A Study of Irregularities in File-Size Distributions

Kylie M. Evans and Geoffrey H. Kuenning
Computer Science Department, Harvey Mudd College
1250 N. Dartmouth Ave, Claremont, CA
{kevans,geoff}@cs.hmc.edu

Keywords: multimedia, MP3, lambda distribution, trace analysis

Abstract

Understanding file sizes is critical to designing and evaluating filesystems. To uncover the effects of audio and video on the distributions found by previous studies, we gathered and analyzed data from a number of different systems at a small undergraduate institution, and attempted to fit curves to the observed size distributions. We found that audio files are already changing the size distributions found in previous years, and video files appear to be starting to do the same. We also found that popular distributions fit poorly in terms of statistical reliability, and that anomalous spikes in distributions are the norm. We discuss the reasons for the failure to achieve fits and implications for file system design and simulation.

1 INTRODUCTION

Over the years, many researchers [1, 2, 3, 9, 14, 16] have investigated the distribution of file sizes in running computer systems. These studies have converged to the conclusion that file sizes can be approximately modeled by a simple lognormal statistical distribution with appropriately selected parameters. The results of these studies have had a significant influence on filesystem design [8, 13].

Today, however, people are storing more media files, such as music, TV shows, and movies, on their personal computers. These large files often violate earlier filesystem design assumptions. For example, common Unix filesystems are optimized primarily for small files [5]. Large files require double- and triple-indirect blocks in the inode, which increases the overhead of storage and access with relatively little benefit (since large files are primarily accessed sequentially [10, 12]). Since many aspects of future system design (disk layout, buffer caches, backup methods, etc.) depend on the sizes of files, it is important to understand the kinds and sizes of files being stored today, and to be able to predict future trends in file characteristics.

A thorough understanding of file-size characteristics would also allow investigators to generate realistic samples of files for testing purposes, rather than experimenting solely on real-world systems under irreproducible conditions. Unfortunately, as we will show in Section 6, simple statistical models are not representative of reality, suggesting that more complex approaches will be needed for such simulations. To the contrary, we found that media files introduce significant irregularities that must be accounted for separately. Even on a machine without media files, a single program can create many files of similar size, making simple models invalid.

2 PROBLEM

As part of a project to design a new filesystem targeted at large files, we undertook an investigation of modern file sizes. Initially, we had intended to use models based on Douceur and Bolosky's work [2]. However, we soon realized that we had access to a data source unavailable to previous researchers: large MP3 collections belonging to undergraduate students at our institution. A preliminary study of one machine confirmed that media files significantly skew the overall file-size distribution.

Therefore, we initiated a new study to better characterize the impact of multimedia files on file-size distributions. We obtained permission to study systems belonging to a number of students, two different academic departments, and the general-purpose college server. Based on the data we collected, we then attempted to develop models of file sizes that would incorporate the effects of larger files.

Previous studies, as well as this one in initial stages, looked for a simple mathematical model to describe the file-size distribution. However, we were not able to find a model that was accurate enough for reliable use. Individual computers exhibit "specialty bumps," which are peaks caused by an often-used program that either creates files in a limited size range or has a large number of similarly sized data files associated with it. These peaks vary from machine to machine, and do not fit into simple statistical models.

We also found subsidiary peaks associated with common multimedia file sizes. We believe that these peaks represent an important consideration for future filesystem design.

3 RELATED WORK

As mentioned earlier, a number of studies have been done on filesystem contents, covering Unix, Windows, and other systems. These studies examined such parameters as file size, timestamps, and extensions, among others. Significant observations of these studies include the lognormal distribution of file sizes, and that the mean file size has been increasing with time. More extensive discussions of earlier studies can be found in Douceur and Bolosky [2] and in Mummert and Satyanarayanan [6].

The most recent relevant studies are those of Douceur and Bolosky [2] and Roselli *et al.* [12]. Douceur and Bolosky studied over 10,000 filesystems on 4801 machines in active use at Microsoft Corporation. This work is notable for the large sample size, the investigation of machines used for non-programming purposes, and the care used in deriving results. Although Roselli *et al.* worked with smaller data sets, their conclusions generally support Douceur and Bolosky's. However, today's rapid pace of change has already invalidated some of the observations made by both of these research groups. In particular, neither study investigated media files as a separate class. (Douceur and Bolosky did investigate still pictures and small sound clips used for effects, but did not have access to significant numbers of full-length music and video files.)

4 APPROACH

After reviewing earlier work, we designed our study specifically to analyze the effect of media files on the conclusions reached by previous researchers. To do so, we located a number of computers that contained large collections of sound and video files, and compared their files to those of more traditional filesystems.

4.1 Data Collection

With three exceptions, all of the computers studied were located on the campus of Harvey Mudd College. Harvey Mudd is a small undergraduate institution (680 students) specializing in engineering and the sciences. Because of its highly technical nature, we expected to find a larger than usual proportion of students who were using their computer(s) for cutting-edge multimedia applications.

The three computers not on the Harvey Mudd campus were located at the Marine Biological Laboratories (MBL) in Woods Hole, Massachusetts. We collected data from these machines for comparison purposes, to help ensure that the Harvey Mudd results were not unduly skewed due to the nature of the college.

We gathered the data in the summer of 2001, from both personal computers and large servers. There were six systems running various editions of Windows, eight running Linux distributions (including the three machines from MBL), five that dual-booted both Windows and Linux, and three large Unix-based campus servers. For four additional computers, one of which ran MacOS X and three of which ran various Windows distributions, we were able to collect data only on video files. We attempted to collect full data from Macintoshes as well, but we discarded the non-video data because we were only able to collect two complete samples (one each from OS IX and OS X). The studied computers were selected by asking cooperative system administrators and students to volunteer, via personal contacts and a posting on a general campus mailing list. We looked at the overall file contents of each system, as well as separating out any MP3s, `.wavs`, and video files. The characteristics of the machines and file types can be found in Table 1.

The data for the Linux and Unix machines was collected by a shell script that used `find` (as root) to locate all files. To maintain privacy, we discarded directory and file names, keeping only the size in bytes, type, modification date, directory depth, extension, and the number of blocks used by the file. (The size collected was the file length kept by the operating system, so in terms of blocks, we overestimated the size of sparse files. Fortunately, sparse files represent a vanishingly small percentage of all files on a real system, so our results are not significantly affected.) The data for each system was kept separately, although we combined the data for analysis.

Windows data was collected similarly: `dir/S` produced a complete file list, which was immediately fed to a script that discarded private information, keeping the same information as for Unix except for the block size (which is not available under Windows).

Of the three servers studied, two belonged to individual departments (Computer Science and Mathematics) whose faculty and students make extensive use of computers, and the third provides general mail and computing services to the entire campus.

As it turned out, our efforts to maintain privacy were occasionally problematic. For example, we had expected that `.avi` files (a video format) would generally be large. However, the data revealed a number of extremely small `.avi` files. Fortunately we were able to contact the owners of these files individually to ask whether they could explain the small sizes, and learned that they were samples associated with particular software packages such as video drivers. (This example illustrates the "special file" phenomenon discussed further in Section 6.)

4.2 Bias

There are several possible sources of bias for our data. First, all of the data (with the exception of the three computers from MBL) is from machines at Harvey Mudd College, which has an admittedly narrow population. Further, the participants were self-selected from summer campus residents who have their own computers.

In addition, some people were directly selected for MP3 information based on our knowledge of their large MP3 collections. Although this selection is useful for our study of MP3 sizes, it could introduce bias if the remainder of their files were not representative of the population at large.

Since we asked for the full names of video files so that we could categorize the information they contained, some people gave us permission to keep file names, but only provided video files. If their non-video files were very different from that on the other computers (such as software for video edit-

Fig.	OS Type	Files Incl.	# of Machines	#of Files	# of Zero-Size Files	# of Media Files
1	Linux	all	8	901,055	17,242	2373
2	Multi	all	3	2,632,014	104,132	567
3	Windows	all	6	204,037	2118	20,740
–	Dual	all	5	752,192	9683	11,806
4	Windows	non-media	6	183,297	2108	0
5	Linux	HMC non-media	5	362,847	6039	0
–	Linux	MBL non-media	3	535,835	11,199	0
–	Dual	non-media	5	740,386	9666	0
6	all	MP3	18	18,303	25	18,303
7	all	`.wav`	20	18,089	6	18,089
8	all	`.avi`	20	566	0	566
9	all	MPEG	19	861	2	861
–	All OSes	all files	22	4,489,298	133,175	35,486

Table 1. General characteristics of the machines and files studied. The "All OSes" line gives the totals for all files across all machines, which is equal to the sums of the first four lines of the table. The total of the last four lines differs due to the inclusion of statistics for 4 users who provided only information on media files.

ing), this approach could be another source of bias.

Finally, the nature of our institution and the study necessarily means that our sample size is quite small. The limited sample size suggests that our numeric curve fits should be viewed as indicative, not representative. However, as we argue in Section 7, a larger sample size is unlikely to lead to significantly more predictive power.

4.3 Graphing and Data Analysis

To study `.mp3`, `.wav`, `.mpg/.mpeg` and `.avi` files, we filtered the full data from each computer to select only those files with the extension being graphed. For `.mpg/.mpeg` and `.avi` files, we had additional data from people who were only willing to supply us information on those types of files. To study video files, we filtered the overall data for the appropriate extensions and then combined it with the video-only data to produce the samples that we analyzed.

We first attempted to analyze the data using traditional histograms, but we found that histograms were overly sensitive to parameter choices and tended to hide many interesting features. Instead, we have chosen to use kernel density estimation [15], which produces probability density functions that reveal much more of the structure of the underlying distribution. We used the simplest non-adaptive estimation method, with Gaussian kernels, chose the bandwidths by hand, and graphed the size distribution of all files on a per-OS basis. Because media files clearly distorted the observed curves, we also generated per-OS graphs with media files removed based on the file extension (`.mp3`, `.wav`, `.avi`, `.mpg`, and `.mpeg`), and combined-OS graphs for each of four major media types (`.avi`, `.mp3`, `.wav`, and `.mpg/.mpeg`).[1] We chose to filter out and separately analyze only those four media types because they seem to be the most common and have the greatest effect on the file-size curve. In this paper, references to "non-media files" mean all files except those with one of these five extensions, and "media files" refers to those five extensions only.

[1]We did not treat files with double extensions, such as `.wav.gz`, as media files because their sizes would not have been reflective of common media sizes.

Our hope was that by separately studying media files, we would be able to fit two statistical distributions, one each for non-media and media files, and sum them to generate an overall distribution. Although we were able to fit distributions to each of these components, the fit is not good, and we now believe that fitting a simple curve will not provide a statistically reliable model (Section 7).

We also extrapolated the information from MP3s, `.wavs`, and the current video data in an attempt to predict the characteristics of filesystem contents as video files become more common.

We graphed our data as a probability density function (pdf) of file sizes. The density function was generated from a kernel density estimation [15] using a Gaussian kernel of varying bandwidth. In all graphs, the X axis is the base-2 logarithm of file size and the Y axis represents the probability density at that size. For convenience, the X axis is also marked with equivalent file sizes in bytes.

We chose logarithmic analysis for two reasons. First, the extremely wide range of file sizes (single bytes to gigabytes) means that a linear graph would hide important information. Second, previous studies found a lognormal or nearly lognormal distribution of file sizes [2, 3, 16]; on a logarithmic scale this curve shows up as the familiar Gaussian distribution, simplifying analysis.

As in previous studies, we observed large numbers of zero-sized files. Since the logarithm of zero is $-\infty$, we were not able to incorporate these files into our logarithmic analysis.

We considered artificially placing these files at an X coordinate of -1, representing a file size of 0.5 bytes, but discarded this option because we found that it distorted the density estimates. Instead, following previous researchers [2], we believe that the best way to model zero-sized files is to add an independent impulse function to the distribution.

4.4 Curve Fitting

We looked for curves to fit the distribution of the log of the file size (excluding the zero-size files). Following Douceur and Bolosky, we believe that the proper way to model file sizes is to use a mixture of distributions with a Dirac delta function at zero to account for those files. For this reason, our graphs do not show zero-size files.

At first we looked for common curves to fit to our distributions, such as Gaussian (normal). However, the fitted normal failed a Kolmogorov-Smirnov test [4, 7] at all levels of significance. This is consistent with Douceur and Bolosky's findings [2]:

> At a 0.01 level of significance, all of our continuous approximations fail the Kolmogorov-Smirnov test, and all of our discrete approximations fail the chi-square test. 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.

The closest we came to a normal distribution was for Windows machines with media files removed. In this case, a Gaussian provided a reasonably good visual fit, but still failed the K-S test. For the other operating systems (including dual-boot machines) and types of files, we were not able to achieve a good fit to the observed data with a normal distribution.

One of the primary tests we used to determine if the observed data was indeed normal was to calculate the skewness, given by

$$\frac{\sum(X_i - \mu)^3}{(N-1)\sigma^3}$$

and kurtosis, given by

$$\frac{\sum(X_i - \mu)^4}{(N-1)\sigma^4} - 3$$

where μ and σ are the mean and standard deviation, respectively, observed from the sample data [7]. The normal distribution has a skewness and kurtosis of zero.

Roughly speaking, data with a small skewness and a kurtosis smaller than ± 0.5 can probably be approximated by a normal curve. Larger values of skewness indicate an asymmetric distribution, with positive values of skewness associated with larger tails to the right of the mode.

The kurtosis value indicates the "peakiness" of a distribution. If it is positive, the data follows a distribution that is steeper than a normal, while a negative value indicates a flatter curve.

As discussed above and detailed in Section 5, the non-Windows file distributions could not be explained as normal. After investigating several options for fits, we chose the lambda distribution [11] for its flexibility. (See Appendix A for the distribution's defining equations.) We used the lambda distribution for all fitted curves in this paper. All curve fits were done numerically, rather than graphically.

It is important to note, however, that the lambda curves are also poor fits. This issue is discussed further in Section 7.

In addition to the methods detailed above, in a few cases we applied additional techniques to validate our analysis. These approaches are discussed in Section 5 along with the accompanying data. We did not do the additional analysis in every case due to the cost and the fact that the extra checks supported our original conclusions.

5 RESULTS

5.1 Choosing a Curve

We began by combining the observed data for all files from a given operating environment, estimating the density and graphing the result, and numerically fitting curves. The normal curve and the lambda distribution were determined to be appropriate for the study. However, as shown in Figure 1, the lambda distribution, with four parameters as opposed to two, could be tuned to fit the graphs more precisely.

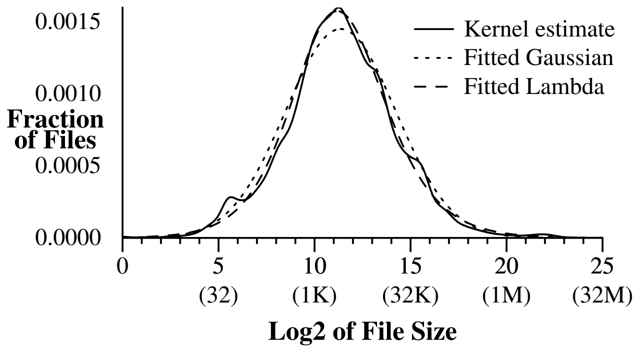

Figure 1. Kernel density estimate (bandwidth = 0.335) of the distribution of sizes of Linux files (media files included), with numerically fitted lambda and normal distributions.

The parameters for the fitted distributions are given later, in Tables 3 and 4. In comparing the parameters of these fits, it is important to recall that λ_1 is a location parameter *similar* to the mean, but is not the actual mean. Table 2 gives statistical parameters for all of our graphs. Here, for example, one can see that although the λ_1 values are generally smaller in Table 3 than in Table 4, the mean sizes for all files (Figures 1 through 3) are larger (as one would expect) than the corresponding means when media files are removed (Figures 4 and 5). This effect is most visible in the arithmetic means, which are the more sensitive to the presence of large files.

Fig.	OS Type	Files Incl.	Arith. Mean	Geo. Mean	Median	Mode	Kurtosis	Skewness
1	Linux	all	76K	2.6K	2.5K	2.4K	0.96	0.15
2	Multi	all	48K	2.2K	1.9K	0.26K	0.57	0.21
3	Windows	all	720K	11K	10K	16K	0.39	0.41
–	Dual	all	150K	3.2K	2.7K	2.4K	1.1	0.33
4	Windows	non-media	160K	6.8K	6.7K	12K	0.15	0.0088
5	Linux	HMC non-media	22K	2.3K	2.6K	2.4K	0.80	-0.13
–	Linux	MBL non-media	100K	2.7K	2.4K	2.5K	0.80	0.20
–	Dual	non-media	75K	2.9K	2.7K	2.4K	0.76	0.14
6	all	MP3	5.8M	3.8M	3.8M	4.0M	11.3	-1.01
7	all	`.wav`	440K	31K	30K	32K	3.7	0.72
8	all	`.avi`	88M	5.1M	9.2M	165M	†	†
9	all	MPEG	53MB	1.8MB	5.4MB	4.1M	†	†
–	All OSes	all files	96K	2.6K	2.2K	3.3K	0.92	0.30

Table 2. Statistical parameters for all categories studied in this paper, given to 2 significant digits. Arithmetic means and medians were calculated on raw sizes, geometric means were calculated logarithmically, and modes were taken from the kernel density estimates, which had the given bandwidths. The kurtosis and skewness are given in one column; the kurtosis is the first number and the skewness the second. †These distributions were so far from a Gaussian that calculating skewness and kurtosis made no sense.

5.1.1 Multi-User Machines

For the multi-user servers on campus we show only the all-files results (Figure 2), since only 0.022% of the files had one of the five media extensions. The graphs of multi-user machines with media files excluded appeared identical to those shown in Figure 2, and the fitted distribution differed only slightly in its constants. As indicated in Table 2, the curve fitted is slightly more peaked than a Gaussian distribution.

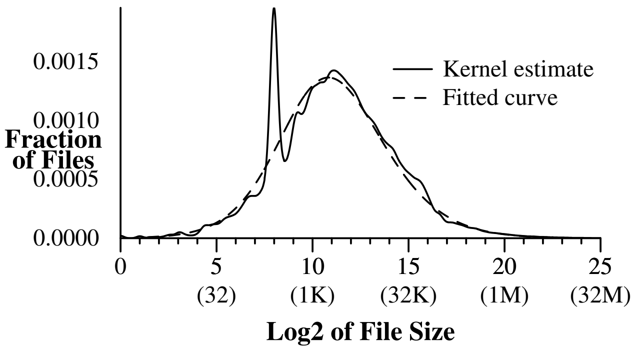

Figure 2. Kernel density estimate (bandwidth = 0.168) of the distribution of sizes of files from the multiuser server machines from HMC (media files included), with a numerically fitted lambda distribution.

5.1.2 Single-User Machines: All Files

Figure 3 shows files found on Windows-only computers. The Linux-only and dual-boot distributions had essentially the same shape. (On dual-boot machines, we could not separate the files by operating system, because the users had all run our script under Linux with the Windows partition mounted, and our process of discarding pathnames also lost

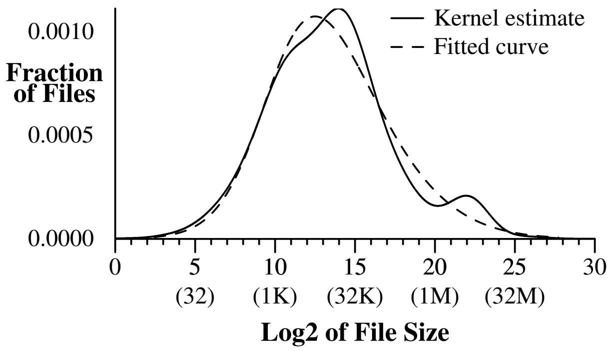

Figure 3. Kernel density estimate (bandwidth = 1.000) of the distribution of sizes of Windows files (media files included), with a numerically fitted lambda distribution.

the information that would allow the partition to be distinguished. For that reason, dual-boot systems were studied separately in all cases.)

The curve shows the difficulty of achieving a good fit when media files are included in a distribution as the tail contains a secondary mode. This mode is especially noticeable at 4.0M ($\log_2 = 22$) in Figure 3, and the distribution is visibly skewed to the right to accommodate it. The skewness of 0.41 (as opposed to 0.0088 without media files) emphasizes the impact of the media files on this mode.

The parameters of the lambda distributions for the all-files fits are given in Table 3.

5.1.3 Single-User Machines: Non-Media Files

Having developed an overview of the total file-size distribution, we then turned to a separate characterization of media and non-media files in an attempt to find multiple curves

OS	λ_1	λ_2	λ_3	λ_4
Windows	12.1	0.0355	0.0610	0.117
Linux	11.1	0.0123	0.0188	0.0217
Dual-boot	11.2	0.00720	0.0106	0.0144
Multi-user	10.7	0.0317	0.0505	0.0658

Table 3. Parameters of fitted lambda distributions for observations of all files, including media files. All parameters are given to 3 significant figures.

OS	λ_1	λ_2	λ_3	λ_4
Windows	12.7	0.0498	0.108	0.110
Linux-HMC	11.4	0.0210	0.0355	0.0310
Linux-MBL	11.1	0.0207	0.0316	0.0393
Dual-boot	11.3	0.0213	0.0344	0.0398

Table 4. Parameters of fitted lambda distributions for observations of non-media files. All parameters are given to 3 significant figures.

that could be used to characterize the whole distribution. For non-media files (all but MP3, `.wav`, `.avi`, and MPEG), we found that the curves seemed similar in flavor, although they differ in details.

The kurtosis and skewness of the Windows data (Table 2) indicates that a normal distribution might fit the observed data. However, the data failed a Kolmogorov-Smirnov test for goodness of fit. Instead, we fitted a lambda distribution (see Figure 4) primarily to remain consistent with our other fits.

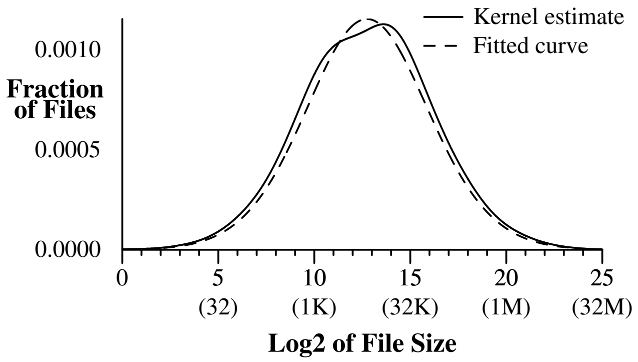

Figure 4. Kernel density estimate (bandwidth = 1.000) of the distribution of sizes of non-media Windows files, with a numerically fitted lambda distribution.

The kurtosis and skewness of the data from both the Linux and dual boot (Linux and Windows) systems indicated that these distributions were more peaked than a normal, a lambda distribution was fitted to them as well. Since dual boot and Linux machines had the same curve without media files, we have only shown the Linux graph, Figure 5.

The parameters used for the fitted curves discussed in this section can be found in Table 4.

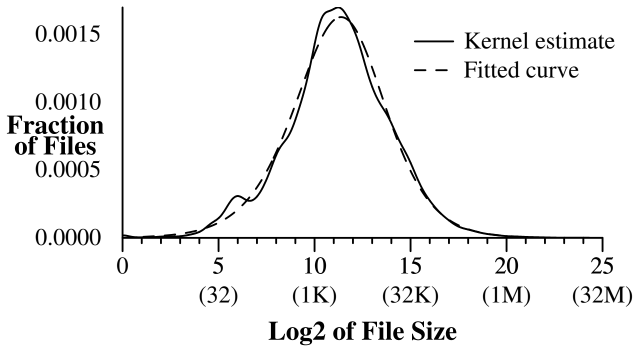

Figure 5. Kernel density estimate (bandwidth = 0.335) of the distribution of sizes of non-media Linux files from HMC with a numerically fitted lambda distribution.

Type	λ_1	λ_2	λ_3	λ_4
MP3	22.1	-0.279	-0.169	-0.123
`.wav`	14.5	-0.0713	-0.0577	-0.0815

Table 5. Parameters of fitted lambda distributions for histograms of audio files. All parameters are given to 3 significant figures.

5.1.4 Media Files—MP3s, Wavs, Avi Files, and MPEGs

Media files are in general larger than the average computer file. In addition, most of them fall in clusters since each movie, TV show, song, and CD-length sound file will have roughly the same size as all the other files in the same category, resulting in clusters around particular sizes. For these reasons, as well as because of their uneven influence on file-size distributions as shown in Figure 1, they merit separate study.

Audio Files Based on their large kurtosis values presented in Table 2, both MP3 and `.wav` files were determined to be non-normal and were fitted with a lambda distribution. Table 5 gives the parameters of the lambda distributions for audio files.

Although the MP3 file distribution is bimodal (Figure 6), we chose to fit a single distribution rather than a mixture, partly due to a desire to avoid an overly complex model, and partly because of the difficulty of mechanically separating the sources of the two modes in the absence of supplementary information.

Table 5 gives the parameters of the fitted lambda distributions for audio files, shown in Figures 6 and 7.

Video Files The systems we studied used two primary formats for video files. The most popular is DivX compression (`.avi` extension); MPEG (`.mpg` or `.mpeg` extension) is also heavily used despite its poorer compression performance. In both studies, our analysis was constrained by the relatively small total number of files (1427), so our results should be interpreted with great caution.

As Figures 8 and 9 show, the distributions of both MPEG

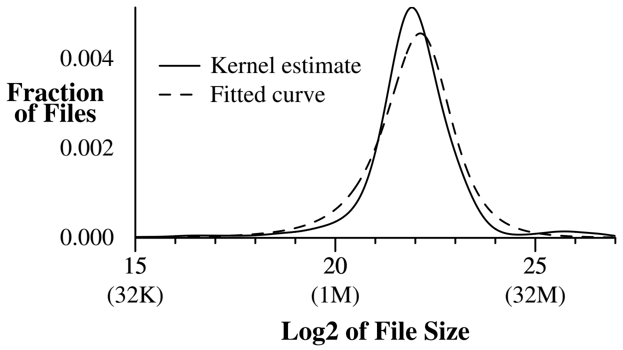

Figure 6. Kernel density estimate (bandwidth = 0.335) of the distribution of sizes of MP3 files with a numerically fitted lambda distribution.

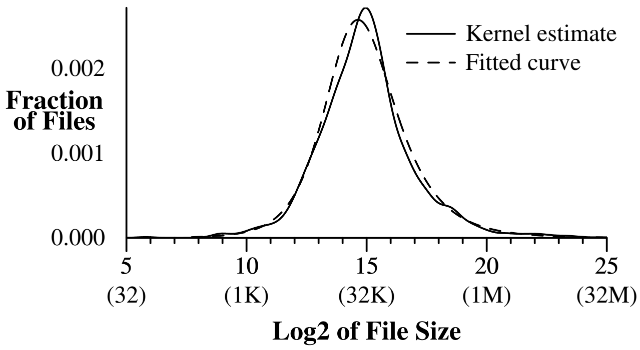

Figure 7. Kernel density estimate (bandwidth = 0.335) of the distribution of sizes of `.wav` files with a numerically fitted lambda distribution.

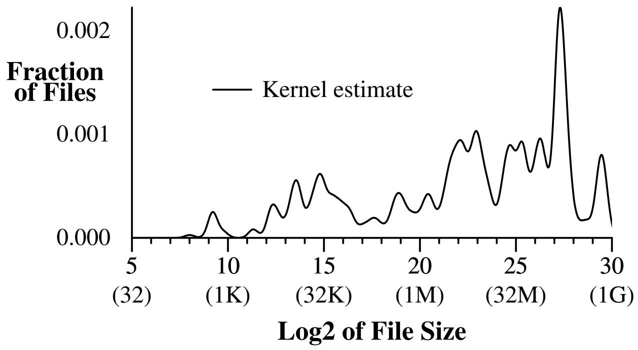

Figure 8. Kernel density estimate (bandwidth = 0.252) of the distribution of sizes of `.avi` files. This estimate is undersmoothed because estimates with a larger bandwidth obscured important details. The reader should not give excessive weight to the smaller variations, which are most likely due to noise in the observations.

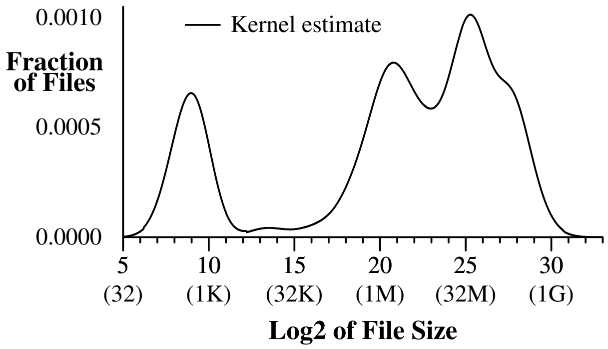

Figure 9. Kernel density estimate (bandwidth = 1.000) of the distribution of sizes of MPEG files.

(`.mpg` and `.mpeg` extensions) and `.avi` files are rather flat and noisy, with multiple spikes. Not until these files are more prevalent and we can determine if the curve becomes smoother will we be able to tell what distribution best fits them.

A surprising result of our study is the relatively small number of video files. In total, there were only 566 `.avi` files and 861 `.mpeg` files. We had expected to discover a much larger sample. We theorize that this is because inexpensive disks are still small enough to prevent students from acquiring large video collections, but the question merits further study, especially as 100+ GB disks become common.

5.2 Verification Methods

One concern we had was that the data would be biased by the nature of Harvey Mudd, since it was only collected from that campus. To determine how serious the bias was, we studied non-media files on Linux boxes at MBL to ensure that they displayed similar size distributions. As shown in Tables 2 and 4, we found that the data from both institutions had very similar distributions and curve fits. Although the MBL distribution is somewhat flatter and wider, both curves had the same general shape as that shown in Figure 5.

For additional verification of the validity of our analysis, we applied the standard technique of calculating the kurtosis of a randomly selected sample of half the total data for comparison to the entire sample. When tested on the dual boot data, this test produced values ranging from 0.77 to 0.81, reinforcing the hypothesis that the distribution of the logarithm of file sizes from dual-boot computers does not fit a normal curve.

5.3 Mode and Spike Analysis

Although lambda distributions were fitted to the KDEs of the file sizes, there were sometimes spikes that could not be explained by any simple mathematical model. There were also some graphs that exhibited multiple modes, even when the kernel density estimate was calculated with a relatively large bandwidth. These spikes and modes merit some explanation as to their cause and the effect they have in modeling the file distribution.

5.3.1 PowerPoint Spike

One peak in Figure 2, at approximately 181 to 362 bytes ($\log_2 = 7.5 - 8.5$) and centered at 256 bytes ($\log_2 = 8$), does not fit the lambda distribution. It contains about 47,500

HTML files that were apparently created on one of the three machines by Macintosh PowerPoint 98 when presentations were saved as HTML. We have learned that although these files do not contain information that contributes to the presentations, each slide contains a reference to one of them, so that they cannot be safely destroyed. The same peak also contains 2,818 `.bmp` files of 16x15 pixels in size, which are probably font characters or icons. Finally, it also has a large number of files with extensions `.pl`, `.sml`, `.c`, `.h`, `.cc`, `.sh`, `.rex`, `.java`, `.class`, and `.cpp`, which are all programming-language source files. The frequency of these files at a small size is probably at least partly a result of the fact that the machine in question is used to teach programming. However, we were still surprised by the extremely small sizes (especially for those source files that were written in a relatively verbose language like C++), and plan to investigate that phenomenon more closely in the future.

Since there were only three sample multi-user machines, it is unsurprising that the files from one of them had a strong influence on the overall distribution. However, as discussed further in Sections 6 and 7, we also believe that this "anomalous" peak provides an important indication of the true behavior of real-life machines.

5.3.2 MP3 Bump

In Figure 3, the graph of all Windows files, the fitted curve is skewed to the right to account for the small mode at approximately 4 MB ($\log_2 = 22$). It contains the most common MP3 sizes. In the Linux and dual boot curves the tail similarly does not fall off as expected at that file size. In fact, we found that for single computers the tail bump could be even more obvious, as in Figure 10.

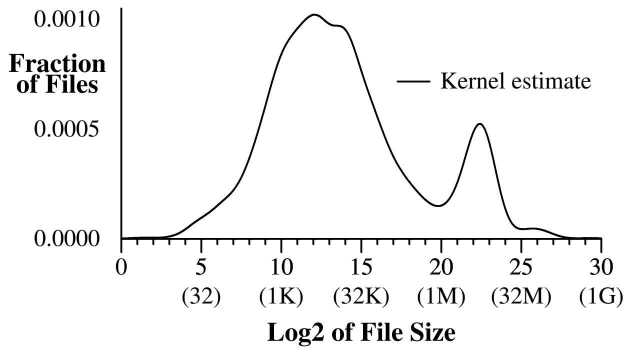

Figure 10. Kernel density estimate (bandwidth = 0.668) of the distribution of sizes of files from a single Windows machine.

5.3.3 MP3's Multiple Modes

The distribution of MP3 file sizes (Figure 6) is subtly bimodal. The first mode is in the 2-4 MB ($\log_2 = 21 - 22$) range, while the second (much smaller) mode is in the 32-64 MB ($\log_2 = 25 - 26$) range.

In our data, fully 77% of MP3 files are between 2 and 8 MB. We believe this is because most MP3 files are used to hold popular songs, which (due to the needs of commercial broadcasting) are primarily around 3 to 5 minutes in length. From a sampling of files, we found these radio-length songs to be 2 to 8 MB in size, which nicely accounts for the peak in that range.

A sampling of files in the second mode showed that files of that size corresponded to MP3s containing an entire CD of audio, or roughly 50 to 75 minutes of music.

5.3.4 `.wav` Concentration

`Wav` files (Figure 7) have only a single mode, with 48% of `.wav` files in the 16 kB to 64 kB range ($\log_2 = 14 - 16$). This makes sense since `.wav`s are mainly used for brief sound effects and alerts. The sound files for Pinball and Windows Themes (both automatically installed with Windows), as well as those for the commonly used Microsoft Office and AOL Instant Messenger programs, fall into that range.

5.3.5 Video Extensions

As mentioned earlier, we filtered for media files by looking at the extension, which was the only file-specific information to which we had access. However, we ran into a slight problem here because sometimes the extension misrepresents the true file contents.

An example is in the very small video sizes. For both `.avi` and MPEG formats, a number of files in the 512-1024 byte ($\log_2 = 9 - 10$) range clearly influence the shape of the curve (see Figures 8 and 9), but do not contain the type of information associated with the extension. Instead, they are the result of cross-architecture file movement. In this case, the small "video" files are actually Macintosh resource forks associated with corresponding (large) `.avi` and `.mpeg` files, created when a Macintosh archive is unpacked on a Windows or Linux machine.

5.3.6 `.avi` Modes and Spikes

The `.avi` format is used for 3 main types of video, producing the spikes visible in Figure 8 that are not the result of extension confusion. Because of our anonymization, we were unable to characterize the small modes at around 8K-32K bytes ($\log_2 = 13 - 15$). However, we were able to identify the cause of peaks at the larger sizes because they were present in the data from computers where the users agreed to give us full information about their video files. Excerpts from TV shows that are about a minute in length, commercials, 3D animations, and sound card demo files tend to range from 4 to 8 MB ($\log_2 = 22 - 23$). The peak at about 128 MB ($\log_2 = 27$) contains half-hour-long TV animations, as well as roughly 45-minute-long live-action TV episodes such as Star Trek. The final significant spike, at about 2^{29}, primarily contains files of 650 to 850 MB, which represent movies of lengths from around an hour to around $2\frac{1}{2}$ hours. At present, we are unable to explain why the wide range of movie running times is not reflected by an equally wide range in file sizes.

5.3.7 MPEG Modes

Figure 9 shows the size distribution for MPEG videos, including 3 main areas of interest in addition to the one resulting from extension confusion. In the 1 to 2 MB ($\log_2 = 20 - 21$) mode, there are video clips that are approximately 10 to 30 seconds in length. The next mode, at 32 to 64 MB ($\log_2 = 25 - 26$), contains music videos and movie previews. The slight irregularity in the downslope at 128 to 256 MB ($\log_2 = 27 - 28$) is caused by half-hour-long animations and TV episodes.

6 DISCUSSION

6.1 Curve Shapes

The most striking aspect of our data, in contrast to previous studies, is the peakiness of the size distributions. In most cases there is far more data near the mode than can be explained by a (log)normal distribution. Although the overly high modes clearly indicate a subtle bias in favor of certain file sizes, we currently have no explanation for that bias.

It is interesting to note that the major exception to the high-kurtosis distributions is found on Windows-only computers, where the distribution is very close to a Gaussian. Had Douceur and Bolosky had access to a wider variety of operating systems, it is likely that they would have reached the same conclusion as we did.

6.2 "Anomalous" Files

As in Douceur and Bolosky's study, a common characteristic of the graphs we have presented is that the fitted curves do not model the observed data well, even though we used a large bandwidth to smooth the data as much as possible. For example, in Figure 4 the mode in the observed data is wider and flatter than that in the fitted curve, while in Figure 5 there is an irregularity at about 64 bytes that frustrates attempts to get a precise fit from any simple equation.

Graphs of individual machines (most of which are not shown here for space reasons) also show significant variations compared to each other and to the overall graphs. For example, the machine in Figure 10 has a noticeably different distribution than the all-Windows calculation shown in Figure 3. Contrary to the conclusions of prior researchers, we believe that this variation from machine to machine is the normal state. Although one could smooth out the variation by combining data from large numbers of machines, doing so would hide details that we consider to be critical to the development of an accurate model of the file sizes seen on a single computer.

It is these irregularities that lead us to the primary conclusion of our study, namely that it may not be possible to find a simple equation that describes file sizes in a general fashion. For example, the graphs in Figures 2 and 10 both show extreme spikes or modes caused by a cluster of files of similar size.

Similarly, some of the lopsidedness of the main mode of Windows machines (Figure 3) is caused by files with the obscure extension `.sea`. These files, associated with Creative Labs sound cards, are settings for surround sound, multiple speakers, headphones, and special effects, as well as files for doing some sound processing. Although these file are in some sense media-related, they deal with configuration and are thus more closely related to operating system files than to multimedia files. For that reason, we did not include them in the study of media files, nor model them separately in our graphs and curve fits.

Close examination of our graphs will reveal that in every case, file-size distributions are "polluted" by large collections of similarly sized files, such as icons or configuration files associated with a particular application. Although prior researchers have dealt with these anomalies by ignoring them, we have found that they are the norm rather than the exception, and that they must be accounted for in some fashion if we are to fully understand file-size distributions. Furthermore, the exact locations of these peaks will vary from machine to machine, depending on its purpose and on the particular applications installed (and used!).

6.3 Multimedia Files

Although multimedia files come in a wide variety of sizes, we have tentatively found that they do not fit nicely into the same distribution parameters followed by "ordinary" files. As one might expect, multimedia files tend to be significantly larger than other files, as reflected by the median sizes of 3.8 MB for MP3s, 30 kB for `.wav`s, 5.4 MB for MPEGs, and 9.2 MB for `.avi` files.

Audio files, which were more prevalent in our study, tend to have the most tractable distributions (Figures 6 and 7). Even here, however, the curve has a far sharper peak than for "general" files, as reflected by the MP3 kurtosis of 11.3. Thus, one can conclude that audio files fall primarily into very limited size ranges. However, we note that this conclusion should be tempered by the realization that MP3s are currently used almost exclusively to store popular music. Our observations should be revisited in a few years to see whether changes in user behavior have produced a significant change in MP3 sizes.

Video files are much more difficult to characterize at present. Although our sample of 1427 files would seem to be large enough to at least indicate an appropriate distribution for modeling, the observed data is highly variable and inconclusive. The only reliable statements we can make are unsurprising: (a) video files are larger than audio or general-purpose files, and (b) their incidence is increasing.

6.4 Changes in File Sizes

In their 1999 study, Douceur and Bolosky fitted a lognormal distribution whose mean was placed at 4.7 kB. In addition, they reported a geometric mean size of 23 kB for `.wav` files. In our own study, we found a geometric mean file size on Windows systems of 11 kB ($\log_2 = 13.4$), representing a more-than-double increase, and of 31 kB ($\log_2 = 14.9$) for `.wav` files, indicating a 39% increase, in only two years. This is consistent with the observation of Vogels [16, p. 100], who found that file sizes had increased by an order of magnitude since the Sprite studies [10].[2] Although we do not have enough information to fit a predictive curve to the rate of increase, it is clear that file sizes are increasing rapidly.

We believe that the primary cause of this increase is the advent of media files. In support of this conclusion, we note that the geometric mean file size across all of the systems we studied (including Windows) is only 2.6 kB, and that the geometric mean of non-media Windows files is 6.8 kB. Since media files were represented disproportionately on Windows systems (cf. the relative sizes of the secondary modes in Figures 1 and 3), we hypothesize that the presence of media files is at least partly responsible for the larger mean file sizes on Windows systems.

7 IMPLICATIONS OF FINDINGS

Our study has found file-size distributions that are considerably more complex than previously believed. This observation has significant implications in the modeling and design of filesystems.

7.1 Modeling

Previous researchers have attempted to model file-size distributions using relatively simple equations [14] or mixtures of equations [2]. However, we have found that simple mixtures omit important behavior that should be considered in filesystem design (see below).

Rather than being relatively simple distributions or mixtures of distributions, we believe that filesystem models must include complex factors that simulate the rather odd distributions of file sizes found in real computer systems. Douceur and Bolosky [2] have already suggested a mixture of a lognormal distribution for nonzero file sizes with a fixed distribution for zero-sized files. We suggest extending and generalizing their approach in several ways:

1. Rather than a lognormal model, use a more flexible method such as the lambda distribution, so that the significant peakiness shown by some systems can be accurately reproduced.

2. Increase the number of components in the mixture to model the major modes caused by different types of files, notably media files.

3. Add large numbers of extra files in small, randomly selected size ranges to model clustered spikes such as that shown in Figure 2.

The random selections in item 3 are especially challenging, because at present we have no data to suggest where the random spikes should be placed or how they should be distributed. A study of the distribution of spike placement would be a fruitful area for future research.

7.2 Design

Although our results may influence filesystem modeling, the design implications of our observations have the potential to have a much wider effect. Previous filesystem designs have worked on the assumptions that most files are small and that file sizes are randomly distributed. Our results suggest that these assumptions are not valid.

To deal with changes in the distribution of file sizes, we suggest several corresponding design approaches:

1. In contrast to the conventional wisdom that small files are the most common, a good filesystem layout should be able to store large numbers of medium-sized files. Although small and large files are important, they should not exclude efficient storage of the most common sizes.

2. The exception to the above observation is that zero-sized files are extremely common and must also be handled efficiently.

3. Designers should be aware that there are unpredictable peaks in file-size distributions. Filesystems should not degrade in performance simply because of an unusual number of files of a particular size.

4. Large files cannot be ignored, and will continue to become more important as time passes. Thus, filesystems should handle very large files effectively and efficiently.

5. Since multimedia files are primarily accessed sequentially, it may not be necessary to provide for efficient random access to every large file.

6. Mean file sizes are increasing rapidly. Since it is quite possible that the rate of increase will accelerate as multimedia files become more prevalent, filesystems should be designed under the assumption that file sizes will become far larger in the future.

These observations lead to the conclusion that designers should not hardwire static assumptions and configuration parameters into their filesystems. Instead, filesystems should adapt to the data at hand. It is only through dynamic adaptation that a filesystem will be able to efficiently handle unusual spikes, general-purpose files, an ever-growing mix of media files, and overall growth in the average file size.

7.3 Validity of Trace Data

Finally, the rapid change in file sizes that we observed compared to Douceur's work suggests that researchers should be cautious about conducting studies with old trace data. Even the relatively recent Coda [6] traces are now sufficiently outdated that they do not represent an accurate model of modern filesystems. This rapid devolution of trace validity presents

[2] The reader may wish to note that Vogels' bibliography gives an incorrect year for this citation.

a conflict between the need for accuracy and the desire to be able to compare results with those of other researchers.

To address this difficulty, we suggest that researchers use both old and new traces in their investigations. The older, well-established traces will allow new results to be compared with prior work, while the new traces will validate that the results are still applicable to current computer systems.

In addition, it may be desirable to develop a model of the evolution of file characteristics, so that current traces can be projected into the future to predict the behavior of file systems 3-5 years from now.

8 FUTURE WORK

In our small study, we have identified many characteristics of file systems that have not been previously recognized, and have also discovered a number of important effects of media files.

However, there are a number of unanswered questions remaining from our work. Some of the issues that we would like to investigate further include:

- The cause of the unexpectedly small size of source files on our multiuser machines (Section 5.3.1).
- The model underlying the placement and height of unusual peaks and spikes in the size distributions (Section 5.3).
- The reasons for the relative constancy of movie-file sizes despite the wide variation in running times (Section 5.3.6).
- The underlying reasons for the high kurtosis, which indicates a bias toward certain file sizes (Section 6.1). Since the modes for non-media files tend to be around 2K, one suggestion has been that software designers tend to be aware of the system block size and lay out their files to use as much of a disk block as possible, without extending beyond that limit. However, that suggestion has not been verified, and it does not apply to media files, where the kurtosis might be better explained by a bias toward certain song or video lengths.
- The file sizes preferred by popular applications such as Web browsers, mail agents, etc.

Finally, we are currently preparing to make our data sets available for distribution to other researchers. We plan to create a Web site with our raw data in early June of 2002.

9 CONCLUSION

We have studied the distribution of file sizes under Windows, Solaris, and Linux. We found that in general, observed distributions have too high a kurtosis to be described by a lognormal curve. In addition, concentrations of files at popular sizes produce secondary modes that cannot be described by any simple statistical model. These modes may be due to media files or may be caused by particular software that tends to create files of a certain size. Because of these irregularities, we believe that future researchers should use more complex models when generating size distributions, and that filesystem designers should ensure that their systems can handle large concentrations of similarly sized files.

ACKNOWLEDGMENTS

Without the data provided by others we would not have been able to do this study. We would like to thank Chris Marble for providing data from the academic computing Linux machines; Geoff Romer for running the scripts on the Computer Science Department's machines; Eric Harley for providing the MBL data; Professors Henry Krieger, Michael Moody, and Andrew Bernoff of the math department for their help in the statistical analysis; and Harvey Mudd faculty and summer residents for providing data from their personal computers. Finally, we would like to thank Mike Erlinger, Peter Reiher, Andy Wang, and Mark Yarvis for reviewing drafts of the paper and providing valuable suggestions.

A THE LAMBDA DISTRIBUTION

When fitting probability distributions to experimentally collected data, it is often helpful if the distribution can be easily molded to a wide variety of shapes, especially those that are either flatter or more peaked than a normal distribution, and those that are skewed to one side. A number of well-known distributions, such as the beta and Weibull distributions [4], answer this description. The four-parameter lambda distribution [11] is an exceptionally flexible approach to fitting data; it can easily be tuned to generate density curves with almost any desired skewness and kurtosis.

A.1 Defining the Lambda Distribution

Unlike most other distributions, which are defined by their probability density function (pdf) or cumulative distribution function (CDF), the lambda distribution is defined by its percentile function $R(p)$, which is the inverse of the cumulative distribution function (i.e., if $F(x)$ is a CDF, then $F(R(p)) = p$ for all $p \in [0, 1]$).

The percentile function of the lambda distribution is given by Ramberg [11] as:

$$R(p) = \lambda_1 + \frac{p^{\lambda_3} - (1-p)^{\lambda_4}}{\lambda_2}$$

for $0 \leq p \leq 1$.

The shape of the density function is affected by the parameters as follows: λ_1 is a location parameter similar to the mean, λ_2 is a scaling parameter similar to the standard deviation, and λ_3 and λ_4 interact to control the shape of the pdf.

There is no simple relationship between the moments of the observed data and the lambda parameters, but numerical methods can be used to derive values for all four parameters given estimates of the mean, standard deviation, skewness, and kurtosis of the observed data. We used a simple Maple package to generate the parameters used in this paper.

A.2 Generating a Lambda Distribution

The probability density function for a lambda distribution cannot be derived directly as a function of x, but it can be written implicitly as a function of $R(p) = x$:

$$f(R(p)) = \frac{\lambda_2}{\lambda_3 p^{\lambda_3 - 1} + \lambda_4 (1-p)^{\lambda_4 - 1}}$$

again for $0 \leq p \leq 1$. The latter function can be used in association with the acceptance-rejection method [4, p. 478] to generate random numbers with the fitted distribution.

References

[1] Baker, M. G.; J. H. Hartman; M. D. Kupfer; K. W. Sherriff; J. K. Ousterhout, 1991. Measurements of a Distributed File System. In *Proceedings of the Thirteenth Symposium on Operating Systems Principles*, (October). ACM, 198–211.

[2] Douceur, J. W. Bolosky, 1999. A Large-Scale Study of File-System Contents. In *ACM SIGMETRICS Conference Proceedings*, (Atlanta, Georgia, USA, May). ACM.

[3] Downey, A. B., 2001. The Structural Cause of File Size Distributions. In *Proceedings of the Ninth International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, (Cincinnati, OH, August). IEEE.

[4] Law, A. M. W. D. Kelton, 1991. *Simulation Modeling and Analysis*. McGraw-Hill, New York, NY,

[5] McKusick, M.; W. Joy; S. Leffler; R. Fabry, 1984. A Fast File System for UNIX. *ACM Transactions on Computer Systems*, 2, No. 3, (August): 181–197.

[6] Mummert, L. B. M. Satyanarayanan, 1996. Long Term Distributed File Reference Tracing: Implementation and Experience. *Software—Practice and Experience*, 26, No. 6, (June): 705–736. Also available as Carnegie Mellon University Technical Report CMU-CS-94-213.

[7] National Institute of Science and Technology (NIST). *NIST/SEMATECH Engineering Statistics Handbook*. Available at `http://www.itl.nist.gov/div898/handbook/`, last visited 6 July, 2001.

[8] Ousterhout, J. K.; A. R. Cherenson; F. Douglis; M. N. Nelson; B. B. Welch, 1988. The Sprite Network Operating System. *IEEE Computer*, (February): 23–36.

[9] Ousterhout, J. K.; H. D. Costa; D. Harrison; J. A. Kunze; M. Kupfer; J. G. Thompson, 1985. A Trace-Driven Analysis of the Unix 4.2 BSD File System. Technical Report UCB/CSD 85/230, UCB,

[10] Ousterhout, J. K.; H. D. Costa; D. Harrison; J. A. Kunze; M. Kupfer; J. G. Thompson, 1985. A Trace-Driven Analysis of the Unix 4.2 BSD File System. In *Proceedings of the Tenth Symposium on Operating Systems Principles*, (December). ACM, 15–24.

[11] Ramberg, J. S. B. W. Schmeiser, 1974. An Approximate Method for Generating Symmetric Random Variables. *Communications of the ACM*, 17: 78–82.

[12] Roselli, D.; J. R. Lorch; T. E. Anderson, 2000. A Comparison of File System Workloads. In *USENIX Conference Proceedings*, (San Diego, CA, June). USENIX.

[13] Rosenblum, M. J. K. Ousterhout, 1992. The Design and Implementation of a Log-Structured File System. *ACM Transactions on Computer Systems*, 10, No. 1, (February).

[14] Satyanarayanan, M., 1981. A Study of File Sizes and Functional Lifetimes. In *Proceedings of the Eighth Symposium on Operating Systems Principles*, (December).

[15] Silverman, B. W., 1986. *Density Estimation for Statistics and Data Analysis*. Monographs on Statistics and Applied Probability. Chapman and Hall, London,

[16] Vogels, W., 1999. File System Usage in Windows NT 4.0. In *Proceedings of the 17th Symposium on Operating Systems Principles*, (Kiawah Island, SC, USA, December). ACM.

Kylie M. Evans is an undergraduate student in the joint Computer Science/Mathematics major at Harvey Mudd College. The research described in this paper was performed during the summer following her sophomore year.

Geoffrey H. Kuenning is Assistant Professor of Computer Science at Harvey Mudd College. He received his Ph.D. in computer science from UCLA in 1997, and his B.S. and M.S. in computer science from Michigan State University in 1973 and 1974. From 1974 to 1989, he worked in the areas of operating systems and embedded systems. His research interests include file systems, performance analysis, and computer systems security.