Availability and Locality Measurements of Peer-to-Peer File Systems

Jacky Chu Kevin Labonte Brian Neil Levine
Department of Computer Science
University of Massachusetts, Amherst, MA 01003
{cchu,klabonte,brian}@cs.umass.edu

ABSTRACT

Although peer-to-peer networking applications continue to increase in popularity, there have been few measurement studies of their performance. We present the first study of the locality of files stored and transferred among peers in Napster and Gnutella over month-long periods. Our analysis indicates that the locality of files is skewed in all four cases and fits well to a log-quadratic distribution. This predicts that caches of the most popular songs would increase performance of the system. We also took baseline measurements of file types and sizes for comparison over time with future studies. Not surprisingly, audio files are most popular, however a significant fraction of stored data is occupied by videos. Finally, we measured the distribution of time peers in Gnutella were available for downloading. We found that node availability is strongly influenced by time-of-day effects, and that most user's tend to be available for only very short contiguous lengths of time.

1. INTRODUCTION

Traffic from Web and peer-to-peer applications rank among the most dominant on the Internet. However, compared to studies of the web, there are many fewer measurement studies of peer-to-peer (P2P) file sharing networks. In this paper, we present the preliminary results of two studies on P2P applications. For the first, we periodically recorded the names of files stored by several thousand users of the Napster application in January 2001 and Gnutella P2P file system in March 2002. For the second, we periodically measured the availability of peers on the Gnutella network in March 2002.

Our analysis of these measurements shows strong evidence that caches can improve the performance of these systems as seen by the user and reduce the use of network resources. There are many factors that suggest this is true.

This paper was supported in part by National Science Foundation awards ANI-033055 and EIA-0080199, and by a gift from Sprint Advanced Technology Labs.

First, we calculated the locality of Napster and Gnutella stored and transferred files, recorded more than a year apart. We found that both networks exhibit locality characteristics that fit a log-quadratic distribution (and are closely approximated by two power laws). The most popular 10% of files in Gnutella account for 50% of the total number of stored files. The locality of transferred files is even higher: the most-popular 10% of transferred files account for over 60% of total transfers by our estimates. We are unaware of other published work showing results for stored or transferred files, or results for the effectiveness of caching file downloads; the most closely related work has focused on the locality and caching of queries.

Second, we present the distribution of file types and sizes that we found on the network. Not surprisingly, most files are MP3 encodings of about 4Mb in size. Only about 3% of all files are videos, but they account for 21% of all stored bytes. These measurements serve as a baseline for changes we expect to track and observe over the coming years.

Third, we present our analysis of node *availability*, which we define as the amount of time peers are available to serve file transfer requests for others. Our results show that a majority of nodes who are connected to the network are actually busy when checked, meaning they are not able to handle requests from other peers. In Gnutella, nodes may at times answer queries but not serve files. For example, nodes may not be available when a user-set limit on concurrent downloads is reached. Our findings show that availability oscillates over a 24-hour period, indicating that the time of day affects when nodes are busy or offline. We found that the distribution of the length of continuous times nodes are available is heavily skewed to short times. This skewed performance can be approximated by a log-quadratic distribution.

In sum, we found that transferred files have high locality, the most popular shared files are about 4 MB, and nodes tend to be unavailable. These results suggest that caches of popular P2P shared content, perhaps collocated with web caches, would be a significant improvement for users of these systems. For example, in our study, the most popular 5% of files accounted for 50% of all transfers. This corresponds to about 45,000 songs, which can be stored in about 175 GB.

This paper is organized as follows. Section 2 reviews previous work and background material on the operation of Napster and Gnutella. Section 4 describes our method of measurement and data collection. Section 5 presents our results on locality, node availability, and other characterizations. Section 6 offers our conclusions.

2. BACKGROUND

2.1. Previous Work

Although similar studies on P2P applications have been reported, our contributions can easily be distinguished from these earlier works. We are not aware of previous work that has any measurements on stored or transferred files, nor on time-of-day effects present in node availability. In addition, where we have repeated analysis, we discuss how our results differ from previous work.

Ripeanu et al.[1] mapped out the Gnutella topology by monitoring Ping/Pong messages, measured percentages of traffic by type (e.g., queries and pings), and found that the Gnutella topology does not match the underlying network topology.

Markatos[2] took measurements from three Gnutella clients at separate geographic locations for one hour and analyzed the effects of caching search queries. Due to the high temporal locality of queries observed, a simple query caching scheme was shown to reduce query traffic by as much as a factor of two. This traffic does not include the traffic caused by file transfers between peers, which is what we propose to reduce.

Saroiu et al.[3] did a study on latency and bandwidth in the Gnutella network. They studied the availability of 17,125 nodes over a 60-hour period, probing each node every seven minutes. We also studied node availability, but we ran our experiment over a six-week-long period, greatly extending their results.

Adar and Huberman[4] measured Gnutella Query and Ping/Pong messages. Pong messages contain the number of files shared by users. They inferred peer downloads from QueryHit messages by assuming that users download all files that appear in all such messages, clearly an incorrect and poor assumption. In our experiment, we explicitly obtained the shared file list periodically from each user by taking advantage of the capabilities of specific Gnutella implementations. We calculated the differences over time of each user's shared file lists to infer which songs were transferred.

We know of only a single paper that has proposed analytical models of P2P networks, by Figueiredo et al..[5] This preliminary work must make assumptions about some of the performance attributes we examined, including session length and node availability. We expect our results to be useful for such modeling studies.

The remainder of this section briefly overviews the aspects of the Napster and Gnutella protocols relevant to our study. For detailed protocol information, please see the Napster protocol specification[6] and the Gnutella v0.4 protocol specification.[7]

2.2. Napster Protocol

The Napster protocol uses a centralized approach to keep track of which files are stored by each of its peers. When a peer comes online, it sends its list of shared files to the central server. When a peer goes offline, its list of shared files is removed from the central database. All search requests are sent directly to the central server, which in turn searches its database for clients who are sharing files that satisfy the request. The list of results is sent back to the client, and the file transfers occur directly between two peers. We made use of the protocol's *browse* message in our experiment, which allows a client to get a specific user's song list from the Napster server.

2.3. Gnutella Protocol

The Gnutella protocol differs from the Napster protocol in that it uses a distributed approach to locating files in the network. Since there is no central server to connect to, a peer must maintain connections to a set of other known Gnutella peers, called *friends*. Search requests are carried out by sending the query to each friend, who in turn relay the query to their friends, and so on until the query has flooded the network up to a certain depth. Search results are routed through peers along the reverse path of the query until they reach the originating peer. Like in Napster, file transfers occur directly between two peers.

3. EXPERIMENT METHODOLOGY

This section describes the methodology of two measurement experiments we performed. In the first experiment, we recorded the files downloaded by Napster and Gnutella users. In the second experiment, we measured the amount of time nodes were available in the

Set	Dates	Users seen	Total Stored files	Unique Stored files	Total Transfers Recorded	Unique Transfers Recorded
0	12/21 – 12/25	20,969	11,808,017	3,029,731	24,557	21,933
1	12/26 – 12/30	6,375	3,064,179	1,244,044	14,526	13,790
2	12/31 – 1/4	10,443	6,052,391	2,033,928	62,741	53,046
3	1/5 – 1/9	1,896	992,858	507,392	9,653	9,049
4	1/10 – 1/14	-	-	-	-	-
5	1/15 – 1/19	37,737	6,649,273	2,115,010	296,516	215,452
6	1/20 – 1/24	45,541	19,029,580	4,107,157	399,067	261,544
7	1/25 – 1/29	33,767	15,322,370	3,520,122	454,300	282,954
8	1/30 – 2/3	22,992	6,340,511	1,636,540	342,466	222,125

Table 1. Song files and transfers recorded from Napster clients.

Set	Dates	Users seen	Total Stored files	Unique Stored files	Total Transfers Recorded	Unique Transfers Recorded
0	2/24 – 2/28	5,414	9,139,684	984,576	1,573,592	507,493
1	3/1 – 3/5	4,692	8,746,235	908,922	1,108,673	460,974
2	3/6 – 3/10	4,397	10,020,038	918,520	1,184,953	435,874
3	3/11 – 3/15	3,885	9,557,035	877,600	1,066,794	419,411
4	3/16 – 3/20	3,559	10,799,077	874,036	5,474,660	682,291
5	3/21 – 3/25	3,108	9,582,559	708,496	5,560,798	562,590

Table 2. Song files and transfers recorded from Gnutella clients.

Gnutella network. The large amount of data we gathered was analyzed to produce the results described in Section 5.

3.1. File List Collection

In order to record the file lists of users from the two P2P networks, we first discovered a large set of users on the network. We then periodically probed each user for their list of files, if the node was available at the time. Each file list was stored with an associated timestamp and user identification. Specifics as to how this was done with each of the two networks are described briefly in the following subsections.

4. EXPERIMENT METHODOLOGY

This paper describes the analysis of two measurement experiments. In the first experiment, we recorded the files downloaded by Napster and Gnutella users. In the second experiment, we measured the amount of time nodes were available in the Gnutella network. This section describes the methodology of both experiments.

4.1. File List Collection

In order to record the file lists of users from the two P2P networks, we first had to discover a set of users on the network. Then, we would periodically probe each user for their list of files if the node was available at the time. Each file list was stored with an associated timestamp and user identification. Specifics as to how this was done with each of the two networks are described briefly in the following subsections.

The large amount of data gathered from this part of the experiment was analyzed to produce the results described in Section 5.

Replicas of a file in a P2P system are usually renamed by users according to their preference. Therefore, to map those replicas with different filenames to an original file, we shortened names to signatures. Our shortening process generated a signature from a given file name as follows:

1. Drop stop words, e.g., “and” and “the”.
2. Take out immediately repeated letters (e.g., “collins” becomes “colins”).
3. Drop vowels.
4. Convert any non-alphanumeric character into a space.
5. Condense and drop leading white space.
6. Sort the space-delimited name to obtain a signature.

We considered the set of files with same signature a replica set of a file.

4.1.1. Napster Collection Details

Our Napster measurements took place from December 21, 2000 until February 3, 2001 (including a four-day break). During this time, we recorded the files of thousands of users and hundreds of thousands of transfers. Table 1 shows more details.

Note that these dates are prior to legal rulings that forced Napster to filter out copyrighted content and resulted in users altering filenames artificially. While this initial experiment was not executed as well as our subsequent measurements of Gnutella, we analyze the data because it is no longer possible to collect such data from Napster. However, we were pleased to see that many of the results match well with data collected a year later from the Gnutella network.

We connected to the Napster server from a client that we wrote based on the Napster protocol specification.[6] Our client continuously submitted search requests using random words picked from an English dictionary file. When the results returned from the server, we were able to determine a large list of users available on the network. As we discovered user IDs for the first time, we added these names to a database maintained throughout the experiment. Another client cycled through the database's list of users and sent a *browse* message to the server to retrieve each user's file list, which would succeed only if that user was online at the time. The file lists were stored with an associated timestamp for later analysis, presented in section 5.

4.1.2. Gnutella Collection Details

We collected file lists from several thousand Gnutella users from February 24, 2002 until March 25, 2002. Specific details are shown in Table 2. We modified the JTella API[8] to create a custom measurement program. The program created a list of available nodes by connecting to an "always-up" node (e.g., router.limewire.com). This node is actually resolved to a random node that happens to be online at the moment. Our client learns about other Gnutella peers through Ping/Pong messages and eventually connects to a fixed number of other nodes as *friends.*

Our program examined QueryHit messages as they were routed from neighbors. These messages contain identifying information about other nodes on the network, such as their GUID, IP address, and listening port. We uniquely identified nodes by their IP-port pair*. As each IP-port pair was examined, known unroutable IPs[9] were discarded from our list, since these would be impossible to contact directly for file transfers. We could have used the Gnutella push protocol to initiate transfers, however, this would have taken too much time given the number of users we monitored.

Since only some Gnutella clients allow their users' file lists to be retrieved, we focused our experiment on two of the most popular clients that allow this, Bearshare and SwapNut. By sending an HTTP request directly to these clients on their listening Gnutella port, we obtain an HTML page listing every file the peer is sharing. This HTML document is parsed and stored in a database that associates the user with each file listed and its respective file size, along with a timestamp indicating when this file list was obtained.

Once we obtained a list of 20,000 known peers, we periodically collected information about the files they were sharing. From this data, we could infer over time which files they have downloaded.

In our experiment, we cycled through the static list of known peers, trying to obtain the file list from each one of them. Each cycle where we attempted to contact each peer took approximately three to four hours due to the size of our peer list and the hardware we used.

4.2. Node Availability

To study node availability, we gathered a fixed list of nodes by tracking Gnutella network traffic. We extended the JTella API[8] to create a custom Gnutella client that extracted IP address and port number information from all QueryHit messages that were routed through our client as part of normal Gnutella operation. In our experiment, we collected observed Query and QueryHit messages. Query messages can reflect the popularity of search words, though we do not present that analysis here.

Once we collected a node's IP address, we no longer needed the Gnutella or HTTP protocols. We simply attempted to contact a node by opening a TCP connection. To quickly initiate and close a TCP connection, we created a *tracking manager* which used nmap, a customizable UNIX administrative tool used for port scanning. We set the maximum timeout and the RTT value to five seconds, a value that is small enough to cycle through the list in a relatively short time period but large enough to allow sufficient time for TCP connection set up to occur if the node is online. Our tracking

*Using GUIDs to identify nodes might seem more appropriate, but the Gnutella protocol does not allow messages to be routed to nodes solely based on the GUID of the destination. Also, the GUID in practice is not guaranteed to be globally unique, nor is it guaranteed to remain fixed for subsequent sessions of the same client.

manager cycled through a list of node IP addresses and port numbers. For each *cycle*, nmap determined which of three states each node was in:

- *Up* - A node accepts our incoming TCP connection, meaning the node is *available*.
- *Closed* - A node is responding to our probe, but its listening port is not accepting TCP connections, meaning the node is not currently connected to Gnutella; thus, the node is not available.
- *Down* - We are not able to create a route to a node, meaning the client is either too busy to handle more requests or the node is disconnected from the Internet; thus, the node is not available.

We considered the *Down* state as not available because even if a node is running a Gnutella client but does not accept any more connections, it cannot respond to messages from other nodes, meaning it is unavailable to other nodes wishing to download files.

Using a single process to cycle through a large node list would not allow us to track each node frequently enough. Therefore, to effectively track a large node set with a relatively small time interval, we use a script to spawn a new tracking manager every 10 minutes that cycles through the node list once. Thus, each node was tracked approximately every 10 minutes, but with some slight inconsistency depending on network delays.

Moreover, immediately after we discovered a new node during the node collection process, that particular node was probed immediately so that the tracking delay that results from the node collection process was eliminated, which provides us with some base data to compare the rest of the experiment with.

We conducted our experiment from March 28th until May 5th without interruption and collected 5,000 cycles of data for 5,000 nodes.

5. ANALYSIS

5.1. File Transfer Locality

Several previous works have analyzed the locality of accesses to web proxies and servers, commonly fitting access patterns to skewed distributions (e.g., power laws[10]). These skewed distributions are easily taken advantage of by web caches for improved performance. We also found a heavily skewed file popularity for Gnutella and Napster. Our analysis indicates that caches for a group of users (e.g., collocated on a university campus) should be an effective method of increasing the performance of P2P applications.

One would expect that applications like Gnutella would be "self-caching", in that as a file becomes more popular, more nodes will store it. One would expect that the widespread appearance of a file is likely to improve the average transfer time of users downloading the file. There are several reasons this may not be true.

The application-level TTL fields limit the scope of a broadcast query in Gnutella to peers closer in the topology. However, the application-level topology of existing P2P applications does not follow the underlying IP network topology.[1] For this reason, adjacent nodes are not necessarily close; less can be expected from a neighbor's neighbor, and so on. Second, Gnutella clients have no sophisticated method of directing users to the best of several peers all discovered to be sharing the same file. Server selection methods have been extensively researched (Hanna et al.[11] provides a good overview of such research), but applying these results to peer-to-peer networks is not trivial. Simple pings to differentiate peers is a poor measure,[11] and the low occurrence of peer availability (as we show in the next section) does not make it worthwhile to perform network tests on each peer that are more costly (e.g., hop counts via traceroute). Tracking the past history of each node also has no value if nodes are never seen or used again.

By maintaining a cache of popular songs for users on a common intranet, the problem of peer selection is removed: popular songs can be downloaded quickly from the cache without testing or doubt. Furthermore, the benefit of widely shared files is hampered by the low availability of nodes. We expect caches to be always available to all local users and contain the most popular songs for quick download.

To determine the effectiveness of caching on P2P file systems, we first calculated the locality of transferred files. That analysis and others that we present in this section led us to a number of conclusions. In sum,

- Stored file popularity is skewed and follows a log-quadratic distribution. For stored files, the highest-ranked 10% of files accounted for about 50% of the total number of stored files.
- File transfers have a slightly higher locality of reference and also follow a log-quadratic distribution closely. The highest-ranked 10% of files accounted for about 60% of the total number of transferred files.

5.1.1. File Locality

Figure 1 shows the cumulative percentage of stored data as a function of files ranked by their popularity. The graph shows that for our Napster and Gnutella experiments, the most popular 10% of files account for about 50% of all stored data. Files we found stored on the Napster network demonstrate similar locality. The fact that the measurements were recorded almost a year apart on two different types of P2P applications gave us confidence that the curves are perhaps more inherent to music interests of users than to the characteristics of the applications.

Figure 1 shows the CDF of Napster and Gnutella stored file locality. The PDF of file locality of each network is shown in Figure 2. We used Matlab's least-squares curve-fitting tools to find best fits. This distribution does not easily fit a Zipf's distribution, as has been observed for other caching systems. A slightly curving quadratic better fits the most and least popular files. In future work, we hope to model and explain this observed process more carefully. We are not sure what factor has introduced scale into the distribution. The log-quadratic distribution shown, and the others we show in the paper, are easily approximated by two Zipf's distributions, which may be one clue.

Regardless of a proper fit, the skewed distribution of data clearly predicts that caching would be an effective method of reducing the cost of P2P file transfers as well as improving their download latency from remote peers.

Our measurements also allowed us to infer an estimate of file transfers that occurred during that week. Figures 3 and 4 show that transferred files exhibit higher locality than stored files. Again, the distribution exhibits scale and is best fit with a log-quadratic function. A Zipf's distribution is shown for comparison. We believe the lower locality we observed for Napster is due mostly to how infrequently we contacted users.

We determined transfers by taking differences from week to week in files that we recorded as appearing in a user's library. During each week, we determined the most popular stored songs. We weren't always able to contact a user to determine their current file set. Table 2 shows that we contacted about 5,000 to 3,000 Gnutella users of the 20,000 we monitored within each week. Our Napster experiment was less robust: our list of the users we were observing was so long, we probed users too infrequently during each week. However, as this data cannot be collected again due to Napster's demise in popularity and legal troubles, we still present the results.

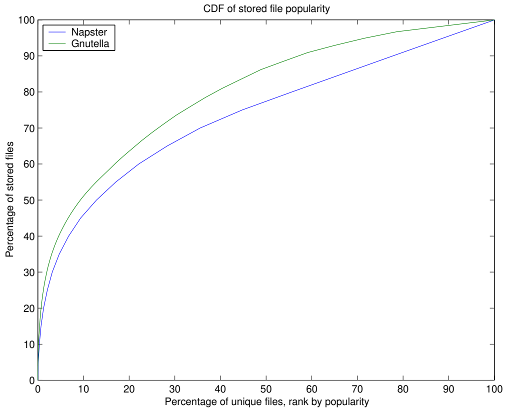

Figure 1. CDF of stored file locality in Napster and Gnutella.

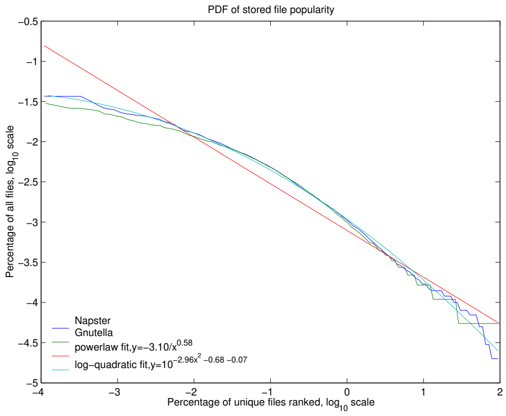

Figure 2. Popularity distribution of stored file locality in Napster and Gnutella.

5.1.2. File Type Demographics

At the time of our data collection from Napster, only files with an "MP3" extension were allowed to be shared by users. However, Gnutella places no such restriction on shared content. Table 3 shows the demographics of the file types we recorded from Gnutella. Audio files and image files are the most popular. When the size of the files is taken into account, audio files and video files make up the bulk of shared data. The average MP3 file was 4.2 MB. We plan to monitor this distribution over the coming years to analyze changes.

In the appendix, we include lists of the most popular shared files stored in the Gnutella network during our collection period. In three separate tables, we show the top 50 files of all types, of just audio types, and of just video types. The ranking is actually by signatures. Numbers in parentheses show the number of users who stored each file's corresponding signature; only one full

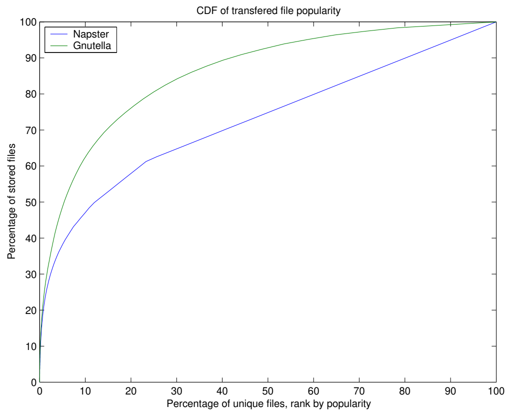

Figure 3. CDF of transferred file locality in Napster and Gnutella.

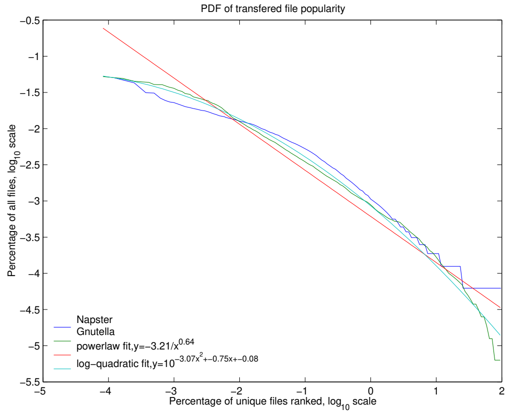

Figure 4. Popularity distribution of transferred file locality in Napster and Gnutella.

type	% of files	% of bytes
audio	75.5%	77.5%
image	14.2	0.0
html	3.8	0.0
video	2.7	21.4
text	1.8	0.1
exe	1.0	0.7
archive	0.1	0.2
others	0.9	0.1

Table 3. Demographics of stored data in Gnutella.

name of each signature is shown. For example, "beck" is really all files who signature is "bck".

5.2. Node Availability

Studies are beginning to appear that propose analytical models of the performance characteristics of P2P applications. Two important measures that may be assumed is the length of sessions (on-line time) and the amount of time away from the network (off-line time). For example, Figueiredo et al.[5] assume that off-line time is exponentially distributed with some mean. The same study assumes that nodes stay on for as long as the number of files they wish to download, plus some "think" time also assumed to be exponentially distributed. We were able to characterize these measures, or measures related to them, from our experiment.

In all Gnutella applications, nodes can limit the number of peers that can download from them concurrently. We distinguish when nodes are *available* as servers of files. Nodes may be unavailable because the application is not running, or if a user-set limit on the number of concurrent downloads has been reached.

The results we present in this section can be summarized as follows:

- The number of nodes available in the networks fluctuates and is strongly affected by time of day.
- Exactly which nodes are available constantly changes; a small percentage of nodes are available for downloads at any instant.
- 31% of the time, nodes were available for only about a 10-minute period before becoming unavailable again. Approximately 20% of the sessions are available for at least two hours. The distribution of session lengths follows a log-quadratic function.

Figure 5 shows the percentage of peers who were available (i.e., the port was open) after a period of time since the peer was first discovered. Several lines appear on the graph. The lowest line is the number of nodes available at specific cycles of the experiment. However, just above is the percentage of nodes that were available at least once during that cycle or cycles that occurred in the previous one hour period before the time on the x-axis (about 5 other cycles). Other lines show the percentage of nodes seen during larger time ranges, including if nodes were seen once or more during the entire experiment. Some nodes are never seen again. This may be because they get a new IP address during a DHCP reconfiguration, or because they use Gnutella more infrequently than once every five weeks.

We observed significant time-of-day effects in the experiment. Figure 6 illustrates this point more clearly than the previous graphs. It shows the average number of nodes available per hour of day (E.S.T., local

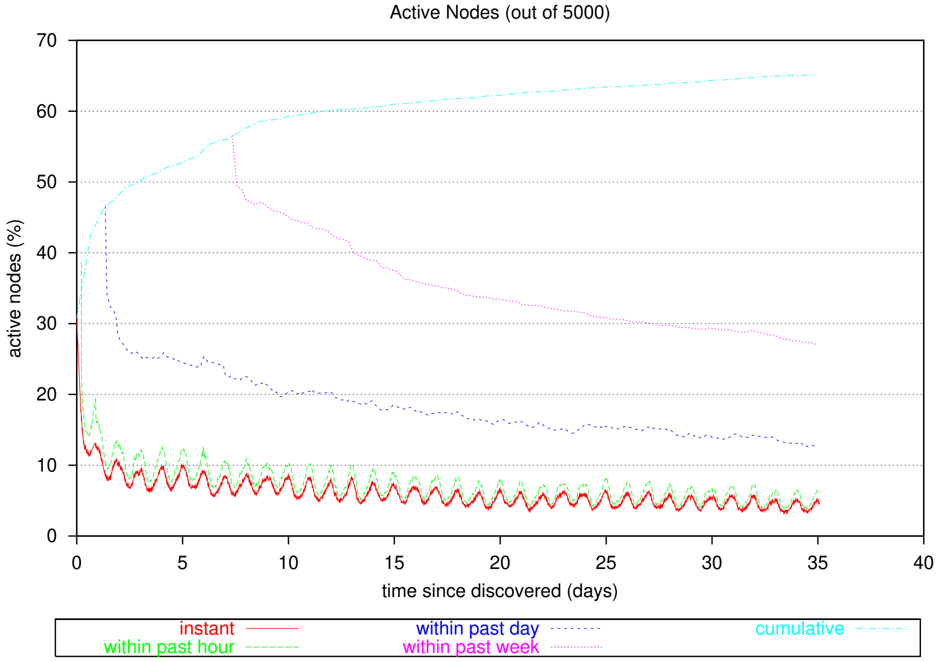

Figure 5. Node availability since the nodes were first discovered (Mar 28, 2002).

to the experiment). We were surprised to see this result. We expected the peers we discovered to be geographically and topologically dispersed. The logical topology created by Gnutella should have no correspondence to geography. We are not aware of any restrictions on network or geographic location for joining the Gnutella network. Messages sent on Gnutella do have a 10-application-layer-hop limit, which is enough to get to other continents. Therefore, geographical location should be irrelevant to proximity. However, this is not what we observed. One explanation is that the majority of users of Gnutella may be in the U.S. Unfortunately, we had no accurate means of determining geographical location of individual clients.

We also analyzed the distribution of the lengths of continuous time that peers were available, shown in Figure 7. The figure also shows a comparison to a log-quadratic distribution. The implications are that most nodes are available for only a short time. About 31% of the sessions have a length of 10 minutes. We were unable to further define this 31% of our observations since we probed nodes every ten minutes.

The log-quadratic could be approximated well by two Zipf's distributions, and that would mean there are two different behaviors regarding a node's session length. Our speculation on the two Zipf's distribution

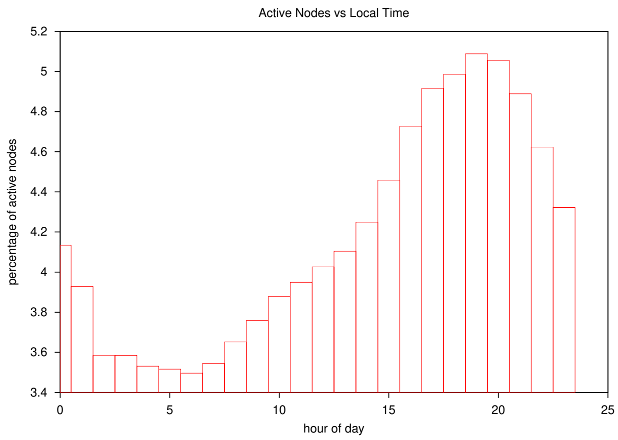

Figure 6. Node availability as a function of the hour of the day (local time).

is that there are two types of Internet connection users: first, users with low-bandwidth, high latency and unstable connections who are guarded about the number of concurrent downloads they allow; and second, users with more resourceful connections who do not limit the number of concurrent downloads as strictly. It may be that users with larger numbers of shared files exhibit proportionately shorter availability times; this multi-

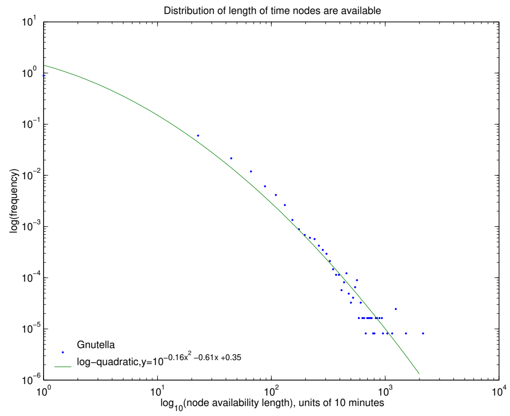

Figure 7. Node availability session length compared to a log-quadratic.

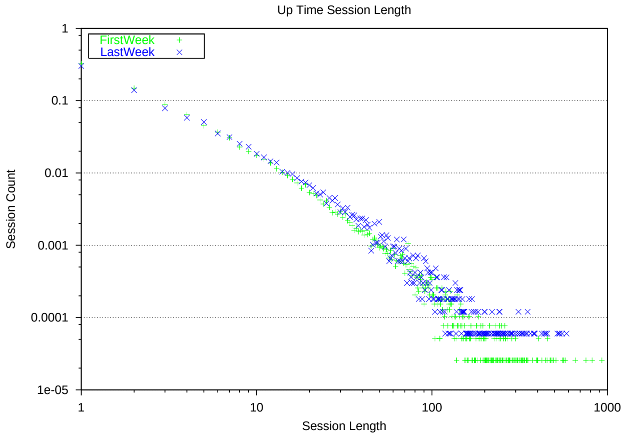

Figure 8. Node availability session length for first and last week.

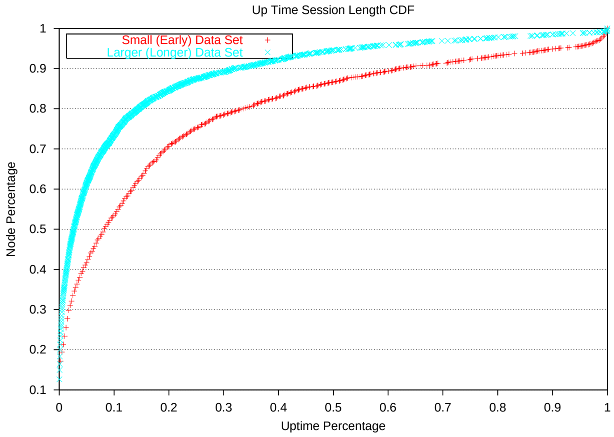

Figure 9. Node availability CDF for a short period of time vs. long period of time.

plicative factor and two user types may be the cause of the two power laws.

We were concerned that a five-week period was too long to average and bin results. However, we found that the session length distribution is rather consistent over the five-week period. Figure 8 shows the session length distribution using just data collected in the first week of the experiment and the session length distribution using data from just the last week of the five-week period. The two distributions are almost identical.

We wished to compare our measurements with Saroiu[3] et al.'s study on lifetime measurement of nodes. We produced the same availability CDF distribution they have shown with a subset of our data that was collected in the first seven days our experiment. (Sariou's experiment lasted 60 hours.) The lower line shown in Figure 9 is the CDF distribution, which is very similar to the results shown by Saroiu. The upper line is the same performance analysis computed without the full set of data. The CDF of the larger data set is left-shifted significantly, meaning that the number of sessions with shorter duration dominates more acutely than previously reported. This further supports the conclusion that peers in the Gnutella network tend to have very short availability times.

6. CONCLUSIONS

We have shown that significant amounts of locality exist in both the stored and transferred files on Napster and Gnutella. These measurements are closely approximated by a log-quadratic (or double power law) distribution. The demographics of stored data in Gnutella show that audio files represent the bulk of shared files. While video files only account for about 3% of the files in the network, their larger size means that they still occupy approximately 20% of the total bytes shared. We expect this will change in the future. We also measured node availability in the Gnutella network and found that availability is influenced strongly by the time of day. Finally, we observed that the length of time nodes remain available continuously also fits well to a log-quadratic curve: nodes tend to be available for only short lengths of time.

These two main results — the skewed distribution of file popularity and low peer availability — suggest strongly that caching the most popular files on the system would greatly improve system performance. We imagine P2P caches could be deployed as web caches are deployed today. Such caches would improve the download times of users by removing the need to guess which peer out of many offers the best download speed, if any are good. Automated peer-selection methods have not been deployed or widely researched. Secondly, a cache would mitigate the affects of the low node availability that we have observed. The asynchronous online times and low availability of users dampers the gains of widely shared popular files.

ACKNOWLEDGMENTS

This work benefited greatly from the efforts of Vincent Scarlata (Georgia Tech) and Ryan O'Boyle (Fidelity) while they were undergraduates at UMass. Vincent and Ryan were part of our team when we collected the Napster data presented in this paper. We also thank Ariyeh Maller (UMass Amherst) for stimulating conversations, and Kathryn McKinley (UT Austin) for her early encouragement.

REFERENCES

1. M. Ripeanu, I. Foster, and A. Iamnitchi, "Mapping the Gnutella network: Properties of large-scale peer-to-peer systems and implications for system design," *IEEE Internet Computing Journal* **6**(1), 2002.
2. E. P. Markatos, "Tracing a large-scale peer to peer system: an hour in the life of Gnutella," in *in Proc. CCGrid 2002: the second IEEE International Symposium on Cluster Computing and the Grid*, May 2002.
3. S. Saroiu, P. K. Gummadi, and S. D. Gribble, "A measurement study of peer-to-peer file sharing systems," in *Multimedia Computing and Networking (MMCN)*, January 2002.
4. E. Adar and B. Huberman, "Free riding on Gnutella," *First Monday* **5**, October 2000.
5. D. R. Figueiredo, S. Jaiswal, Z. Ge, D. Towsley, and J. Kurose, "Modeling peer-peer file sharing systems." UMass Technical Report, January 2002.
6. "Napster protocol open specification," April 2000. Available at http://opennap.sourceforge.net/napster.txt.
7. "The Gnutella protocol specification v0.4." Available at http://www.clip2.com/GnutellaProtocol04.pdf.
8. K. McCrary, "JTella API v0.7." Available at http://www.kenmccrary.com/jtella/.
9. Y. Rekhter *et al.*, "Address allocation for private internets." IETF RFC 1918, Feb. 1996. Available at http://www.ietf.org/rfc/rfc1918.txt.

Rank	Filename	Ext
0 (2171)	divider	GIF
1 (2168)	cm	SMI
2 (2168)	upsell	GIF
3 (1893)	in the end - linkin park	MP3
4 (1685)	Shakira-whenever,whereever	MP3
5 (1549)	A-Nickleback How You Remind Me	MP3
6 (1505)	creed - -my sacrafice	MP3
7 (1482)	Alein ant farm - Smooth Criminal	MP3
8 (1471)	hey baby - no doubt	MP3
9 (1362)	It's Been Awhile - Staind	MP3
10 (1331)	-jay-z - h to the izo	MP3
11 (1307)	RA	MP3
12 (1299)	Angel Shaggy	MP3
13 (1297)	readme	TXT
14 (1296)	pod - alive	MP3
15 (1252)	alecia keys - fallin'	MP3
16 (1250)	usher - you got it bad	MP3
17 (1245)	Shaggy - It wasent me	MP3
18 (1230)	Drops of Jupiter - Train	MP3
19 (1228)	The Calling - Wherever You Will Go	MP3
20 (1228)	(A)R. Kelly -The Worlds Greatest	MP3
21 (1211)	Crawling - Linkin Park	MP3
22 (1198)	Creedd - With Arms Wide Openn	MP3
23 (1191)	Blige, Mary J - Family Affair	MP3
24 (1166)	Incubis - I Wish You Were Here	MP3
25 (1161)	nickelback-this is how you remind me	MP3
26 (1152)	more_full_coverage	GIF
27 (1152)	topnews	GIF
28 (1145)	Setup	EXE
29 (1139)	usher - you remind me	MP3
30 (1126)	outkast - the whole world	MP3
31 (1124)	Get The Party Started - Pink	MP3
32 (1102)	Country Grammer - Nelly	MP3
33 (1098)	blurry - puddle of mud	MP3
34 (1071)	drive - Incubis	MP3
35 (1069)	craig david - feel me in	MP3
36 (1063)	control - Puddle Of Mud	MP3
37 (1053)	Cread - Higher	MP3
38 (1042)	get this party started - pink	MP3
39 (1028)	a-Lifehouse - Hanging by a Moment	MP3
40 (1027)	five for fighting - superman	MP3
41 (1023)	enrique eglasias - hero	MP3
42 (1020)	Eagels - Hotel California	MP3
43 (1016)	All or Nothing - O-Town	MP3
44 (1012)	lead zepplin - Stairway to Heaven	MP3
45 (994)	Last Resort-Poppa Roach	MP3
46 (992)	Nelley-EI	MP3
47 (978)	Ludacris - Role out	MP3
48 (974)	chop suey - System of a Down	MP3
49 (956)	Linken Park - One Step Closer	MP3

Table 4. Top 50 - All files

Rank	Filename	Ext
0 (1893)	in the end - linkin park	MP3
1 (1685)	Shakira-whenever,whereever	MP3
2 (1549)	A-Nickleback How You Remind Me	MP3
3 (1505)	creed - -my sacrafice	MP3
4 (1482)	Alein ant farm - Smooth Criminal	MP3
5 (1471)	hey baby - no doubt	MP3
6 (1362)	It's Been Awhile - Staind	MP3
7 (1331)	-jay-z - h to the izo	MP3
8 (1307)	RA	MP3
9 (1299)	Angel Shaggy	MP3
10 (1296)	pod - alive	MP3
11 (1252)	alecia keys - fallin'	MP3
12 (1250)	usher - you got it bad	MP3
13 (1245)	Shaggy - It wasent me	MP3
14 (1230)	Drops of Jupiter - Train	MP3
15 (1228)	The Calling - Wherever You Will Go	MP3
16 (1228)	(A)R. Kelly -The Worlds Greatest	MP3
17 (1211)	Crawling - Linkin Park	MP3
18 (1198)	Creedd - With Arms Wide Openn	MP3
19 (1191)	Blige, Mary J - Family Affair	MP3
20 (1166)	Incubis - I Wish You Were Here	MP3
21 (1161)	nickelback-this is how you remind me	MP3
22 (1139)	usher - you remind me	MP3
23 (1126)	outkast - the whole world	MP3
24 (1124)	Get The Party Started - Pink	MP3
25 (1102)	Country Grammer - Nelly	MP3
26 (1098)	blurry - puddle of mud	MP3
27 (1071)	drive - Incubis	MP3
28 (1069)	craig david - feel me in	MP3
29 (1063)	control - Puddle Of Mud	MP3
30 (1053)	Cread - Higher	MP3
31 (1042)	get this party started - pink	MP3
32 (1028)	a-Lifehouse - Hanging by a Moment	MP3
33 (1027)	five for fighting - superman	MP3
34 (1023)	enrique eglasias - hero	MP3
35 (1020)	Eagels - Hotel California	MP3
36 (1016)	All or Nothing - O-Town	MP3
37 (1012)	lead zepplin - Stairway to Heaven	MP3
38 (994)	Last Resort-Poppa Roach	MP3
39 (992)	Nelley-EI	MP3
40 (978)	Ludacris - Role out	MP3
41 (974)	chop suey - System of a Down	MP3
42 (956)	Linken Park - One Step Closer	MP3
43 (915)	(a) Michelle Branch - Everywhere	MP3
44 (915)	pod - youth of a nation	MP3
45 (908)	Diddo - Thank You	MP3
46 (900)	fat joe feat. r. kelly - we thuggin	MP3
47 (896)	Affroman - Because I Got High	MP3
48 (892)	i do - Toya	MP3
49 (868)	I'm A Thug - Trick Daddy	MP3

Table 5. Top 50 - Audio files

10. L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: Evidence and implications," in *In Proc. IEEE INFOCOM 1999*, March 1999.
11. K. M. Hanna, N. Natarajan, and B. N. Levine, "Evaluation of a novel two-step server selection metric," in *in Proc. IEEE ICNP 2001*, November 2001.

Rank	Filename	Ext
0 (537)	beck	ASF
1 (140)	(Comedy) - Basketball (so funny you'll pee your pants)	AVI
2 (131)	sample	MOV
3 (118)	britney spears - i'm a slave for you	MPEG
4 (91)	Shakira - Whenever, Wherever	MPEG
5 (79)	[pornographic]	ASF
6 (64)	Funny Videos - Msu Cheerleader Attacking Wisconsin Mascot	MPEG
7 (63)	Mtv-jackass-shopping Carts	MPG
8 (62)	waiting	AVI
9 (61)	comedy Giving The Finger To A Cop (police brutality-really funny)	AVI
10 (59)	No Doubt - Hey Baby	MPG
11 (59)	Comedy - Sorriest Fight Ever Recorded	MPEG
12 (58)	hlcell	AVI
13 (56)	Comedy - Granny Kicks a Baby..funny!	MPEG
14 (55)	SNL - Celebrity Jeopardy - Adam Sandler, Connery, Cruise	MPG
15 (54)	lord of the rings - fellowship of the ring(1of2)	AVI
16 (53)	Comedy - Cat Attacks Kid (funny)	MPG
17 (53)	Snl - Adam Sandler & Chris Farley - Schmitts Gay Beer	MPEG
18 (52)	Jennifer Lopez - Ain't It Funny	MPG
19 (51)	videotest	RM
20 (51)	[pornographic]	ASF
21 (51)	Comedy - Funny! - Monkey sniffs butt, passes out!	MPEG
22 (51)	The Simpsons - Scary Movie funny	MPG
23 (51)	Pamela Anderson Tommy Lee sex video 1	MPEG
24 (51)	Budweiser - Comedy - Wassup - Simpsons	AVI
25 (51)	[pornographic]	MPEG
26 (50)	family guy - osama bin laden	MPEG
27 (50)	firstrun	RM
28 (49)	jennifer_lopez_feat_ja_rule-im_real_(remix)-(buggout-xvcd)-hhv	MPG
29 (49)	Pamela Anderson with Tommy Lee	MPG
30 (49)	blink_182_-_dammit	MPG
31 (49)	nelly_-_#1	MPEG
32 (48)	pink-get the party started	MPG
33 (48)	Adult Movies - Wifey - Student 2	MPEG
34 (48)	[pornographic]	MPG
35 (48)	Comedy - Fart - Matrix Fart - Extremely Funny	MPEG
36 (48)	SNL (Saturday Night Live) - Matt Foley - Mexican House - Chris Farley - Jay Mohr	MPG
37 (47)	Southpark - The Matrix	MPG
38 (47)	Sarah Michelle Gellar - acting as Britney Spears (SNL)	MPG
39 (46)	logo	AVI
40 (46)	Jackass - Fast Food Football	MPG
41 (46)	[pornographic]	MPEG
42 (45)	[pornographic]	MPG
43 (45)	SNL - Celebrity Jeopardy - Connery, Jones, Williams (Anal Bum Cover)	MPEG
44 (45)	[pornographic]	MPG
45 (45)	SNL - Celebrity Jeopardy - Connery, Reynolds, Stewart (Ape tit)	MPEG
46 (45)	[pornographic]	ASF
47 (45)	[pornographic]	MPG
48 (45)	jenna jameson nurse	MPG
49 (44)	Faces Of death - Kid Gets Kicked In His Throat By Some Dancer(Funny As Hell)(1)	MPEG

Table 6. Top 50 - Video files