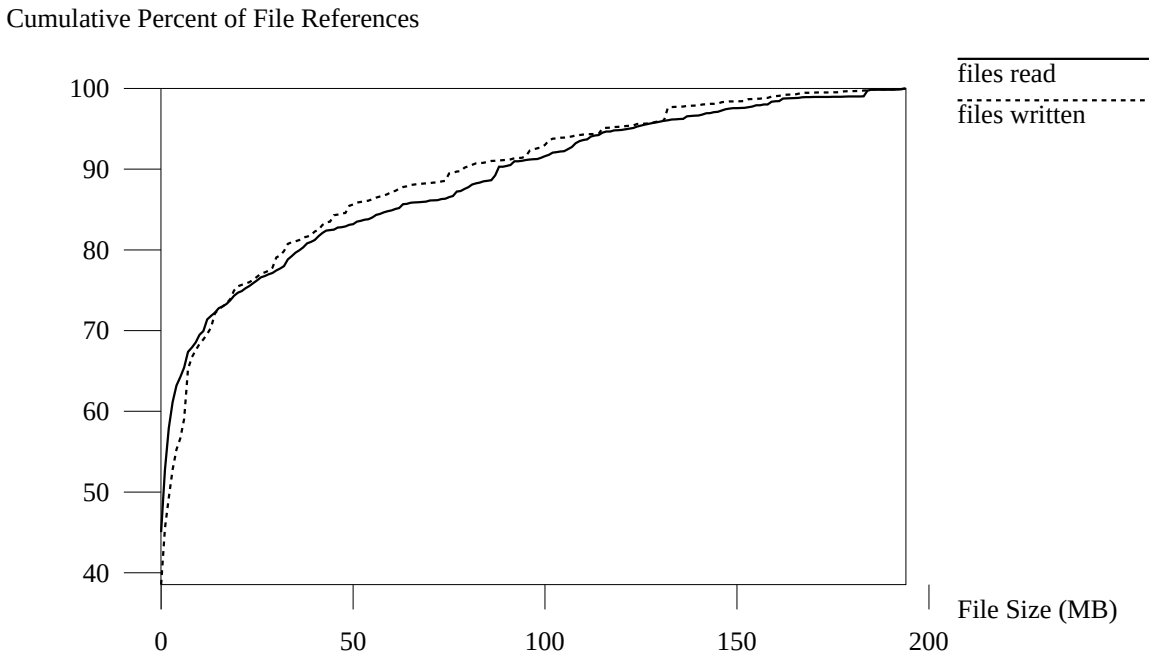

Figure 4: Cumulative file sizes weighted by reference count.

written. This seems to agree more with the concept of writing out files and never rereading them. The small files, which might include configuration files for a simulation, are written once and read more than once. Large files, such as simulation results, are written once and read less than once, on average. While there is a difference between the average size of a file read and the average size of a file written, it is not very large.

5.2 Daily Averages

Figures 5 and 6 show the average amount of data and the number of files transferred each hour of the day. On the graph, 0 is midnight and 23 is 11 PM, as in the European system. As would be expected, activity is highest from 9 AM to 5 PM, with a small drop at noon for lunch. However, this variation is due almost entirely to reads. The amount of data read jumps significantly at around 8 AM, when people arrive at work, and slowly tails off after 4 PM, as people leave. The fall is slower than the rise because most scientists are more likely to stay late than to arrive early. This suggests that most reads on the system are initiated by interactive requests, since there are more reads than writes during the hours when people are at work, and more writes than reads when the Cray is only running batch jobs.

The number of files written remain nearly constant over the course of the day, and the amount of data written varies even less. The low point in the data line, which occurs around 5 AM, might occur because that is when the Cray brought down for service. Because write requests are made mainly by batch jobs and not interactive jobs, they remain relatively constant over the day. Some writes are made by interactive jobs, but they tend to be smaller writes, as can be seen by comparing the lines for file write requests and data write requests.

5.3 Weekly Averages

The weekly data transfer averages, shown in Figure 7 and the weekly file transfer averages, shown in Figure 8 both show similar patterns to the daily averages. Day 0, hour 0 on the graph is Sunday at midnight, and the graph runs through Saturday night. The samples for the graph are two hours

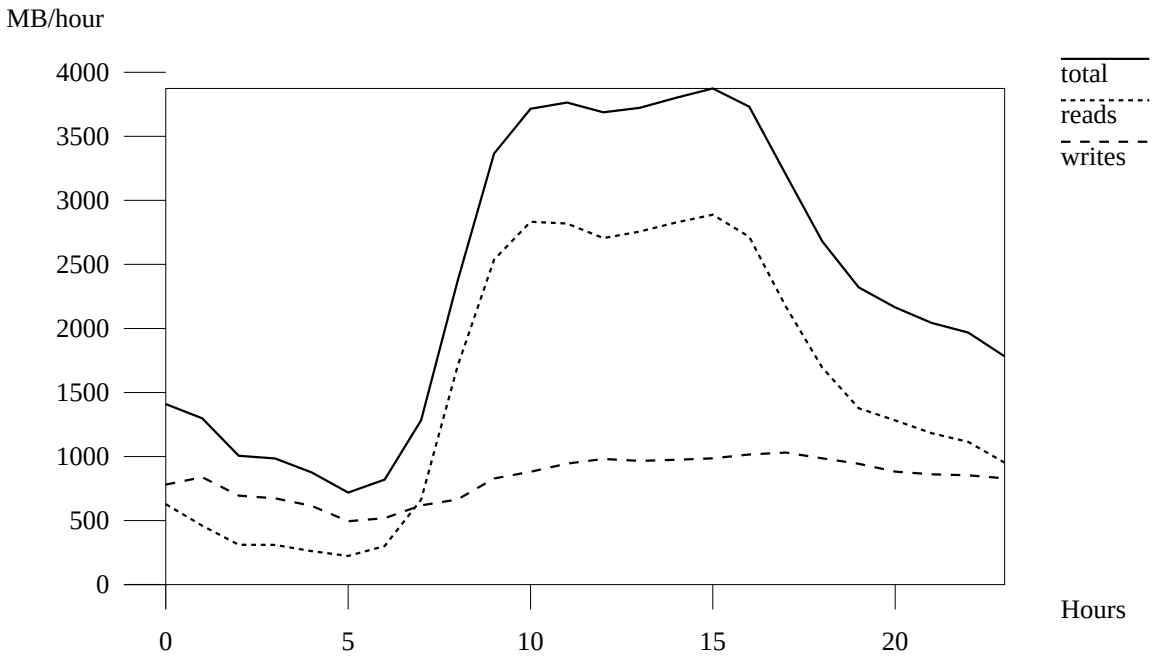

Figure 5: Average number of MB/hour transferred during each hour of the day.

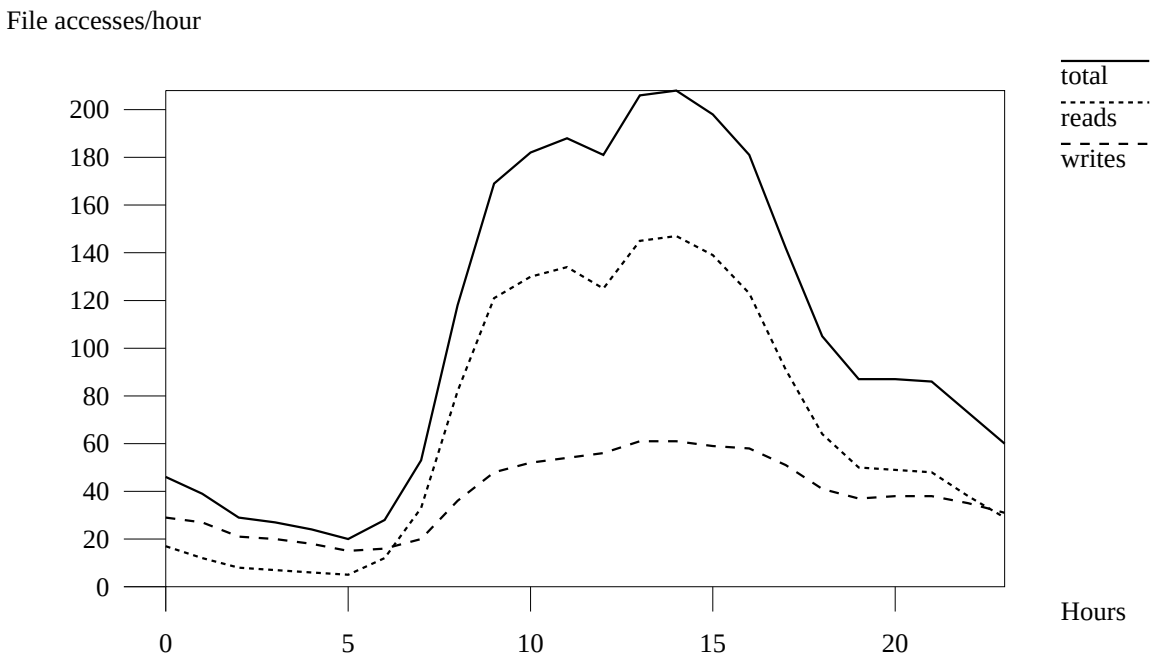

Figure 6: Average number of files/hour referenced during each hour of the day.

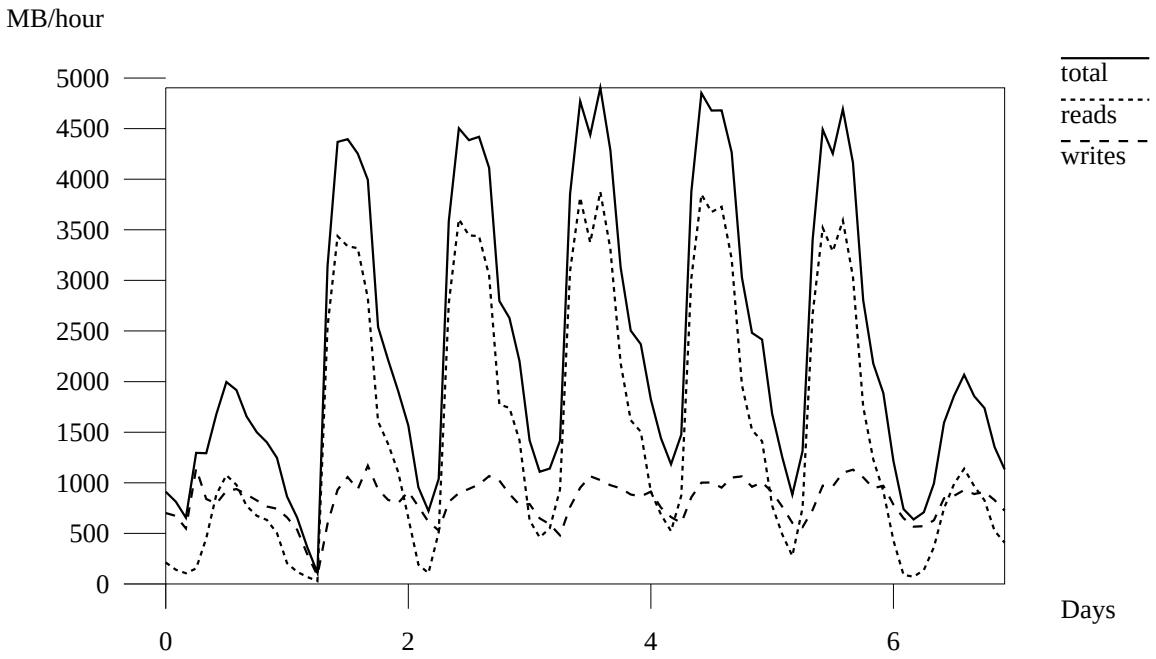

Figure 7: Average number of MB/hour transferred over a week.

apart.

As expected, read activity is lower on the weekends, since there are fewer scientists around to request data to pore over. On the weekdays, the MSS request distribution looks similar to that from the daily graph. However, the drop at lunch only appears on two days, Wednesday and Friday. This drop only appears in the data transfer rate, and not as much in the file transfer rate. We do not know why only those two days have such a steep drop while the other days have little or no drop during that hour.

Writes are much steadier than reads over the course of a week. However, there is a sharp drop in writes early Monday morning. This could be due to two things. First, the system might be brought down for maintenance then. Second, the job queue from the weekend might run out by then, so the machine would be mostly idle and requesting neither reads nor writes.

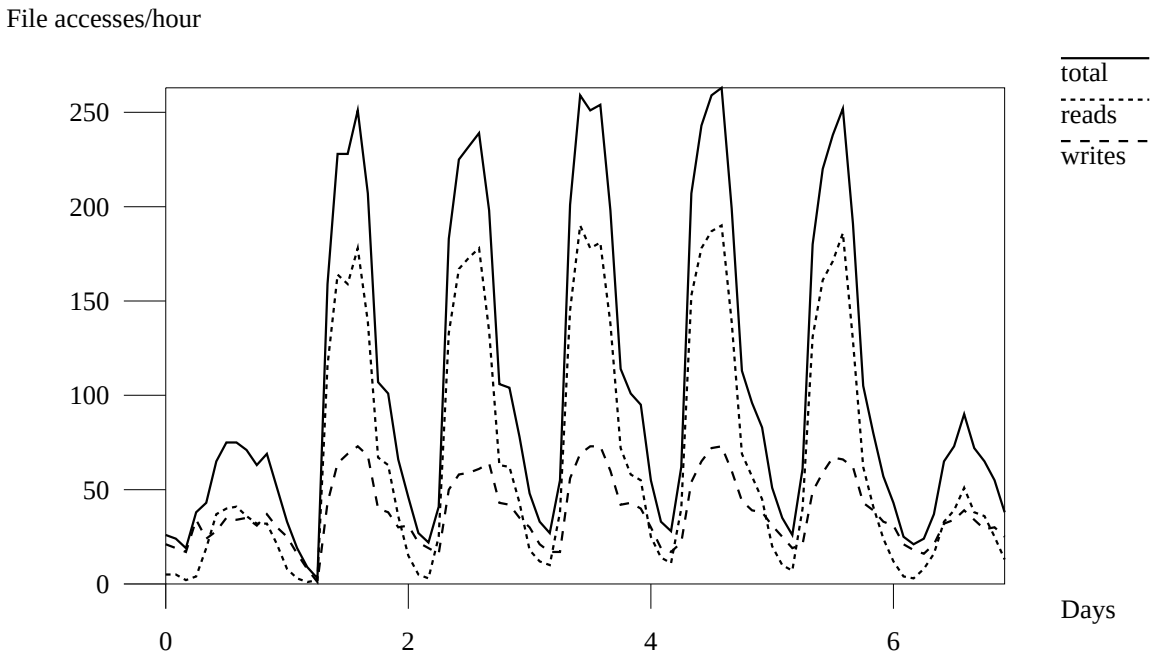

Figure 8: Average number of files/hour referenced over a week.

5.4 Long-Term Averages

The graphs of long-term usage, in Figures 9 and 10, both confirm the observations made by Smith [Smi81a]. The MSS request rate is periodic with a period of one week. There is a low week around Thanksgiving, at approximately day 25, and there is a period of little activity just past day 50 around Christmas.

The MSS request rate, in MB/hour, seems to be increasing over the period shown by the graph. This is likely due to more people using the Cray for computation rather than to any change in which data is stored on the MSS. In particular, as scientific visualization becomes more computationally expensive, more and more computation for it will be done on the Cray instead of on workstations. Since visualization requires reading large amounts of data from the MSS, increasing use of it will require reading more data.

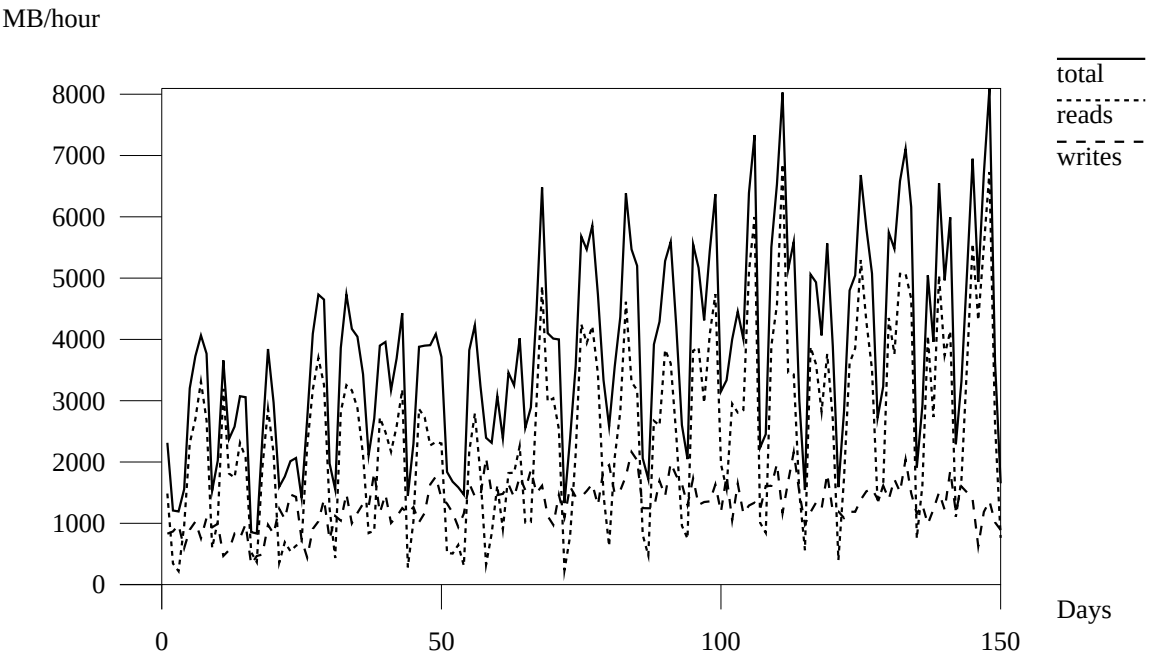

Figure 9: Average number of MB/hour transferred each day starting Nov. 1, 1990.

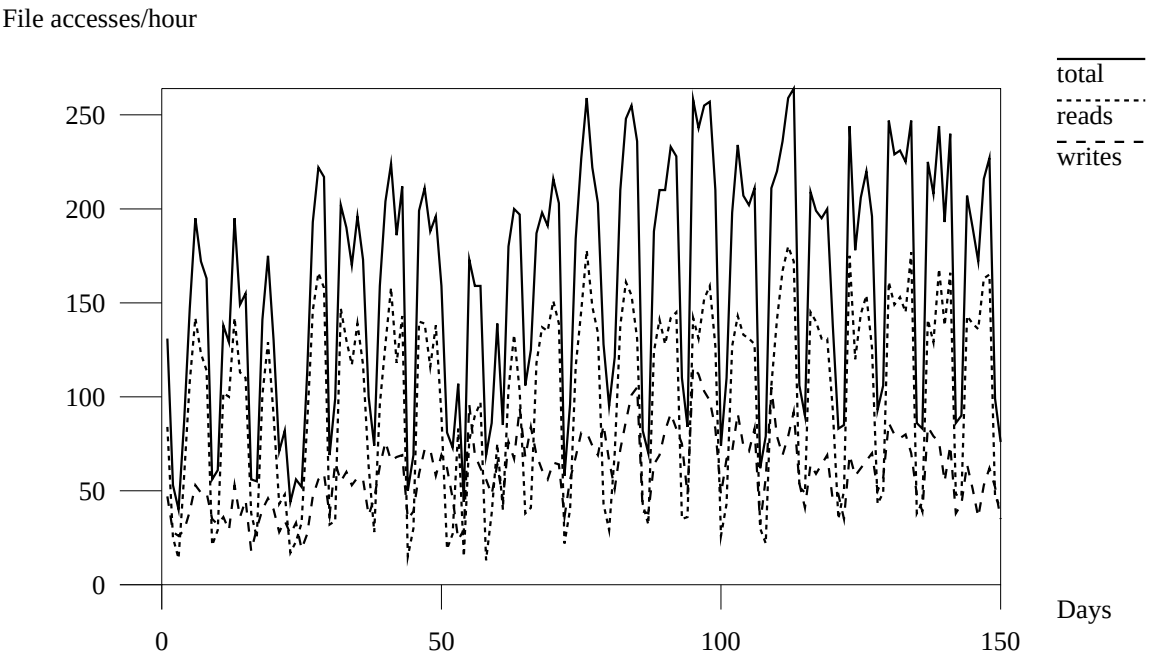

Figure 10: Average number of files/hour referenced each day starting Nov. 1, 1990.

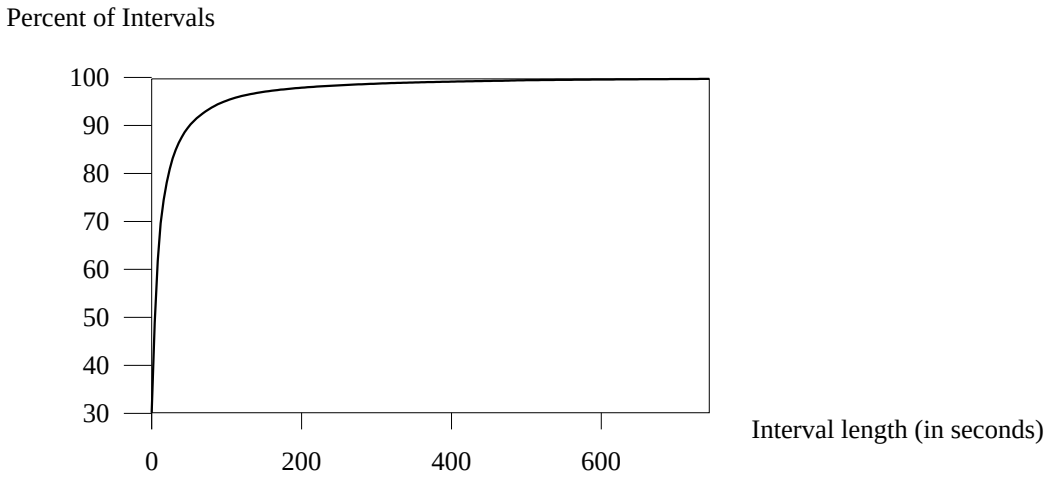

Figure 11: Length of intervals between MSS accesses.

5.5 Interreference Intervals

Figure 11 shows the distribution of intervals between references to the MSS. Since about 780,000 files were referenced over a period of 300 days (approximately 2.6×10^7 seconds), the average interval between MSS requests was 33 seconds.

Looking at the graph, however, shows that the vast majority of references followed another by less than 33 seconds; 69% of all references followed another by less than 16 seconds. This distribution suggests that I/Os are bunched together. There are several possible explanations for this bunching. First, bunching could occur since several files are accessed together by the same program. Since Cray files can be of (nearly) unlimited length, but files on the MSS cannot exceed 200 MB, this is a possibility. Another possibility is that there are really two distributions for intervals—those made by researchers' interactive requests, and those made by batch jobs. The interactive requests are very likely to be bunched together, since a researcher interested in day 1 of a climate model simulation will usually be interested in day 2, and both days will probably be in separate files.

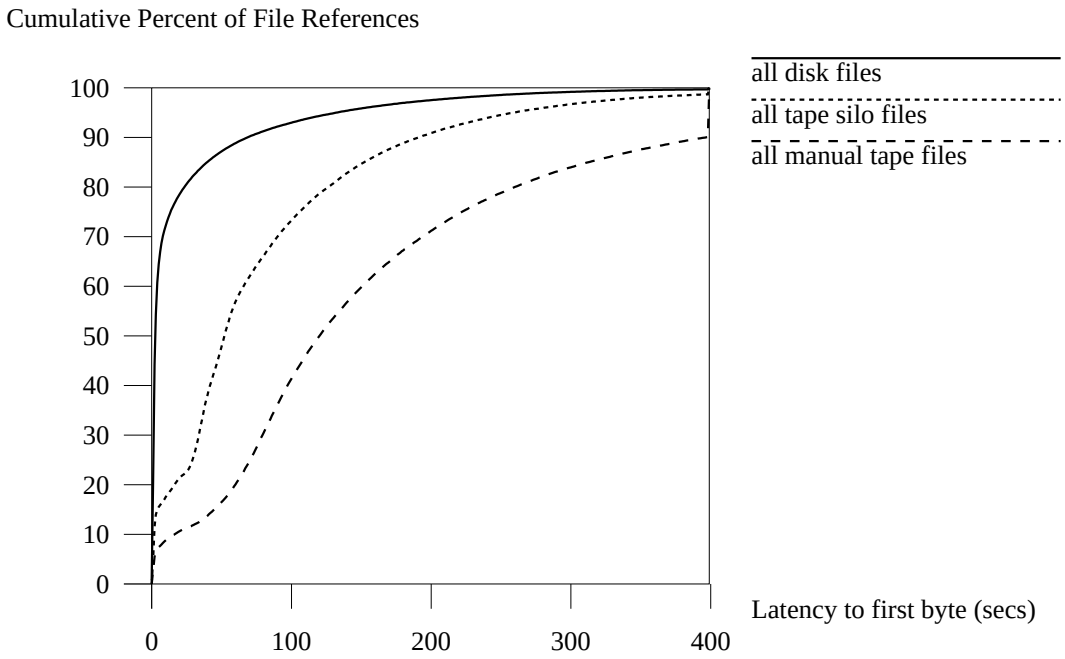

Figure 12: Latency to first byte for various devices.

5.6 Latency to First Byte

Figure 12 shows the total latency from when a request is made to the MSS until the data transfer actually starts. This time is composed of several elements—queueing time on the Cray, queueing time on the MSS, media mounting time, and seek time. For the disk, media mounting time and seek time are very short, usually well under a second. The disk times, therefore, are probably representative of the time spent in queues on the Cray and on the MSS. We can then deduce how much extra time is needed by the tape systems to get the first byte of data.

The first observation is that the tape silo is considerably faster than manually fetching the tape. After subtracting off the queueing time exhibited by the disk, the silo is approximately 2 to 2.5 times as fast as the manual tape drives at getting to the first byte. Since the tape silo tape drives are the same as the operator-loaded tape drives, this difference must come from the time to mount the tape rather than from seek time. The StorageTek 4400 ACS can pick and mount

a tape in under 10 seconds; after subtracting off average queueing time for the disk, which is 25 seconds, the non-seek overhead for reading an automatically-loaded tape is 35 seconds. According to Table 2, tape accesses take 85 seconds on average, so the average seek is 50 seconds long. When the same analysis is applied to manually loaded tapes, the manual tape mounting time is found to be approximately 115 seconds, or about 2 minutes. This is quite good. However, as Figure 12 shows, 10% of all manual tape mounts were not completed 400 seconds after they were requested. Nearly all of the tape silo and disk requests were completed by this time. This is probably the biggest weakness of manual tape mounting—the very long tail of the mounting time distribution. While other data accesses will almost certainly complete in 5 minutes, manual tape mounts may take much longer. This is just a simple analysis, though. There are several factors that we did not consider which may affect our conclusions here. In particular, queueing time for the tape silo may be different from queueing time for the disks. There are only a few tape robots in the silo, and each is tied up for several seconds with a tape load. If several tape loads come in close together, some of them will have relatively long queueing times. This does not happen with disk, as each disk is tied up for relatively little time with each request.

Another useful observation is the relation between latency to access the first byte and time required for the entire transfer. Both the tapes and the disks can transfer at a peak rate of 3 MB/sec, but the observed rates are usually closer to 2 MB/sec. As a result, the transfer times are similar for the two media. For tape, an average file of 80 MB will take 40 seconds to transfer. This is comparable to the additional 60 second overhead from using tape instead of disk. One possible way to improve perceived response time in the system would be to return after the first byte is transferred, as in [HP89]. Under this scheme, a call to open a file returns when the first byte is returned from the MSS, while the operating system continues to load the file from the MSS and keep track of how far it has gotten. When future requests are made, the data is returned immediately unless it has not yet been read. This scheme works because applications often do not read data as fast as the MSS can deliver it. Instead of delaying the application, then, it allows the application and file retrieval from the MSS to overlap. This system would be difficult to use in the

current NCAR configuration, however, since the MSS is not seamlessly integrated with the local disk file system. The bitfile mover processes would have to have special communication protocols with the local file system to let it know how much of the file has been transferred. Nevertheless, it is a useful optimization and should be considered.

6 File Migration Algorithms

Many of the algorithms examined in [Smi81b] and [LRB82] would not be directly applicable to the NCAR system. The biggest reason for this is the extremely low "hit ratio" seen at NCAR. The tertiary store has 16 TB of data, while the secondary store can only hold 47 GB. This is a ratio of 0.3%, but a 1% miss ratio for STP**1.4 in [Smi81b] would require approximately 2.1 TB, or 45 times more disk than actually exists. *Nothing* remains on the local disk for any length of time, because a cleaning program sweeps through it whenever the disk fills up and cleans out any data more than a few days old, usually 35%-50%. Until supercomputer disks become much larger, it is unlikely that traditional file migration algorithms will work effectively.

Another consideration in designing file migration algorithms for supercomputers is the low usage rate of most of the files. Unlike workstation file systems, supercomputer file systems contain large volumes of data that were written once and will never be accessed again. Smaller file systems contain similar data, but not in such large amounts. This data is accumulated by running simulations and storing all of the results. Tape is inexpensive, but Cray time is expensive, so it is useful to store the results of all computations, even if the results seem not to be useful. Even if only 1% of them prove to be useful later, the policy pays off.

Transfer time may dominate file access time for large files, while latency to get the first byte dominates for smaller files. Thus, it seems that NCAR's solution of keeping small and large files on different types of media is a good one. In addition, this policy gives better storage utilization, since small files would only take up a fraction of a tape, but a disk may be packed full of small files with little added overhead. Small files could be likewise packed onto tapes, but the overhead

in finding a tape with an empty space and mounting it would be quite high; the system at NCAR seems to be better. There is still a question of how large a file can get before it is put on tape, though. This is a question that future research could answer.

Instead of trying to minimize miss ratios on the disk, it seems best to minimize miss ratios on the tape silo and MSS disks, as NCAR has done. These two tertiary stores together have nearly the 2 TB needed for migration algorithms to have good hit rates. When analyzing the standard algorithms for such stores, though, the cost of accessing a file is much higher. Such a system uses a tape with a seconds-long seek time instead of a disk with a milliseconds-long seek time, so the parameters are different.

Currently, NCAR uses a simple algorithm to migrate data from the active tertiary store (the ACS and disk farm) to manually-mounted archive. When either the ACS or disk farm is full, files are selected for movement to archive using two keys. The first is time since last reference in days. Within these sets, files are selected by size, with the largest files being archived first. Files are moved from archive to active tertiary store when they have been accessed twice in 5 days. Otherwise, archived files are read directly from manually-mounted tape to the network. One future research direction will be to propose new algorithms for migration in such a system and test them against both real trace data and synthetic workloads.

7 Conclusions and Future Work

This paper has presented an analysis of the file migration activity at the National Center for Atmospheric Research (NCAR), a major supercomputer site. The observations are similar to those done on earlier migration systems, but differ in several ways. The average file size and file reference rate have both grown greatly. Currently, the system transfers between 2 and 8 GB/hour, and references 100-250 files/hour. File read activity is still periodic, with a period of one week, but file write activity remains relatively constant over the entire week. File size distribution is similar to that in earlier studies, though the files are an order of magnitude larger and the distribution tails

off more slowly.

The latency to the first byte of data for the disk, tape silo, and manually-loaded tape was broken down into various components. While tapes have longer seek times than disks, an automatic tape loading system is not as much slower than magnetic disk once transfer time is taken into account.

The time between references to the MSS was also analyzed, showing that these times are somewhat bunched. The reason for this bunching was unknown, but it is likely to occur because individual applications and researchers examining the files reference them in groups.

This paper presents an analysis of the data gathered at NCAR. There is still much work to be done devising file migration algorithms for data sets like those at NCAR. The algorithms presented in earlier file migration papers will be tested, but it is likely that new algorithms will have to be invented to deal with the different data organization and extremely larger tertiary storage system. It is likely that as systems get faster and data storage gets cheaper, more sites will run need to manage tertiary storage file systems as efficiently as most secondary storage file systems are run today.

References

[AN88] Edward R. Arnold and Marc E. Nelson. Automatic Unix backup in a mass-storage environment. In *USENIX — Winter '88*, pages 131–136, Februrary 1988.

[Boy78] Donald L. Boyd. Implementing mass storage facilities in operating systems. *Computer*, pages 40–45, February 1978.

[CM90] Sam Coleman and Steve Miller. Mass storage system reference model: Version 4. IEEE Technical Committee on Mass Storage Systems and Technology, May 1990.

[EP82] Carrel W. Ewing and Arnold M. Peskin. The Masstor mass storage product at Brookhaven National Laboratory. *Computer*, pages 57–66, July 1982.

[HP89] Robert L. Henderson and Alan Poston. MSS II and RASH: A mainframe UNIX based mass storage system with a rapid access storage hierarchy file management system. In *USENIX — Winter '89*, 1989.

[LRB82] Duncan H. Lawrie, J. M. Randal, and Richard R. Barton. Experiments with automatic file migration. *Computer*, pages 45–55, July 1982.

[LYSK87] David D. Larson, James R. Young, Thomas J. Studebaker, and Cynthia L. Kraybill. StorageTek 4400 automated cartridge system. In *Digest of Papers*, pages 112–117. Eighth IEEE Symposium on Mass Storage Systems, November 1987.

[McC87] Fred W. McClain. Mass storage at the San Diego Supercomputer Center. In *Digest of Papers*, pages 81–86. Eighth IEEE Symposium on Mass Storage Systems, November 1987.

[NKMH87] Marc Nelson, David L. Kitts, John H. Merrill, and Gene Harano. The NCAR mass storage system. In *Digest of Papers*. Eighth IEEE Symposium on Mass Storage Systems, November 1987.

[Sam88] A. Dain Samples. Mache: No-loss trace compaction. Technical Report UCB/CSD 88/446, University of California at Berkeley, September 1988.

[Smi81a] Alan Jay Smith. Analysis of long term file reference patterns for application to file migration algorithms. *IEEE Transactions on Software Engineering*, 7(4):403–417, July 1981.

[Smi81b] Alan Jay Smith. Long term file migration: Development and evaluation of algorithms. *Communications of the ACM*, 24(8):521–532, August 1981.

[Spe88] Ken Spencer. Terabyte optical tape recorder. In *Digest of Papers*, pages 144–146. Ninth IEEE Symposium on Mass Storage Systems, November 1988.

[TH88] Erich Thanhardt and Gene Harano. File migration in the NCAR mass storage system. In *Digest of Papers*, pages 114–121. Ninth IEEE Symposium on Mass Storage Systems, November 1988.

[Twe90] David Tweten. Hiding mass storage under UNIX: NASA's MSS-II architecture. In *Digest of Papers*, pages 140–145. Tenth IEEE Symposium on Mass Storage Systems, May 1990.

[Wal91] Sandra J. Walker. Cray Computer, MSS, MASnet, MIGS and UNIX, Xerox 4050, 4381 Front-End, Internet Remote Job Entry, Text and Graphics System, March 1991. Technical report, National Center for Atmosperic Research, Scientific Computing Division, March 1991.

[WKR+87] David L. Williamson, Jeffrey T. Kiehl, V. Ramanathan, Robert E. Dickinson, and James J. Hack. Description of NCAR Community Climate Model (CCM1). Technical Report NCAR/TN-285+STR, National Center for Atmospheric Research, June 1987.

[Woo88] Tracy Wood. D-1 through DAT. In *Digest of Papers*, pages 130–138. Ninth IEEE Symposium on Mass Storage Systems, November 1988.