A Large-Scale Study of File-System Contents

John R. Douceur and William J. Bolosky
Microsoft Research
Redmond, WA 98052
{johndo, bolosky}@microsoft.com

ABSTRACT

We collect and analyze a snapshot of data from 10,568 file systems of 4801 Windows personal computers in a commercial environment. The file systems contain 140 million files totaling 10.5 TB of data. We develop analytical approximations for distributions of file size, file age, file functional lifetime, directory size, and directory depth, and we compare them to previously derived distributions. We find that file and directory sizes are fairly consistent across file systems, but file lifetimes vary widely and are significantly affected by the job function of the user. Larger files tend to be composed of blocks sized in powers of two, which noticeably affects their size distribution. File-name extensions are strongly correlated with file sizes, and extension popularity varies with user job function. On average, file systems are only half full.

Keywords

File-system contents, directory hierarchy, static data snapshot, workload characterization, analytical modeling.

1. INTRODUCTION

We collected data on the contents of over ten-thousand Windows file systems on the desktop computers at Microsoft Corporation. The number of computer systems in our study is greater than that of the largest previous study of file-system contents [30] by over two orders of magnitude. We developed analytical approximations for several parameter distributions of interest and compared these to previously derived analytical distributions [28]. We studied the variation of usage parameters across file systems and investigated the relationship between the usage of a file system and the job function of the user. We also analyzed a file attribute that has not been discussed in previous studies: the compositional granularity of file size.

We found classical distributions that reasonably approximate file sizes, file ages, file lifetimes, directory sizes, and directory depths. Our findings for file lifetimes and directory sizes mirror previous results [28, 30], but those for file sizes and directory depths are different. File sizes have grown since previous studies [2, 30], and the median file age has increased [31]. File and directory sizes are fairly consistent across file systems, but file lifetimes and file-name extensions vary with the job function of the user. We also found that file-name extension is a good predictor of file size but a poor predictor of file age or lifetime, that most large files are composed of records sized in powers of two, and that file systems are only half full on average.

File-system designers require usage data to test hypotheses [8, 10], to drive simulations [6, 15, 17, 29], to validate benchmarks [33], and to stimulate insights that inspire new features [22]. File-system access requirements have been quantified by a number of empirical studies of dynamic trace data [e.g. 1, 3, 7, 8, 10, 14, 23, 24, 26]. However, the details of applications' and users' storage requirements have received comparatively little attention.

File systems are not used directly by computer users; they are used through application programs and system utilities, which are in turn constrained by the operating system. Currently, 75% of all client computers [16] run an operating system in the Microsoft Windows family [5, 34]. However, we know of no published studies of file-system usage on Windows computers.

The next section reviews previous work in the collection of empirical data on file system contents. Section 3 describes the methodology by which we collected the file-system data, and it describes some of our general presentation and analysis techniques. Our results are presented in Sections 4 through 8: Section 4 discusses file sizes, Section 5 discusses directory hierarchies, Section 6 discusses file ages and lifetimes, Section 7 discusses the correlation of various file properties with file-name extensions, and Section 8 discusses overall characteristics of file systems. Section 9 concludes by summarizing our work and highlighting our main contributions.

2. PREVIOUS WORK

Although we are unaware of any prior studies of file-system usage on Windows systems, there have been several studies of file-system contents in other operating-system environments, concentrating particularly on central servers or time-sharing systems. These studies show an increase, over time, in file size and count of files per user.

In 1981, Satyanarayanan [28] captured data from the eight 200-MB disk drives of a Digital PDP-10 used by the Computer Science Department at Carnegie-Mellon University, containing approximately 36,000 files. He recorded each file's size, timestamps, and type according to file-name extension. He introduced the concept of a file's functional lifetime as the time between the most recent access and the most recent modification. He observed that the count of files decreases nearly monotonically with increasing file size or functional lifetime, which led him to find hyperexponential distributions that approximate these two distributions.

SIGMETRICS '99 5/99 Atlanta, Georgia, USA

In 1984, Mullender and Tanenbaum [22] captured file-size data from a Unix system at the Department of Mathematics and Computer Science at Vrije Universiteit. The distribution of file sizes closely matched that of Satyanarayanan's study, despite the difference in operating systems.

In 1991, Bennett, Bauer, and Kinchlea [2] captured a day-long series of quarter-hourly snapshots from three file servers of the Department of Computer Science at the University of Western Ontario, containing 304,847 files. The servers were used by 45 mostly diskless Sun-III workstations with approximately 200 active users. They recorded each file's size, timestamps, and type according to attribute flags. Relative to Satyanarayanan's study, mean text-file size had increased by 5%, mean executable-file size had decreased by 38%, and the mean number of files per user had increased by nearly an order of magnitude.

In 1994, Smith and Seltzer [32] captured a ten-month series of daily snapshots from 48 file systems on four file servers in the Harvard Division of Applied Sciences. Although they captured file size and age information, they did not report distributions for these values, since their primary purpose was to study file fragmentation.

Also in 1994, Sienknecht *et al*. [30] conducted the largest published study of static file-system data of which we are aware. They captured a snapshot of file and directory data from 267 file systems in 46 HP-UX computer systems at the Hewlett-Packard Company, providing a combined 54 GB of disk space that housed 2.3 million files on behalf of 1845 active users. The mean file size ranged from 10 kB to 40 kB across file systems, which is greater than the mean file size found by Bennett *et al*.

3. METHODOLOGY

We collected data on the file systems of commercial desktop machines through voluntary participation of the computer users. We partitioned the data set according to each user's job function, and we analyzed the data using histograms, collective cumulative distribution functions, and analytical approximations.

We developed a simple program that traverses the directory tree of each fixed-disk file system of a computer. It records the name, size, timestamps, and containing directory of each file and subdirectory. All recorded data is written into a uniquely named log file on a server, along with system configuration information and the logon name of the user account.

We distributed the scanning program via email to nearly all of the employees at the Microsoft Corporation main campus, requesting the recipients to execute the program on their personal computers. To reduce the peak load on the network and on the receiving server, we spread the requests over 13 business days between September 1 and September 23, 1998. Of the approximately 20,000 people we contacted, 4418 executed the program on 4801 computers. Thus, self-selection [13] is a potential source of bias in our measurements.

File System	Systems	Files	Megabytes
FAT	6301	48,956,763	4,356,267
FAT32	934	10,957,672	1,020,196
NTFS	3332	80,732,302	5,679,335
WCEFS	1	305	6
Total	10,568	140,647,042	11,055,804

Table 1: File System Types

Job Category	Systems	Files	Megabytes
administration	385	2,408,453	222,415
business	902	6,994,811	715,732
management	939	10,358,877	921,520
non-tech development	548	5,287,621	386,095
technical development	6353	96,935,138	7,406,619
technical support	862	11,449,217	925,513
(other job or no record)	579	7,212,925	477,909

Table 2: Job Categories for Partitioning File Systems

After removing incomplete logs, duplications, and an outlier that contained over 330,000 1-MB files, we had valid scans of 10,568 file systems, containing 140 million files, excluding sparse files and paging files for the virtual memory system.

Our measurements of file size include only the application data, ignoring the effects of internal fragmentation, file meta-data, and any other forms of overhead. By this measure, the total size of all scanned files was 10.5 TB. All computer systems were Intel x86 machines, except for one Digital Alpha machine that contained three file systems. The file systems were primarily 16- or 32-bit FAT [21] and NTFS [34], as indicated in Table 1.

Since a user's work habits can strongly influence file usage [35], we partitioned the file systems according to the job function of the user, grouped into six categories, according to Tables 2 and 3. A disproportionately large share of files and file systems belonged to technical developers, reflecting both the greater likelihood for a technical developer to participate in our study and also the larger number of file systems on a technical developer's machine.

In subsequent sections, we discuss means of analysis specific to the data in each section, but in the present section we describe three techniques used throughout the paper. First, many of our graphs have horizontal axes that span a large range of non-negative numbers. To represent these ranges compactly, we use a logarithmic scale for non-zero values, but we also include an abscissa for the zero value, even though zero does not strictly belong on a logarithmic scale. For those graphs that display histograms, each bin corresponds to values greater than or equal to its label but less than the label of the succeeding bin.

Second, we employ *collective cumulative distribution functions*, which were introduced by Sienknecht *et al*. in 1994 [30]. We borrow the basis of their technique but display the result in a different form. A collective CDF conveys a summary of multiple frequency distributions in a single graph. An example is Figure 4, which illustrates the distribution of file sizes among file systems. The collective CDF curves indicate the density of standard CDF curves for the individual file systems. For example, in 50% of all file systems, 65% of the files are smaller than 16 kB; and in 95% of all file systems, 35% of the files are smaller than 16 kB. In several collective CDF graphs, including Figure 3, one or both extrema lines are beyond the illustrated scale and thus not visible.

Job Category	FAT	FAT32	NTFS	WCEFS
administration	313	61	11	0
business	636	158	108	0
management	580	135	224	0
non-tech development	445	30	73	0
technical development	3432	418	2502	1
technical support	514	72	276	0
(other job or no record)	381	60	138	0

Table 3: File System Types vs. Job Category

Third, to develop analytical approximations of empirical data, we select a distribution that is compatible with both the general shape of the observed histogram and our intuition about the underlying phenomena. We fit the cumulative curve of the analytical distribution to the cumulative curve of the observed frequency, by quasi-Newton search [25] for parameters that minimize mean square differences. We present densities or masses rather than cumulative distributions, in order to emphasize discrepancies between the empirical data and analytical functions. We quantify the fit according to the *maximum displacement of the cumulative curves* (MDCC), which is the greatest absolute vertical separation between the cumulative measured frequency curve and the cumulative analytical frequency curve. At a 0.01 level of significance, all of our continuous approximations fail the Kolmogorov-Smirnov test [19], and all of our discrete approximations fail the chi-square test [11]. Therefore, we do not claim to have found governing distributions for our observations. We claim only to have developed graphical approximations, whose utility is that they provide a highly compact – if not wholly accurate – representation of the empirical data.

4. FILE SIZES

File sizes are consistent across file systems and independent of user job function. Their frequencies fit a log-normal distribution, in contrast to the hyperexponential distribution posited previously [28]. The mean file size ranges from 64 kB to 128 kB across the middle two quartiles of all file systems, which is approximately four times the mean file size found in a recent study [30] of Unix systems. File size values suggest that most bytes are found in files composed of records sized in powers of two.

Figure 1 shows a histogram of files by size. 1.7% of all files have a size of zero, and the median size is 4 kB. We can approximate this histogram with a mixture of a constant distribution for zero-size files and a log-normal distribution [11] ($\mu^{(2)}$ = 12.2, $\sigma^{(2)}$ = 3.43, binary log; or $\mu^{(e)}$ = 8.46, $\sigma^{(e)}$ = 2.38, natural log), illustrated by the line on the graph. The MDCC is 0.009. To achieve this small an MDCC with a previously posited [28] hyperexponential distribution requires four stages for a total of seven fitting parameters, rather than the two required by log-normal. This suggests that hyperexponential is not as natural a fit for file sizes as is log-normal.

Satyanarayanan observed the count of files to decrease nearly monotonically with increasing file size, whereas Figure 1 reveals a general increase for file sizes less than 4 kB, which appears to indicate a profound shift in the gross shape of file-size distribution since 1981. However, this difference is accountable to two factors: First, whereas Satyanarayanan used a linear scale for file size, we use a logarithmic scale. Weighting frequencies by bin width, we observe monotonically decreasing frequency for all file sizes greater than 128 bytes. Second, Satyanarayanan measured file size in increments of 512-byte blocks. Combining frequency values for all bins below 512 bytes into a single bin hides the increase in file frequency therein.

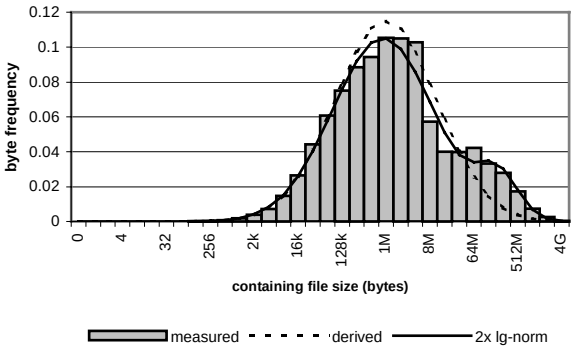

Figure 2: Histogram of Bytes by Containing File Size

Figure 2 shows a histogram of bytes by containing file size, alternately described as a histogram of files weighted by file size. The median size is 2 MB, which is 512 times greater than the median of the file frequency distribution, confirming the oft-repeated observation that most files are small but most bytes are in large files. The dotted line on the graph illustrates the fitted distribution from Figure 1 as weighted by file size. This curve is a poor fit to the observed data, indicating that the approximation in Figure 1 is not valid for file sizes greater than around 2 MB. A better approximation for the byte histogram is a mixture of two log-normals (α_0 = 0.91, $\mu_0^{(2)}$ = 20.3, $\sigma_0^{(2)}$ = 3.45, α_1 = 0.09, $\mu_1^{(2)}$ = 28.1, $\sigma_1^{(2)}$ = 1.35; or $\mu_0^{(e)}$ = 14.1, $\sigma_0^{(e)}$ = 2.39, $\mu_1^{(e)}$ = 19.5, $\sigma_1^{(e)}$ = 0.94), as illustrated by the solid line on the graph. The MDCC is 0.016, which could be improved by increasing the number of log-normals in the mixture, but doing so would run counter to our goal of developing very simple analytical models.

The poor fit of the derived curve in Figure 2 suggests that log-normal is not a good fit for the tail of the file size distribution. Recent evidence [4] suggests that hybrid distributions, with log-normal bodies and Pareto tails, approximate file sizes on the Web.

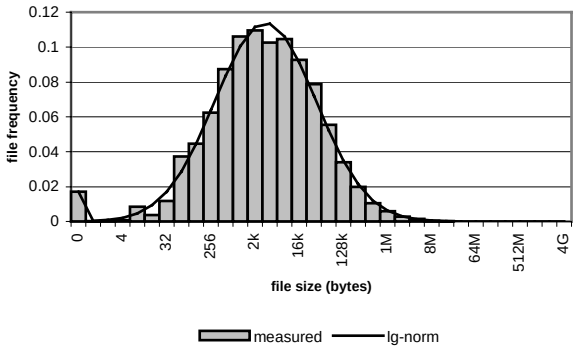

Figure 1: Histogram of Files by Size

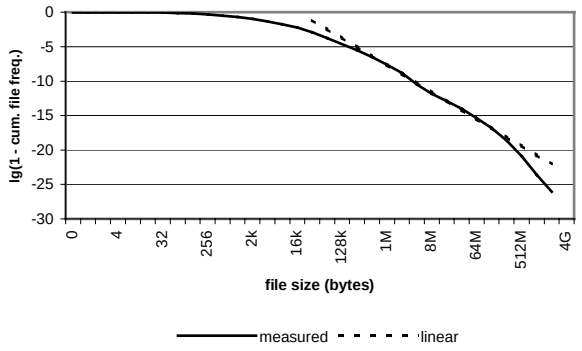

Figure 3: Lg-Lg Complementary Distribution of Files by Size

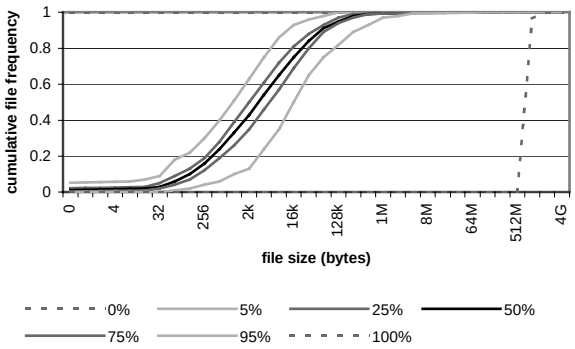

Figure 4: Collective CDF of Files by Size

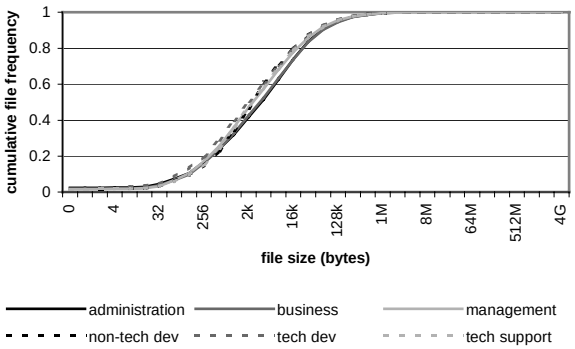

Figure 6: Job-Category CDFs of Files by Size

Figure 3 shows a *log-log complementary distribution* (LLCD) [9] of file sizes, which is the binary logarithm of the complementary cumulative distribution versus log-scaled file size. Approximately straight-line behavior is seen in the range of 256 kB – 256 MB, indicative of a Pareto-like heavy tail. Fewer than 1000 files have sizes beyond 256 MB, so the subsequent drop-off from linear may not be significant. The slope of the fitted straight line in Figure 3 is –1.3, suggesting that $\alpha = 1.3$ in the Pareto tail distribution.

Figures 4 and 5 show collective CDF plots, as described in Section 3, of the histograms illustrated in Figures 1 and 2, respectively. Figure 4 shows that the distribution of file sizes is relatively consistent across file systems: On 50% of file systems, the median file size ranges by a factor of 4 from 2 to 8 kB, and on 90% of file systems, it ranges by a factor of 32 from 1 to 32 kB. Figure 5 shows that the distribution of bytes varies more widely: On 50% of file systems, the median byte-weighted file size ranges by a factor of 8 from 256 kB to 2 MB, and on 90% of file systems, it ranges by a factor of 4096 from 32 kB to 128 MB.

Figures 6 and 7 show standard CDF plots, partitioned according to the job category of the file system's user, of the histograms illustrated in Figures 1 and 2, respectively. These figures show that there is virtually no difference in file-size distribution among different user job categories.

Figures 1 – 7 group ranges of sizes into bins and thereby ignore the effect on file frequency, if any, of the less significant bits of the file size value. If there were no such effect, then roughly half of all files would have even sizes, and half would have odd sizes; half of the even sizes would be divisible by four, and half would not; and so on. However, this is not the case; for example, 63% of all files have even sizes, and 63% of these are divisible by four. We posit that this is because files are often composed of records that are sized in powers of two.

We define the *compositional granularity* (CG) of a file as the size of the records that compose the file. A file's size will always be an integral multiple of its CG. The CG of a file is not necessarily the largest power of two that divides the file size, because the size could be an even multiple of the CG. For example, a file that is composed of six 4-byte records has a size of 24 bytes, which is divisible by 8. On average, half of the files with a given CG will have sizes that are divisible by twice the CG, and half of those will have sizes that are divisible by four times the CG, and so on.

To estimate CG frequency, we define the function $f(n)$ as the frequency of files whose size is divisible by 2^n but not by 2^{n+1}. There are $f(0)$ odd-sized files, all of which have a CG of one byte, and we expect a roughly equal number of even-sized files to have a CG of one byte, so we initially estimate the frequency of files with a one-byte CG as 2 $f(0)$, although we will modify that estimate shortly. We can make a similar estimation for other values of CG, but we must first disregard the files we just determined to have a one-byte CG, by defining a new function $g(n) = f(n) - 2^{-n} f(0)$. Since negative frequencies are meaningless, we actually use the function $f'(n) = \max(g(n), 0)$ for estimating the frequency of files with a two-byte CG, and we subtract any differences between $f'(n)$ and $g(n)$ from our estimate of one-byte CG frequency. Similarly, we use the function $f''(n) = \max(f'(n) - 2^{1-n} f'(1), 0)$ for estimating four-byte CG frequency, and so forth.

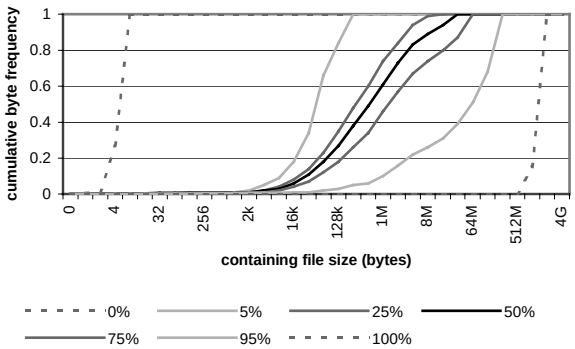

Figure 5: Collective CDF of Bytes by Containing File Size

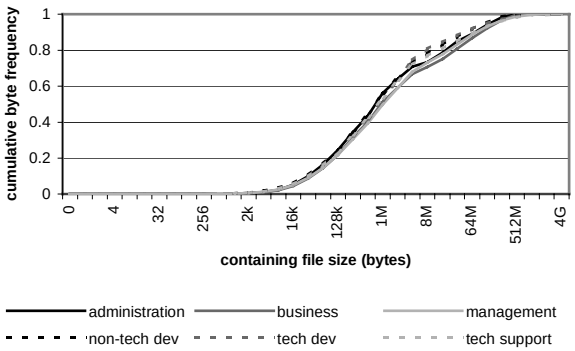

Figure 7: Job-Category CDFs of Bytes by Files Size

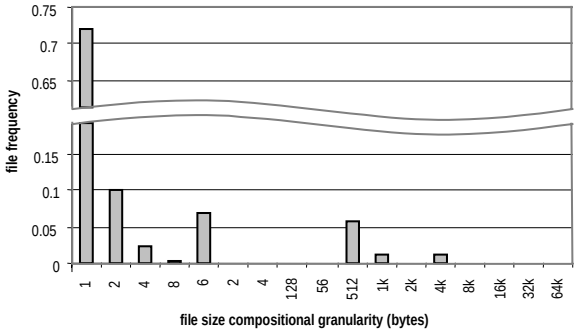

Figure 8: Histogram of Files by File Size Granularity

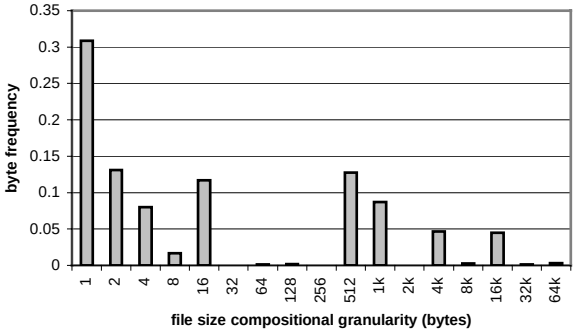

Figure 9: Histogram of Bytes by File Size Granularity

In general, the frequency of files with a CG of 2^n is given by the function $c(n)$, defined as follows:

$$c(n) = 2f^{(n)}(n) - \sum_{k>n} \max\left(2^{n-k} f^{(n)}(n) - f^{(n)}(k), 0\right)$$

The ancillary recurrence is defined as follows, with $f^{(0)}(n) = f(n)$:

$$f^{(k)}(n) = \max\left(f^{(k-1)}(n) - 2^{k-n-1} f^{(k-1)}(k-1), 0\right)$$

In actual practice, we establish a matrix of files, where each file is assigned to a row according to the binary log of the file size and to a column according to the largest power of two that divides the size. We perform the above-described operations only within each row, since gross file size is not uniformly distributed.

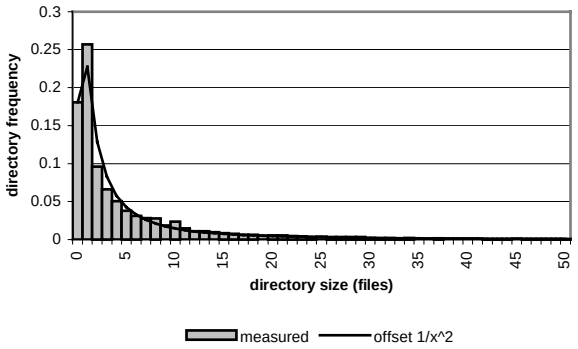

Figure 10: Histogram of Directories by Files

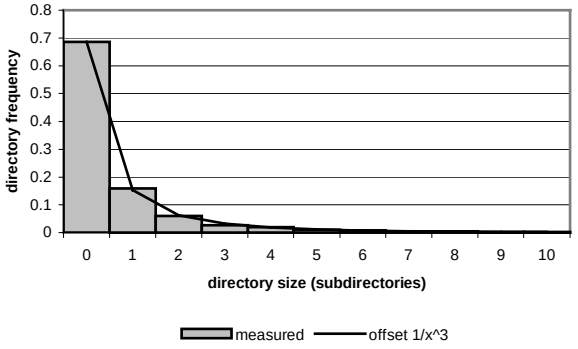

Figure 11: Histogram of Directories by Subdirectories

The result is illustrated in Figure 8. 72% of all files have a CG of one byte, but CGs of 2, 16, and 512 bytes are quite significant. Figure 9 shows a histogram of bytes by containing file CG. The relative frequency of one-byte CGs is only 31%, indicating that compositional granularity is an important factor in the storage of most bytes.

5. DIRECTORIES

Directory sizes are consistent across file systems, independent of user job function, and similar to those found in a recent study [30] of Unix systems; their frequencies fit offset inverse-polynomial distributions. Directory depths are somewhat greater than those in the previous study, and their frequencies fit Poisson distributions.

Figure 10 shows a histogram of directories by size, as measured by count of files in the directory. 18% of all directories contain no files, comparable to the 14% observed by Sienknecht *et al.*, and the median directory size is 2 files. We can approximate this histogram with a mixture of a constant distribution for zero-size directories and an inverse-square distribution with an offset of 2.6, illustrated by the line on the graph. The MDCC is 0.032.

Figure 11 shows a histogram of directories by size, as measured by count of immediate subdirectories in the directory. 69% of all directories contain no subdirectories, 16% contain one, and fewer than 0.5% contain more than twenty. These percentages are close to those reported by Sienknecht *et al.*: 74%, 11%, and 1%, respectively. We can approximate this histogram with a mixture of a constant distribution for zero-size directories and an inverse-cube distribution with an offset of 2.5, illustrated by the line on the graph. The MDCC is 0.006.

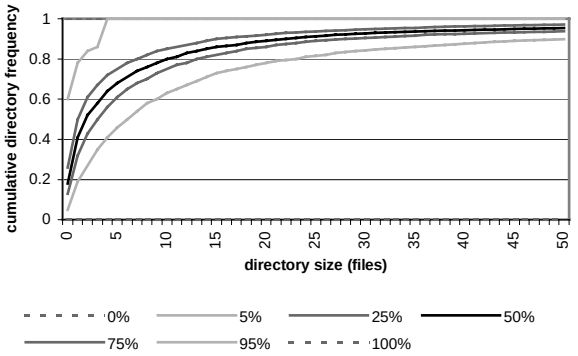

Figure 12: Collective CDF of Directories by Files

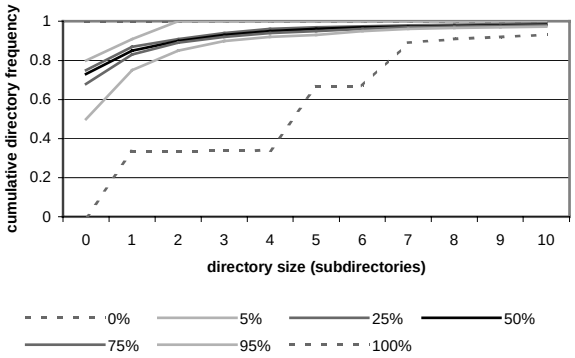

Figure 13: Collective CDF of Directories by Subdirectories

Figures 12 and 13 show collective CDF plots of the histograms illustrated in Figures 10 and 11, respectively. Both figures show that the distribution of directory sizes is relatively consistent across file systems: On 50% of file systems, the median directory size ranges from 1 to 4 files, and on 90% of file systems, it ranges from 0 to 7 files. On 95% of all file systems, the median count of subdirectories per directory is zero.

Figures 14 and 15 show job-category-partitioned CDF plots of the histograms illustrated in Figures 10 and 11, respectively. These figures show that there is very little difference in directory-size distribution among different user job categories, except that the relative fraction of leaf directories is negatively correlated with technical work.

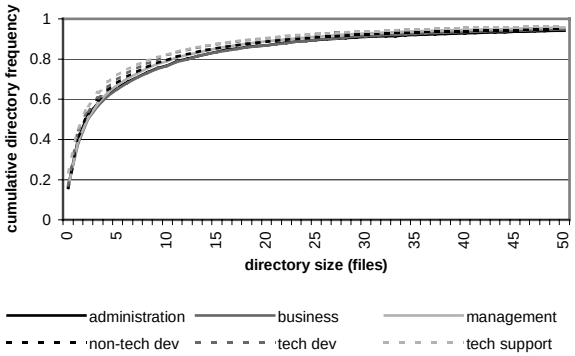

Figure 14: Job-Category CDFs of Directories by Files

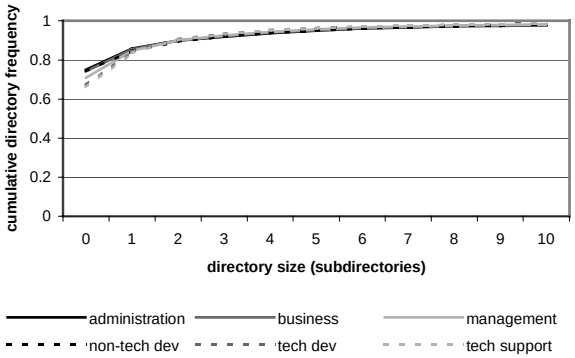

Figure 15: Job-Category CDFs of Directories by Subdirs

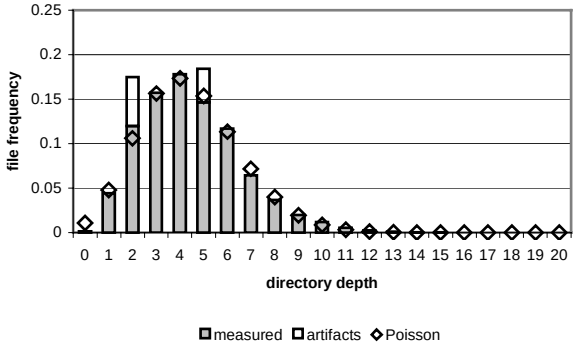

Figure 16: Histogram of Files by Directory Depth

Figures 16 and 17 show the frequency of files and subdirectories, respectively, versus directory depth. We can approximate these histograms with Poisson distributions [11], illustrated by the diamond markers on each graph. For the subdirectory-depth histogram, Poisson (λ = 4.38) is a moderately good fit, with an MDCC of 0.020. It is noticeably poorer for the file-depth histogram, primarily due to the frequency spikes at depths of 2 and 5, both of which are artifacts of the Windows architecture. 5.5% of all files are found in the "System" or "System32" directories at depth 2, and 3.8% of all files are found in the web cache directories at depth 5. Removing these files, as indicated by the lack of shading in Figure 16, yields a distribution that fits Poisson (λ = 4.43) with an MDCC of 0.013.

We found that 15% of all directories are at depth of 8 or greater, which contrasts with the finding by Sienknecht *et al*. that 90% of all directories are within a depth of 5.

6. FILE AGES AND LIFETIMES

File ages and lifetimes vary significantly across file systems and are noticeably dependent upon user job function, with lifetimes showing greater variability and dependence. The frequencies fit hyperexponential distributions, which is consistent with previous approximations [28].

Our scanning program captured three timestamps for each file: creation time, last modification time, and last access time. However, since 16- and 32-bit FAT file systems record only modification timestamps [20], we look exclusively at NTFS file systems (see Tables 1 and 3) for this part of our study. Although the file system accurately tracks file creations, modifications, and

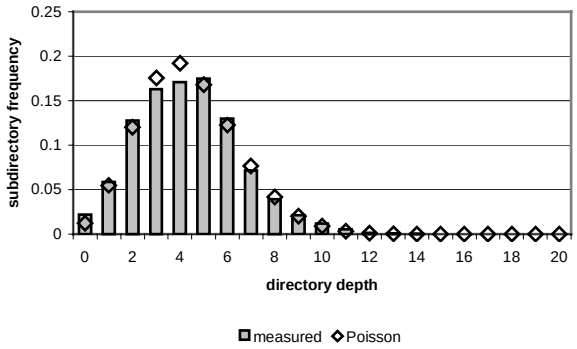

Figure 17: Histogram of Subdirectories by Directory Depth

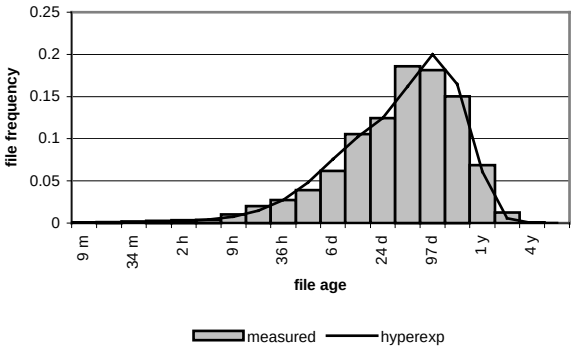

Figure 18: Histogram of Files by Age

accesses, system calls enable application programs to set all three of these timestamps to arbitrary values [20], introducing a potentially significant source of experimental error.

Figure 18 shows a histogram of files by age, calculated as the elapsed time since the more recent of creation or modification, relative to the time of the snapshot. Since it took up to several minutes to scan each file system, file ages less than 9 minutes (found on 0.1% of all files) are within the error threshold and are thus not shown. The median file age is 48 days, which is about three times the value reported by Smith [31] in his analysis of user text-editor data sets from 1975. Studies of short-term trace data [1, 24] have shown that the vast majority of files are deleted within a few minutes of their creation, which we are unlikely to observe with a single snapshot. We can approximate this histogram with a 2-stage hyperexponential distribution [18] (α_0 = 0.80, $\mu_0 = 2^{23.7} = 1.34 \times 10^7$ sec, $\alpha_1 = 0.20$, $\mu_1 = 2^{20.1} = 1.11 \times 10^6$ sec), illustrated by the line on the graph. The MDCC is 0.018.

Satyanarayanan defined *functional lifetime* (f-lifetime) as the difference between a file's age and the time since its last access. F-lifetime indicates the time span over which the data in the file has been demonstrably useful. File creations and modifications are recorded as accesses, so f-lifetime should never be negative; however, 2% of the files in the data set have negative f-lifetimes, presumably due to a commonly used application that modifies file timestamps. We discard the negative f-lifetimes as erroneous, but their mere existence suggests that the results in this section should be viewed somewhat skeptically. Figure 19 shows a histogram of the files by f-lifetime. 44% of the files have an f-lifetime of zero, the median f-lifetime is 8 seconds, and the median exclusive of

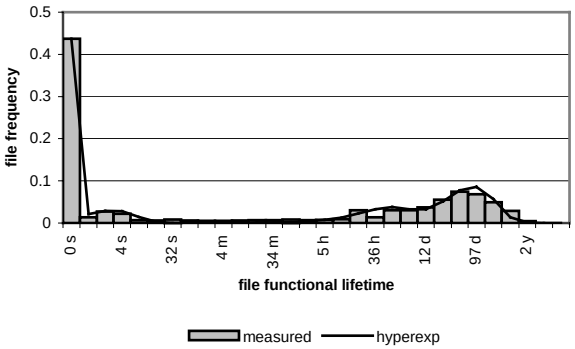

Figure 19: Histogram of Files by Functional Lifetime

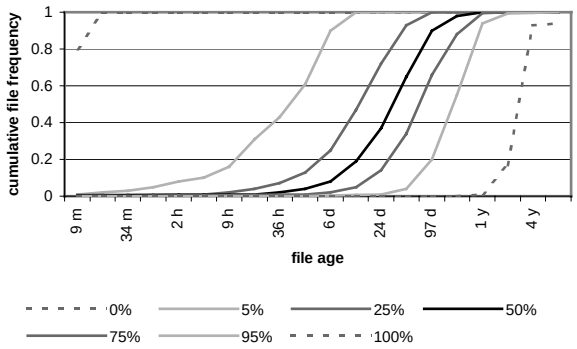

Figure 20: Collective CDF of Files by Age

zero f-lifetimes is 12 days. We can approximate this histogram with a mixture of a constant distribution for zero f-lifetimes and a 3-stage hyperexponential distribution (α_0 = 0.58, $\mu_0 = 2^{23.3} = 1.05 \times 10^7$ sec, $\alpha_1 = 0.21$, $\mu_1 = 2^{18.1} = 2.84 \times 10^5$ sec, $\alpha_2 = 0.21$, $\mu_2 = 2^{2.00} = 4.00$ sec), illustrated by the line on the graph. The MDCC is 0.030. Satyanarayanan approximated f-lifetime with the same distribution mix (with different parameters). The median f-lifetime for his data set was about 30 days, which is far longer than ours and which is near to our median file age.

Figures 20 and 21 show collective CDF plots of the histograms illustrated in Figures 18 and 19, respectively. Figure 20 shows that the distribution of file ages varies significantly across file systems: On 50% of file systems, the median file age ranges by a factor of 8 from 12 to 97 days, and on 90% of file systems, it ranges by a factor of 256 from 1.5 to 388 days. Figure 21 shows that the distribution of f-lifetimes varies widely: On 50% of file systems, the median f-lifetime ranges from zero to 6 days, and on 90% of file systems, it ranges from zero to 97 days, reflecting the strong bimodality seen in Figure 19.

Figures 22 and 23 show job-category-partitioned CDF plots of the histograms illustrated in Figures 18 and 19, respectively. Figure 22 shows that there is little difference in file age distribution among different user job categories, except that non-technical developers (artists, writers, translators, etc.) tend to have somewhat younger files than average. Figure 23 shows dramatic differences in f-lifetime distribution among job categories: 67% of the files on technical support systems have zero f-lifetimes, perhaps because they install many applications solely for testing, which they never use thereafter. At the other extreme,

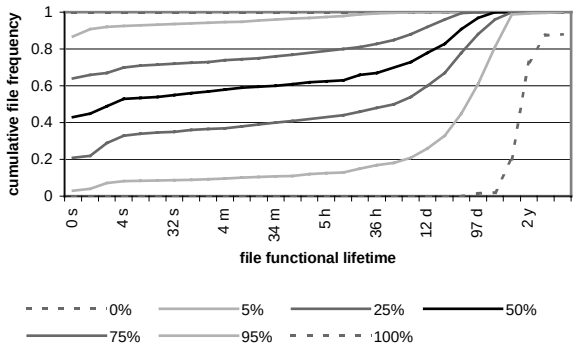

Figure 21: Collective CDF of Files by Functional Lifetime

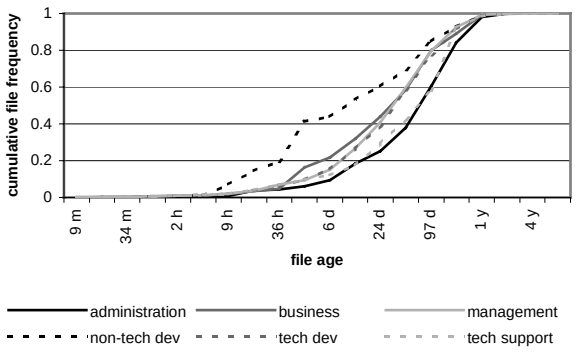

Figure 22: Job-Category CDFs of Files by Age

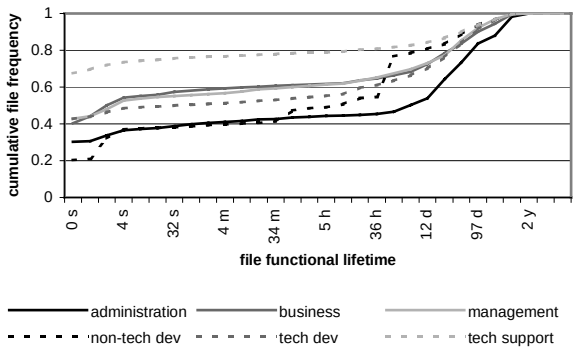

Figure 23: Job-Category CDFs of Files by Func. Lifetime

administration files have very long lifetimes, perhaps reflecting long-term database use. The oddest curve is that for non-technical developers, which shows a 22% spike in the range of 3 to 6 days; we offer no conjecture as to why. It is noteworthy that technical developers fall in the middle of the range: Since most of our data comes from technical developers' systems, it is fortunate that they reflect fairly typical usage.

7. FILE-NAME EXTENSIONS

File-name extensions are strongly correlated with file size and compositional granularity, but weakly with file age and functional lifetime. Extension frequency fits Zipf and Lotka distributions, and extension popularity varies significantly by user job category.

We generated a list of all file-name extensions (the characters following the last period) of five characters or less, excluding those that were purely numeric, resulting in a list of 19,140 entries including the null extension. This list accounts for over 99% of all files in our data set. Table 4 lists the 30 most popular extensions, representing 70% of all files. A histogram of files by these extensions is graphed in Figure 24. We fit a generalized Zipf distribution [12] (b = 7.47, θ = 1.77) to this histogram, illustrated by the diamond markers on the graph. The MDCC is 0.017, and the distribution is a close fit except for the fourth extension, ".dll", which has a significantly greater representation than the approximation indicates.

Rank	Ext.	Lg Size		Lg Age		Lg F-Lifetime		Comp. Gran.		Relative File Frequency (%)						
		mean	std dev	Mean	std dev	mean	std dev	mode	%	total	admin	biz	mgmt	n-t. dev	t. dev	t. supp
1	.gif	10.4	2.26	21.4	2.42	13.9	8.30	1	99	8.9	15.8	17.0	13.6	12.3	7.4	9.7
2	.h	11.8	2.26	22.5	2.12	20.2	5.76	1	99	7.0	0.3	0.4	4.3	2.2	8.5	5.9
3	.htm	11.4	1.74	21.6	2.46	14.2	8.23	1	99	6.4	5.2	7.1	8.5	18.4	5.4	7.5
4	.dll	16.0	2.00	22.4	2.53	18.8	7.38	16	57	6.2	10.2	9.4	7.5	7.1	5.6	7.1
5	–	8.5	3.62	22.5	2.38	17.8	7.95	1	97	3.9	0.8	1.0	1.6	1.0	4.6	4.7
6	.c	12.8	2.52	22.8	2.05	20.7	5.69	1	99	3.5	0.0	0.0	1.7	0.4	4.2	3.7
7	.exe	15.7	2.07	22.5	2.49	18.4	8.00	16	51	3.2	3.9	3.9	3.5	3.1	3.1	3.8
8	.ini	7.3	1.36	22.7	2.04	19.6	6.93	1	98	2.9	1.5	1.3	2.0	1.6	3.4	1.5
9	.cpp	12.7	2.21	22.4	2.11	19.5	6.28	1	99	2.6	0.1	0.1	1.6	0.9	3.2	2.0
10	.inf	12.9	2.63	22.6	2.59	18.8	7.90	1	98	2.5	5.4	4.8	3.2	2.9	2.2	2.6
11	.obj	13.7	2.65	20.4	3.15	15.8	7.98	1	95	2.3	0.0	0.0	0.9	0.1	3.2	0.5
12	.txt	10.2	3.17	21.5	2.65	17.2	7.46	1	96	1.9	2.0	1.9	1.8	1.7	1.6	1.7
13	.bmp	11.7	3.31	22.5	2.22	18.9	7.34	2	98	1.5	1.2	1.3	1.5	1.8	1.5	1.5
14	.lib	15.0	3.07	21.9	2.58	17.6	7.91	2	91	1.3	0.0	0.1	1.0	0.4	1.6	1.0
15	.jpg	13.3	2.11	22.0	2.24	14.1	8.65	1	98	1.2	2.0	2.1	1.8	1.2	1.0	1.2
16	.ico	10.1	0.98	23.1	2.06	19.6	7.34	2	99	1.2	1.4	1.0	1.2	0.9	1.1	1.5
17	.hlp	15.3	2.05	23.1	2.38	19.0	8.55	1	99	1.2	2.0	1.8	1.4	1.6	1.0	1.4
18	.lnk	8.9	0.77	22.1	2.41	20.7	4.57	1	99	1.1	1.9	1.8	1.4	1.5	1.0	1.3
19	.html	11.8	2.21	21.4	2.56	11.0	9.02	1	99	1.0	1.9	1.8	1.3	2.0	0.8	1.0
20	.wav	14.5	2.06	22.4	2.37	16.8	8.72	2	91	1.0	2.4	2.2	1.4	1.1	0.8	1.2
21	.mfc	10.5	2.26	20.1	1.05	16.7	2.17	16	69	0.9	0.0	0.0	0.0	0.0	1.4	0.0
22	.log	12.5	2.42	19.2	2.56	13.8	6.78	1	95	0.9	0.2	0.2	0.2	0.2	0.3	0.5
23	.wmf	12.6	1.50	23.7	2.11	17.6	8.74	2	99	0.9	3.0	2.2	1.7	1.2	0.6	1.4
24	.pdb	17.6	1.75	20.7	2.68	12.7	8.87	1024	86	0.8	0.0	0.0	0.4	0.1	1.1	0.2
25	.tmp	12.4	3.26	21.2	2.56	13.9	8.99	512	37	0.8	3.2	1.7	1.0	1.2	0.6	1.0
26	.rc	10.7	2.29	22.8	2.05	19.7	6.52	1	99	0.7	0.0	0.0	0.5	0.2	0.9	0.7
27	.pnf	14.6	1.36	21.9	2.10	15.6	9.37	4	98	0.7	0.4	0.6	0.8	0.6	0.7	0.7
28	.dbg	13.4	3.04	21.3	2.35	15.2	8.15	4	96	0.7	0.0	0.0	0.2	0.0	0.8	1.0
29	.cur	9.5	0.86	22.7	2.40	19.6	7.50	2	98	0.6	1.4	1.0	0.7	0.6	0.5	0.7
30	.doc	15.3	2.29	22.8	2.06	17.5	8.08	512	84	0.6	1.6	1.5	0.9	1.0	0.4	0.6
overall		12.2	3.39	22.1	2.51	17.8	7.60	1	72	100	100	100	100	100	100	100

Table 4: Properties of Files with Popular File-Name Extensions

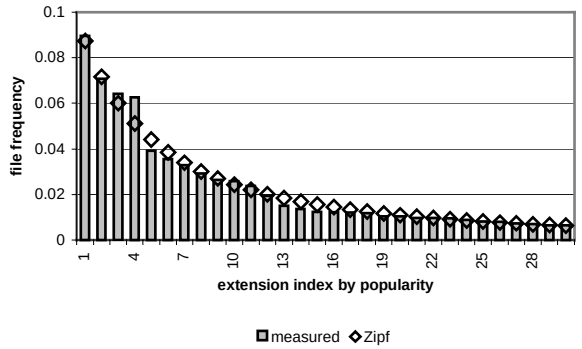

Figure 24: Histogram of Files by Name Extension

Figure 25 shows a histogram of extensions by file count on each file system. 31% of all extensions are found on one file per file system. We fit a generalized Lotka distribution [12] ($\theta = 1.38$) to these frequencies, illustrated by the diamond markers. The MDCC is 0.012. Since a Lotka distribution has an unbounded first moment if $\theta \leq 2$, this distribution is unable to estimate the mean number of files per name extension. This is unsurprising since the extensions in Table 4 dominate the file count.

Table 4 displays the mean and standard deviation of the binary logarithm (lg) of several properties, including size, age, and f-lifetime. We quantify the correlation between file properties and name extension as follows: For each extension and each property, we define the *relative coefficient of variation* (RCOV) as the ratio of the standard deviation to the absolute difference between the overall mean and the mean for the extension. For example, the RCOV of size for the extension ".gif" is 2.26 / abs(10.4 – 12.2) = 1.3. For file size, 37% of all files have extensions whose RCOV is less than one, indicating that the extension is a good predictor of file size for a significant fraction of files. For file age and f-lifetime respectively, only 4% and 2% of the files have extensions whose RCOV is less than one. File-size prediction could perhaps enable disk-space allocation mechanisms to reduce fragmentation [32], by selecting disk regions for newly created files based upon the name extension.

Each row displays the mode of the compositional granularity and the percentage of files accountable to that mode, indicating, for example, that most files with the extension ".exe" and ".dll" have a CG of 16 bytes. This table also shows that there is significant variation in file-type popularity among different job categories.

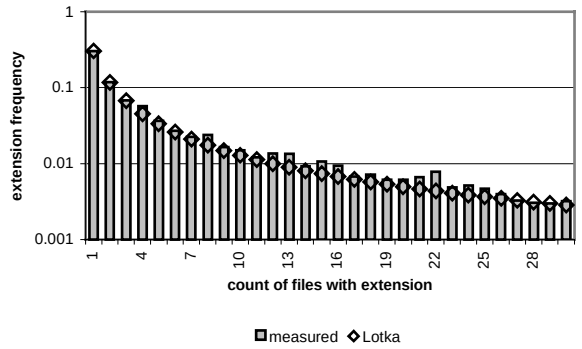

Figure 25: Histogram of Name Extensions by File Count

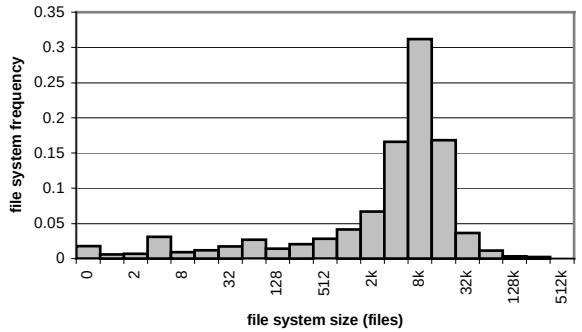

Figure 26: Histogram of File Systems by File Count

8. SYSTEMS

File systems are mostly consistent in size, but the frequency of small systems varies with user job category. File systems are on average only half full, and their fullness is largely independent of user job category. On average, half of the files in a file system have been created by copying without subsequent writes, and this is also independent of user job category.

The total space allotted to each file system is fairly consistent across our sample set: 62% of file systems have 1 to 2 GB of total space, and this mode is consistent across job categories. 81% of file systems have 2 GB or less of total space; however, Table 1 shows that 60% of file systems are 16-bit FAT file systems [21], which are limited to 2 GB in size. The mean number of files per user is 31,835, which is 20 to 26 times that found on Unix systems in 1991 [2] and 1994 [30], largely because each user's machine has a separate installation of the operating system and application programs.

Figures 26 and 27 show the distribution of file system size as measured by count of files. 31% of all file systems contain 8k to 16k files, a mode visible in all job categories. 30% of file systems have fewer than 4k files, and there is a noticeable correlation with job category in the relative frequency of these smaller file systems.

Figures 28 and 29 show the distribution of file system size as measured by count of directories. 28% of all file systems contain 512 to 1023 directories, and 29% of file systems have fewer than 256 directories. The correlation with job function is very similar to that shown by Figure 27.

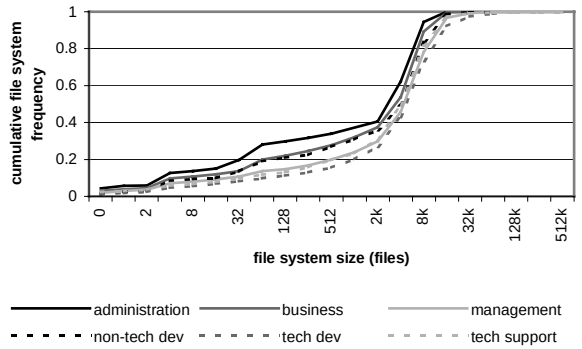

Figure 27: Job-Category CDFs of File Systems by File Count

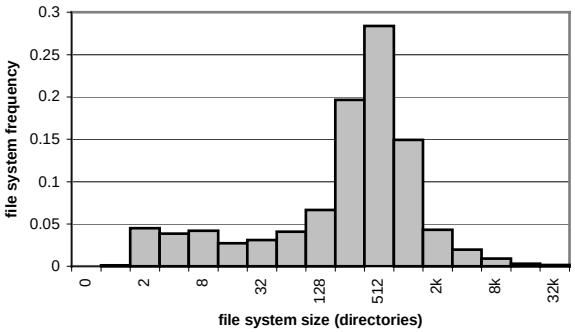

Figure 28: Histogram of File Systems by Directory Count

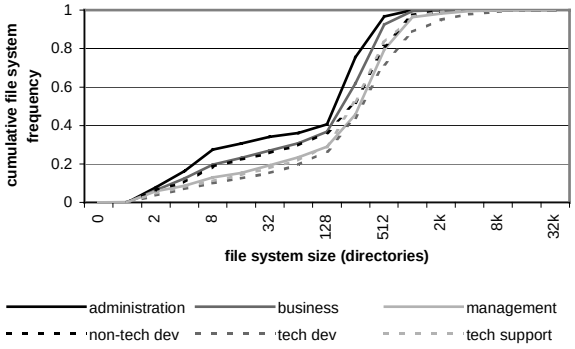

Figure 29: Job-Category CDFs of File Systems by Dir Count

Figures 30 and 31 show the distribution of space usage among file systems. Contrary to earlier studies [2] and informed speculation [8], it is not the case that most file systems are nearly full. Aside from the spike at 0% to 1% usage, the density is relatively flat, rising slightly with increasing usage. The mean space usage is 53%, including virtual-memory paging files and file-system overhead. The contributions of job categories to the low-usage spike are in the same order as their relative fractions of small systems in Figures 27 and 29: A space usage of 0% to 1% is seen on 25% of administrators' file systems but on only 7% of technical developers' file systems.

When a file is copied, the copy's creation timestamp is set to the current time, but its modification timestamp is set to that of the original file. Therefore, a file that is copied and not subsequently

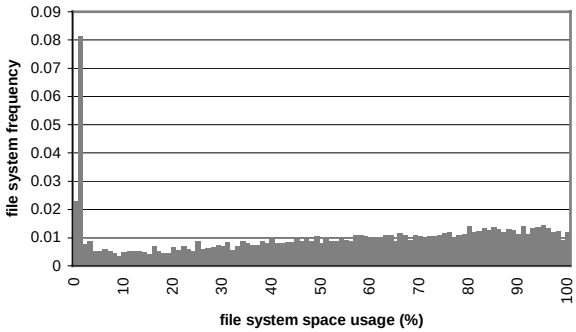

Figure 30: Histogram of File Systems by Space Usage

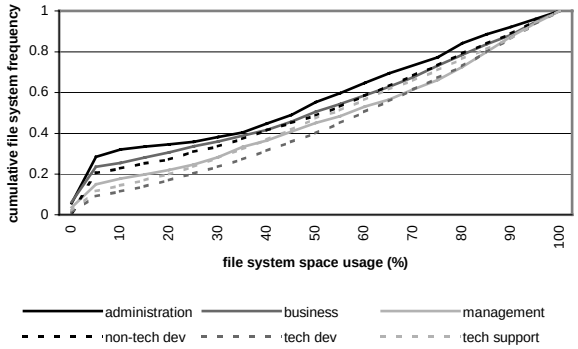

Figure 31: Job-Cat. CDFs of File Systems by Space Usage

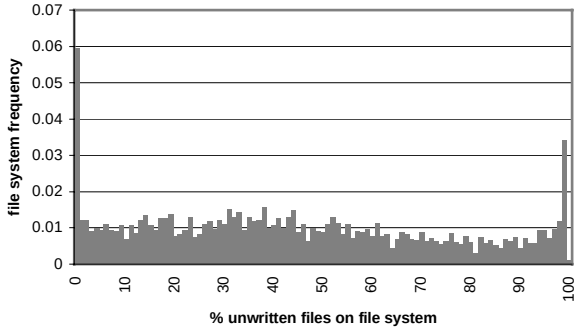

Figure 32: Histogram of File Systems by Unwritten Files

modified will have a modification timestamp that is earlier than its creation timestamp. Figures 32 and 33 show the distribution of such copied-and-subsequently-unwritten files among file systems, limited to NTFS as explained in Section 6. Aside from the spikes at each end of the spectrum, the density is relatively flat. There are noticeable differences among the cumulative curves, but no profound dissimilarities, such as differing modality.

9. CONCLUSIONS

We collected data from 10,568 file systems of 4801 Windows computers in a commercial environment, through voluntary participation of 4418 computer users. The systems contained 140 million files totaling 10.5 TB of data.

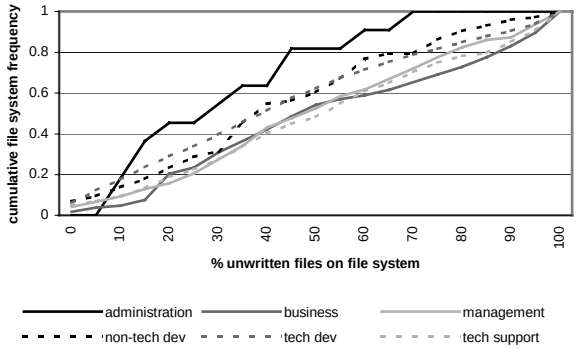

Figure 33: Job-Cat. CDFs of File Systems by Unwritten Files

We developed analytical approximations for several parameter distributions of interest. We found that file sizes fit a log-normal distribution, file ages and lifetimes fit hyperexponential distributions, directory sizes fit offset inverse-polynomial distributions, and directory depths fit Poisson distributions.

We studied the variation of usage parameters across file systems and investigated the relationship between the usage of a file system and the job function of the user. We found that file and directory sizes are fairly consistent across file systems, but file lifetimes vary widely. We also found that job category has little effect on file and directory sizes, but it has a dramatic effect on file lifetimes.

We compared our results to those published for other file systems. Our analytical function for file size is different than that previously posited [28], but our function for file lifetime is the same. Our findings for directory size and structure are similar to previous results [30], but our directory depths are somewhat greater. The mean file size is four times that reported for Unix systems in 1994 [30], and the median file age is three times that reported in 1975 [31].

We analyzed the compositional granularity of files: Files are often composed of records that are sized in powers of two. Although most files have a compositional granularity of one byte, most bytes are in files with greater compositional granularity.

We found that file-name extensions are strongly correlated with file size and compositional granularity, but weakly with file age and functional lifetime. File-extension popularity varies greatly by user job category.

One surprising result is that the file systems in our study were only about half full on average, which contradicts earlier results [2]. This suggests that a log-structured file system [27], which is inefficient when nearly full, may be a more reasonable design today than it has been in the past.

We see the main lessons for file-system designers as follows: File sizes continue the growth trend observed over the last couple decades, and file age is also increasing. It is still true that most files are small and most bytes are in large files. It is reasonable to generalize about certain system attributes, such as file and directory sizes; however, others, such as file ages and types, vary according to user job function, which suggests that adaptive strategies may be helpful in tuning file systems to individuals' work habits. File-name extension can be used at creation time to predict file size and compositional granularity, which may help to optimize disk allocation.

We view our main contributions as follows: First and foremost is the raw data set itself, a sanitized version of which is available by request to any researchers who are interested in its further study (contact johndo@microsoft.com). It is the largest data set yet collected of static file-system contents, and it is the first such published study of Windows file systems. Second, we contributed new analytic approximations for several file-system parameter distributions, some of which contrast interestingly with previously published approximations. Third, we introduced the study of a file attribute, compositional granularity, that is hidden by binning file sizes into contiguous ranges. Fourth, we showed that file-name extension is a good predictor of file size and compositional granularity. And finally, we discovered that file systems are only half full on average.

ACKNOWLEDGMENTS

A data collection project of this magnitude would not have been possible without the assistance of many helpful people. We owe thanks to the MS Research technical support staff for the use of their server for receiving all of the log data; to Robert Eberl for setting up and supporting the server; to Cliff Bates, Angela Schmeil, and the other folks at Product Implementation Management for allowing us to spam the entire Microsoft Corporate Campus; to Damon Wickersham, Bill Conner, Blake Holaday, Sherry Douceur, and Microsoft Mailing Services for their help in distributing incentives to the study participants; to Dan Kusnetzky for allowing us permission to cite reference 16; to Laura Bain, Drew Roselli, and Jeanna Matthews for helping us find previous studies of file system usage data; to Eric Ringger for knowing where to find all sorts of nifty tools; to Mark Crovella for suggesting that we examine the tail of the file size distribution for Pareto-like behavior; to Rick Rashid and our anonymous SIGMETRICS reviewers for their extremely helpful suggestions on organizing and presenting our work; to the twenty thousand people whom we spammed to collect this data, not one of whom flamed us for doing so; and especially to the 4418 people who trusted us enough to actually run our scanning program on their machines.

APPENDIX – Analytical Distributions

Continuous Densities

binary log-normal, $0 < x$:

$$f_l(x;\mu,\sigma) = \frac{1}{x\sigma\sqrt{2\pi}\ln 2} e^{\frac{-(\lg x-\mu)^2}{2\sigma^2}}$$

inverse-polynomial of degree N with offset α, $0 < x$:

$$f_p(x;N,a) = (N-1)a^{N-1}(x+a)^{-N}$$

R-stage hyperexponential, $0 < x$:

$$f_h(x;R,\boldsymbol{\alpha},\boldsymbol{\mu}) = \sum_{i=1}^{R} \alpha_i \mu_i e^{-\mu_i x}$$

Pareto, $k \le x$:

$$f_P(n;k,\alpha) = \alpha k^{\alpha} x^{-\alpha-1}$$

Discrete Distributions

Poisson, $0 \le n$:

$$f_P(n;\lambda) = \frac{\lambda^n e^{-\lambda}}{n!}$$

generalized Zipf, $1 \le n \le N$:

$$f_Z(n;N,b,\theta) = \frac{(-1)^{\theta}\Gamma(\theta)}{\psi^{(\theta-1)}(b+1) - \psi^{(\theta-1)}(N+b+1)}(n+b)^{-\theta}$$

generalized Lotka, $1 \le n$:

$$f_L(n;\theta) = \frac{1}{\zeta(\theta)} n^{-\theta}$$

REFERENCES

[1] M. G. Baker, J. H. Hartman, M. D. Kupfer, K. W. Shirriff, J. K. Ousterhout. "Measurements of a Distributed File System." *13th SOSP*, p. 198-212, Oct 1991.

[2] J. M. Bennett, M. A. Bauer, D. Kinchlea. "Characteristics of Files in NFS Environments." *1991 ACM Symposium on Small Systems*, p. 33-40, 1991.

[3] M. Blaze. "NFS Tracing by Passive Network Monitoring." *1992 Winter USENIX*, p. 333-343, 1992.

[4] P. Barford, M. Crovella. "Generating Representative Web Workloads for Network and Server Performance Evaluation." *1998 ACM SIGMETRICS*, 26 (1), p. 151-160, Jul 1998.

[5] R. Borland, J. Ross. *Introducing Microsoft Windows 98*. Microsoft Press, 1998.

[6] P. Bosch, S. J. Mullender. "Cut-and-Paste File-Systems: Integrating Simulators and File-Systems." *USENIX 1996 Annual Technical Conference*, p. 307-318, Jan 1996.

[7] G. P. Bozman, H. H. Ghannad, E. D. Weinberger. "A Trace-Driven Study of CMS File References." *IBM Journal of Research and Development*, 35 (5-6), p. 815-828, Sep-Nov 1991.

[8] C-M. Chiang, M. W. Mutka. "Characteristics of User File-Usage Patterns." *Systems and Software*, 23 (3), p. 257-268, Dec 1993.

[9] M. E. Crovella, A. Bestavros. "Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes." *1996 ACM SIGMETRICS*, 24 (1), p. 160-169, May 1996.

[10] R. A. Floyd, C. S. Ellis. "Directory Reference Patterns in Hierarchical File Systems." *IEEE Transactions on Knowledge and Data Engineering*, 1 (2), p. 238-247, Jun 1989.

[11] J. E. Freund. *Mathematical Statistics*, Fifth Edition. Prentice Hall, 1992.

[12] G. H. Gonnet, R. Baeza-Yates. *Handbook of Algorithms and Data Structures: In Pascal and C*, 2nd Edition. Addison-Wesley, 1991.

[13] F. J. Gravetter, L. B. Wallnau. *Statistics for the Behavioral Sciences*, Fourth Edition. West, 1996.

[14] S. D. Gribble, G. S. Manku, D. Roselli, E. A. Brewer, T. J. Gibson, E. L. Miller. "Self-Similarity in File Systems." *1998 ACM SIGMETRICS*, 26 (1), p. 141-150, Jul 1998.

[15] J. Griffioen, R. Appleton. "Reducing File System Latency using a Predictive Approach." *Proceedings of the 1994 Summer USENIX Technical Conference*, p. 197-207, Jun 1994.

[16] International Data Corporation. "Client Operating Environments 1998 Worldwide Markets and Trends." IDC Report #16086, May 1998. (cited with permission)

[17] T. Kimbrel, A. Tomkins, R. H. Patterson, B. Bershad, P. Cao, E. W. Felten, G. A. Gibson, A. R. Karlin, K. Li. "A Trace-Driven Comparison of Algorithms for Parallel Prefetching and Caching." In *2nd OSDI*, Oct 1996.

[18] L. Kleinrock. *Queueing Systems: Volume 2, Computer Applications*. John Wiley & Sons, 1976.

[19] D. E. Knuth. *The Art of Computer Programming: Volume 2, Seminumerical Algorithms*, Second Edition. Addison-Wesley, 1981.

[20] Microsoft Corporation. "File Times." *MSDN Library Edition*, Jul 1998.

[21] S. Mitchell. *Inside the Windows 95 File System*. O'Reilly and Associates, 1997.

[22] S. J. Mullender, A. S. Tanenbaum. "Immediate Files." *Software – Practice and Experience*, 14 (4), p. 365-368, Apr 1984.

[23] L. Mummert, M. Satyanarayanan. "Long Term Distributed File Reference Tracing: Implementation and Experience." *Software – Practice and Experience*, 26 (6), p. 705-736, Nov 1994.

[24] J. K. Ousterhout, H. Da Costa, D. Harrison, J. A. Kunze, M. Kupfer, J. G. Thompson. "A Trace Driven Analysis of the UNIX 4.2 BSD File System." In *10th ACM SOSP*, p. 15-24, Dec 1985.

[25] W. J. Orvis. *Excel for Scientists and Engineers*, Second Edition. Sybex, 1995.

[26] K. K. Ramakrishnan, P. Biswas, R. Karedla. "Analysis of File I/O Traces in Commercial Computing Environments." *1992 ACM SIGMETRICS*, 20 (1), p. 78-90, Jun 1992.

[27] M. Rosenblum, J. K. Ousterhout. "The Design and Implementation of a Log-Structured File System." *ACM Transactions on Computer Systems*, 10 (1), p. 26-52, Feb 1992.

[28] M. Satyanarayanan. "A Study of File Sizes and Functional Lifetimes." *8th ACM SOSP*, p. 96-108, Dec 1981.

[29] M. Satyanarayanan. "A Synthetic Driver for File System Simulations." *Modelling Techniques and Tools for Performance Analysis*, p. 439-457, May 1984.

[30] T. F. Sienknecht, R. J. Friedrich, J. J. Martinka, P. M. Friedenbach. "The Implications of Distributed Data in a Commercial Environment on the Design of Hierarchical Storage Management." *Performance Evaluation*, 20 (1-3), p. 3-25, 1994.

[31] A. J. Smith. "Analysis of Long Term File Reference Patterns for Application to File Migration Algorithms." *IEEE Transactions on Software Engineering*, SE-7 (4), p. 403-417, Jul 1981.

[32] K. A. Smith & M. I. Seltzer. "File Layout and File System Performance." Harvard University technical report TR-35-94, 1994.

[33] K. A. Smith & M. I. Seltzer. "File System Aging – Increasing the Relevance of File System Benchmarks." *1997 ACM SIGMETRICS*, 25 (1), p. 203-213, Jun 1997.

[34] D. A. Solomon. *Inside Windows NT*, Second Edition. Microsoft Press, 1998.

[35] C. D. Tait, D. Duchamp. "Detection and Exploitation of File Working Sets." *11th International Conference on Distributed Computing Systems*, p. 2-9, 1991.