Characterizing Files in the Modern Gnutella Network

Daniel Stutzbach, Shanyu Zhao, Reza Rejaie

University of Oregon

{agthorr, szhao, reza}@cs.uoregon.edu

The Internet has witnessed an explosive increase in the popularity of Peer-to-Peer (P2P) file-sharing applications during the past few years. As these applications become more popular, it becomes increasingly important to characterize their behavior in order to improve their performance and quantify their impact on the network.

In this paper, we present a measurement study on characteristics of available files in the modern Gnutella system. We develop two new methodologies to capture accurate snapshots of available files in a large scale P2P system. This methodology was implemented in a parallel crawler that captures the entire overlay topology of the system where each peer in the overlay is annotated with its available files. We have captured more than fifty snapshots of the Gnutella system that span over one year period. Using these snapshots, we conduct three types of analysis on available files: *(i)* Static analysis, *(ii)* Topological analysis and *(iii)* Dynamic analysis. Our results reveal several interesting properties of available files in Gnutella that can be leveraged to improve the design and evaluation of P2P file-sharing applications.

1 Introduction

During the past few years, the Internet has witnessed an explosive increase in the popularity of Peer-to-Peer (P2P) file sharing applications. Today's popular P2P file-sharing applications such as eDonkey, FastTrack, and Gnutella have more than one million users each at any point of time [2] and make up a significant portion of Internet traffic [3]. These applications are primarily used for exchanging multimedia files among participating peers. Participating peers collectively form an overlay that is used to search for desired files among those available throughout the system. As file sharing applications become more popular, characterizing their behavior becomes increasingly important because it reveals performance limitations of these applications in practice as well as their impact on the network. To fully characterize the behavior of file-sharing applications, three

This paper extends and supplants the earlier version of this paper presented at MMCN 2006 [1].

equally important and related aspects of these applications should be examined through measurement: *(i)* Overlay topology [4, 5], *(ii)* Query workload [6], and *(iii)* Available files [7]. In particular, characterizing available files among participating peers is valuable for several reasons. First, it reveals the properties, distribution and heterogeneity of the contributed resources (*i.e.,* storage space and available files) by individual users in the system. Second, it allows us to identify any potential design anomaly that might be exposed in a practical setting or any opportunity that can be used to improve performance of these systems. Third, collected traces and derived characteristics of available files through measurement can be also used to conduct more realistic simulations or analytical modeling of available files in P2P systems.

During the past few years, a handful of previous studies have characterized the distribution of shared files in various P2P file sharing applications [7–10]. While these studies shed an insightful light on the characteristics of files in file-sharing applications, they have several limitations. First, almost all the previous studies have focused on a subset of peer population in their target file-sharing applications (*i.e.,* less than 20k peers). However, to our knowledge, none of these studies have verified whether the captured peers are indeed representative of the entire population. Second, many of the previous studies (except [7, 8]) are more than three years old and thus rather outdated. During the past few years, P2P file-sharing applications have significantly grown in size and have incorporated new features. In particular, most popular file sharing applications have adopted a two-tier architecture along with new search mechanisms (*e.g.,* dynamics querying in Gnutella) to improve their scalability. However, the effect of these new features on the characteristics of files has not been studied. Third, previous studies have not examined the dynamics of file characteristics over time nor any possible correlation between the overlay topology and file distribution.

In this paper, we empirically characterize available files across reachable peers in the modern Gnutella network. We present two new measurement methodologies to capture an accurate snapshot of the system at a particular point of time. A snapshot contains participating peers in the system, available files at each peer and the pair-wise connectivity among peers (*i.e.,* the overlay topology). We have developed a fast P2P crawler called Cruiser [11]. Using Cruiser, we have captured more than 50 snapshots of the files available in Gnutella that span over one year and each snapshot contains more than 100 million distinct files in each snapshot. Using these snapshots, we conduct the following analysis on shared files in Gnutella:

- **1) Static Analysis:** We examine properties of contributed resources (*i.e.,* files, and storage space) by participating peers in individual snapshots of the system.

- **2) Topological Analysis:** We investigate any potential correlation between the pattern of file distribution

among peers and the overlay topology.

- **3) Dynamic Analysis:** We study variations in the popularity of individual files in the system, the variations in available files at individual peers, and the percentage of changes in the Top-N files over time.

We also leveraged an unbiased sampling technique for P2P system that we developed in our earlier work [12,13] to select a representative subset of peers. This sampling technique is implemented in a light-weight sampling tool, called `ion-sampler`. We use `ion-sampler` to derive static properties of available files in Gnutella and validate the similar characteristics that are derived from complete snapshots captured by Cruiser.

Our main findings can be summarized as follows: *(i)* Free riding has significantly decreased among Gnutella users during the past few years and is significantly lower than in other P2P file-sharing applications such as eDonkey. *(ii)* The number of shared files and contributed storage space by individual peers are heavily skewed. *(iii)* The popularity of individual files is also heavily skewed, with a handful of files being extremely popular while a majority of files are very unpopular. *(iv)* The most popular file type is the MP3 file, which accounts for over two-thirds of all files and one-third of all bytes. Both the popularity and occupied space by video files has tripled over the past few years. Furthermore, the number of video files are less than one-tenth of audio files but they occupy 25% more bytes. Overall, 93% of bytes in the system are occupied by audio and video files. *(v)* Files are randomly distributed throughout the overlay and there is no strong correlation between the available files at peers that are one, two or three hops apart in the overlay topology. *(vi)* Shared files by individual peers slowly remain relative stable over the timescale of few days. Over the entire system, more popular files experience larger variations in their popularity. Furthermore, the popularity of most files follows a roughly similar pattern of initial increase for a few months until they reach their peak popularity and then gradual decrease after staying at the peak popularity for a period of time. *(vii)* the percentage of changes in the Top-N files over time does not strongly depend on N. The Top-N list undergo roughly 20% changes after one week. Furthermore, 60% of files that remain in the Top-N list for at least one week, will remain in the Top-N for at least 10 weeks.

Why Characterize Gnutella? We conducted our empirical study of Gnutella based on a number of considerations. First, Gnutella is one of the most popular P2P file-sharing networks on the Internet [2], with millions of simultaneous users. Second, Gnutella has a protocol hook that allows a list of shared files to be accurately and efficiently extracted from a peer. Finally, Gnutella is one of the most studied P2P systems in the literature. This enables us to compare and contrast the behavior of modern Gnutella with earlier empirical studies of Gnutella and gain insights on changes in the system.

The rest of this paper is organized as follows: We provide an overview of the Modern Gnutella in Section 2.

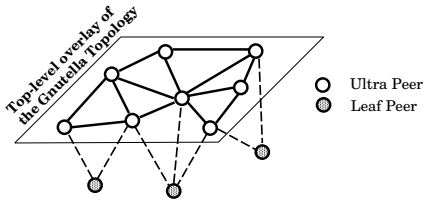


Figure 1: Two-Tier Topology of Modern Gnutella

Section 3 describes the challenges in capturing accurate snapshots and describes our measurement methodology and tools. Section 4, 5 and 6 present static analysis, topological analysis, and dynamics analysis of files in the Gnutella network, respectively. Section 7 provides an overview of previous studies. Finally, Section 8 concludes the paper and sketches our future plans.

2 Overview of Modern Gnutella

Gnutella is widely regarded as the first fully decentralized peer-to-peer file-sharing system. However, it has evolved considerably since its initial release in early 2000, and grown dramatically in size (especially over the last couple of years [4, 14]). Today, Gnutella is one of the largest P2P networks in operation [2]. Similar to many unstructured P2P networks, each Gnutella peer joins the network by establishing TCP connections to several existing peers in the system. In the original Gnutella protocol, participating peers form an unstructured overlay topology that is used by each peer to perform flood-based searches among other peers. To improve the scalability of the original protocol, modern Gnutella clients adopt a two-tier overlay structure along with a dynamic query distribution mechanism [15].

Two aspects of Gnutella are pertinent to our study. First, because one of our goals is to examine correlations between the distribution of shared files and location in the overlay topology, a general understanding of Gnutella's overlay topology is required. Second, we use Gnutella's Browse-Host extension [16] to acquire the list of files shared by each peer.

Two-Tier Topology: As shown in Figure 1, modern Gnutella clients implement a two-tiered overlay structure by dividing peers into two groups: *ultrapeers* (or super-peers) and *leaf peers*. Each ultrapeer neighbors with several other ultrapeers to form the top-level overlay. The majority of the peers are leaf peers and connect to the overlay through a few (2 to 3) ultrapeers. High-bandwidth, unfirewalled leaf peers become ultrapeers on demand in order to maintain a proper ultrapeer-to-leaf ratio. When a leaf connects to an ultrapeer, it uploads a hashes of its filename keywords to that ultrapeer. This allows the ultrapeer to only forward messages to the leaf peers that

might have matching files. Leaf peers never forward messages. This approach reduces the number of messages forwarded towards leaf peers which in turn improves the scalability of the system.

The Browse-Host Extension: One important reason for selecting modern Gnutella for file characterization is its support for a suite of open and moderately well-documented protocols [17]. The Browse-Host extension [16] is an extension of Gnutella protocol that enables one peer to view the list of files shared (called a *sharing list*) by another peer. This feature allows users with similar interests to learn about new material which may appeal to them. Browse-Host is supported by the two major Gnutella implementations, BearShare and LimeWire, among others. These two implementations collectively make up roughly 95% of Gnutella ultrapeers [4]. This implies that the Browse-Host extension is able to capture the shared files by most peers in Gnutella.

3 Measurement Methodology

Our goal is to capture a *snapshot* of the Gnutella network at a given point of time that contains *(i)* all participating peers and the pair-wise connectivity among them (*i.e.,* an overlay snapshot), and *(ii)* available files at each participating peer in the overlay (*i.e.,* a file snapshot). In essence, we need to capture snapshots of the overlay topology and annotate each peer with its available files. A common approach to capture a snapshot is to use a P2P crawler. Given a set of initial peers, a crawler contacts individual peers to capture their available files and collect information about their neighboring peers in the session. The crawler progressively learns about more peers in the session and contacts them until no other new peers are discovered. However, because the overlay changes as the crawler operates, snapshots captured by a crawler are inherently *distorted* [18]. More specifically, as the crawler explores the overlay, many peers join or leave the system which results in changes in the overlay topology and possibly changes in the set of available files throughout the system. Therefore, the captured snapshot contains a group of peers that have departed or arrived during a crawl. This problem is further aggravated in large overlays since the arrival of a sufficiently large number of new peers may significantly increase the duration of a crawl and thus inflate the population of peers in a snapshot[1].

Previous studies implicitly addressed this problem by adopting one of the following ad-hoc sampling schemes to capture a partial snapshot of a P2P system: *(i) Partial Snapshot Through a Short Crawl*: Some studies periodically capture a small subset of participating peers (*i.e.,* a partial snapshot) through a short crawl [10], *(ii) Periodic Probe of a Fixed Group*: Other studies identify a subset of some participating peers (using a partial crawl or passive monitoring) and periodically probe the same group of peers to collect information about their

[1] In the extreme case, the crawler may never terminate since there are always new peers to contact.

available files [9]. In the absence of any solid understanding of churn and file characteristics in P2P systems, it is not clear whether these sampling strategies capture a representative population of peers. In our earlier work, we showed that these ad-hoc sampling techniques could lead to significant bias towards the properties of the short-lived or high degree peers [12].

We developed the following two measurement methodologies to capture lists of shared files in the Gnutella network: *(i)* Capturing complete snapshots, *(ii)* Capturing unbiased samples. We describe any of these measurement approaches in the following subsections. Capturing complete snapshots is our primary characterization technique and we use unbiased sampling for validation.

3.1 Capturing Complete Snapshots

Our goal is to capture the entire population of participating peers in the Gnutella network (*i.e.,* a complete snapshot) within a short period to minimize any potential bias in our characterization. Note that the time required to obtain the list of available files at individual peers is significantly longer than for obtaining neighbor information. For example, obtaining a list of neighbor peers from a peer may take less than a second whereas the list of available files may take several minutes to download if the peer has many files. In summary, a topology crawl is significantly faster than a file crawl. Therefore, we decouple topology and content crawls to improve the accuracy of captured snapshots and conduct our snapshots in three steps as follows: First, we conduct a *topology crawl* to quickly capture all participating peers and their pair-wise connectivity, *i.e.,* capturing the overlay topology. Second, we conduct a *content crawl* and collect the list of files available at each one of the peers identified during the topology crawl. Third, once the content crawl is completed, we conduct another topology crawl in order to identify those long-lived peers in the initial snapshots that remained in the system during the entire content crawl. At the end of these three steps, we create a snapshot of the overlay topology where each node is annotated with its available file and a label that describes whether it is long- or short-lived. Since some of the captured peers in the first topology crawl depart the system during the content crawl, the collected content in our measurement is slightly biased towards peers with longer uptime. This motivates our second technique, unbiased sampling, described in the next subsection.

To minimize distortion in captured snapshots, we have developed a parallel P2P crawler, called Cruiser [11], that can crawl an overlay orders of magnitude faster than any previous crawler. Cruiser achieves this goal by significantly increasing the degree of concurrency in the crawling process. Toward this end, Cruiser adopts a master-slave architecture where each slave crawls hundreds of peers simultaneously and the master coordinates

among multiple slaves. This architecture allows us to run Cruiser on multiple co-located boxes to further increase the crawling speed. Using six off-the-shelf 1 GHz GNU/Linux boxes in our lab, Cruiser can perform a topology crawl for more than two million Gnutella peers in less than 15 minutes, and perform a content crawl within 5.5 hours, *i.e.,* capturing the annotated snapshot takes 6 hours, (15min + 5.5hr + 15min). During the content crawl, Cruiser collects the file name and content hash (SHA1) for each shared file on every reachable peer, resulting in a log file more than 10 GB in size.

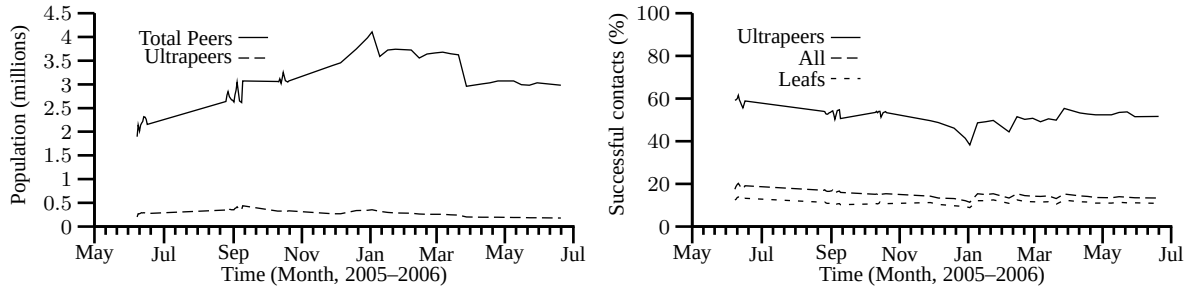
3.2 Capturing Unbiased Samples

Capturing complete snapshots is a heavy-weight operation that provides a tremendous amount of information. However, even with a fast crawler, there will inevitably be some distortion as the network changes while the crawler operates. Sampling is a useful, light-weight alternative that does not suffer from distortion. However, commonly-used sampling techniques for measuring peer-to-peer systems tend to introduce considerable bias for two reasons. First, the dynamic nature of peers (*i.e.,* churn) can bias results towards short-lived peers, much as naively sampling flows in a router can lead to bias towards short-lived flows. Second, the heterogeneous nature of the overlay topology can lead to bias towards high-degree peers, as peers with a high degree are more likely to be discovered. In our prior work [12, 13], we developed a light-weight sampling tool, called `ion-sampler`, for peer-to-peer systems that selects peers uniformly at random. The `ion-sampler` tool performs a random walk and selects the peer at the end of the walk as s ample. To correct for the bias towards high-degree peers, `ion-sampler` employs the Metropolis–Hastings method to weight the walk to result in uniform sampling [19–21]. To avoid any bias based on session length, we allow re-sampling; effectively, instead of attempting to sample a peer we are trying to sample a peer's state (which may vary with time). The `ion-sampler` tool allows us to gather many samples at once by running many walks in parallel. For this study, we use 1,000 parallel walks. We use `ion-sampler` to collect peers, then gather the list of files available from each sampled peer. Our extensive simulations [13] show that the technique yields unbiased samples under a wide variety of peer behavior, degree distributions, and overlay construction techniques.

While sampling does not suffer from distortion, it provides fewer details than a full crawl. In particular, it does not include a topological snapshot or session length information. Additionally, sampling provides little information about unusual peers, *e.g.,* in the tail of distributions. For these reasons, we employ sampling to validate the results of our static analysis from complete snapshots.

Type	TCP Refused	Timeout	Conn. Lost	App. Refused
Ultrapeer	31.30%–46.10%	1.74%–13.25%	1.78%–6.80%	0.97%–2.30%
Leaf Peers	53.53%–72.13%	1.12%–3.85%	4.24%–23.16%	1.27%–3.06%
All Peers	31.63%–46.74%	1.75%–13.03%	1.81%–7.08%	0.97%–2.30%

Table 1: The range of percentage of peers that failed during the file crawl for various reasons across all snapshots



(a) Number of Ultrapeers & Total Peer across individual snapshots

(b) Percentage of successfully contacted peers among ultrapeers, leaves and total peers

Figure 2: Population and reachability across captured snapshots

3.3 Dataset

We have captured more than 50 snapshots of the Gnutella network annotated with the list of files available at each peer. Our snapshots consists of daily snapshots from consecutive days during the following intervals: 6/8/2005–6/18/2005, 8/23/2005–9/9/2005 and 10/11/2005–10/21/2005.[2] Furthermore, we have collected 24 snapshots, roughly once per week between 11/29/2005 and 05/30/2006.[3] To avoid any time-of-day effects, each snapshot was captured at approximately the same time-of-day (2pm PDT). These snapshots enable us to examine characteristics of available files in the system over both short and long timescales (*i.e.,* several days and several months). Additionally, we use unbiased samples collected on June 16th, 2006 and July 4th, 2006 to validate our finding in Section 4.

Figure 2(a) depicts the number of ultrapeers and leaf peers across captured snapshots during our measurement period. A subset of peers in each snapshot are unreachable by our crawler for one of the following reasons: *(i)* firewall or NAT blocking incoming traffic, *(ii)* severe network congestion or overloaded processor at the peer, *(iii)* the peer departure, or *(iv)* the peer does not support the Browse Host protocol. Figure 2(b) shows the percentage of ultrapeers, leaf peers, and the total population in individual snapshots that were successfully contacted by the file crawler. This figure shows that in average around 50% of ultrapeers, 12% of leaf peers and

[2]We are missing snapshots for 9/6/2005, 9/8/2005, 10/17/2005, and 10/29/2005 due to a mixture of software, network, and power failures.

[3]The detailed characteristics of individual snapshots are available online at `http://mirage.cs.uoregon.edu/P2P/files/`.

18% of the total population are reachable by the file crawler. Table 1 presents the minimum and maximum value for the percentage of peers within each group (ultrapeer, leaf, total population) for four different reasons across all snapshots. "TCP Refused" refers to attempts that were rejected by a TCP RST message. "Timeout" refers to attempts that did not receive any response. "Conn. Lost" refers to attempts that established a TCP connection, but the connection closed before the file list could be fully retrieved. Finally, "App. Refused" refers to attempts where TCP connected, but the request for the list of files was rejected at the application level. In a few cases, our crawler machines were under heavy load by other users during our crawls, leading to a large percentage of timeouts and lost connections. Typically these values were only a few percent. Since ultrapeers are not allowed to be firewalled, any reported connection error for ultrapeers indicates that the contacted peer has departed. However, connection errors for leaf peers might occur due to peer departure or a firewall[4]. In our earlier study of Gnutella overlay topology [4], we showed that about half of all leaf peers captured in a topology crawl leave the overlay within a 5 hour period. Independent online statistics [22] report that around 70% of leaf peers in the Gnutella network are firewalled. These evidences support the accuracy of the high ratio of connection errors that we experienced for leaf peers (in the column labeled "TCP Refused"). In summary, while we aim to capture complete snapshots, we can successfully contact only 20% of all peers in one snapshot (around half a million peers) to obtain their list of available files primarily due to two reasons: *(i)* the number of departed peers during the long file crawl, and *(ii)* the large number of leaf peers behind firewalls.

3.4 Challenges and Problems

We briefly discuss several interesting problems that we experienced during data collection and data processing.

Low-bandwidth TCP Connection: Although Cruiser has a timeout mechanism that closes any idle connections after 20 seconds, we noticed that some crawls do not complete after the crawling queue becomes empty. Further examinations revealed that around 80 peers in each crawl send their data at an extremely low rate (around 20 bytes per second) which prevents Cruiser from closing their connections. We instructed Cruiser to terminate a crawl a few minutes after its crawling queue becomes empty. Given the negligible number of these misbehaved peers, this should not have any affect on our analysis.

File Identity: We use the content hash of a file returned by the target peer to uniquely identify individual files. In our initial measurements, we observed many files with the same name but different content hashes (*e.g.*, setup.exe, login.bmp). This illustrates that the trimmed (or even complete) filenames used by previous studies

[4]We are not aware of any reliable technique to distinguish between these two scenarios.

(*e.g.*, [9]), are not reliable file identifiers. We discovered around 3,500 files without content hash value in each snapshot and eliminated them from our analysis.

Post-processing: To compute the popularity of individual files in the system, we needed to keep track of more than 100 million distinct files discovered in each snapshot. The large number of files resulted in memory bottlenecks in our analysis. We leveraged the skewed distribution of popularity to address this problem as follows: We divide captured peers in a snapshot into seven segments where each segment contains a random subset of the peers. Then, we calculate the popularity of files within each segment, trimming all files with fewer than 10 copies in a segment, and combine all the trimmed segments into a single snapshot. This approximation eliminated several million distinct files and prevented memory bottlenecks during our post-processing. While this prevents us from performing analysis on the least popular files (with fewer than 70 copies in the entire network), it does not significantly affect conducted analysis on more popular files.

4 Static Analysis

In this section, we examine the following static characteristics of available files across peers in individual snapshots regardless of their location in the overlay topology: *(i)* the percentage of free riders, *(ii)* the degree of resource sharing among cooperative peers, *(iii)* the distribution of file popularity, and *(iv)* the distribution of file types. Furthermore, we investigate the variations of these characteristics across our ten-month measurement period to examine the extent to which the properties are fundamental rather than ephemeral. We derive these characteristics from both complete snapshots and sampled peers and compare them to verify the accuracy of our results. We also compare our findings with previous studies to illustrate any potential changes in these characteristics of the Gnutella network over the past few years.

4.1 Ratio of Free Riders

The success of P2P file sharing systems depends on the willingness of participating peers to share their files. However, previous studies have frequently reported that participating peers do not have an incentive to contribute their resources (*e.g.*, disk space and network bandwidth) to the system and thus only use resources offered by others, *i.e.*, become "free riders". In particular, Adar *et al.* [23] reported that 66% of Gnutella peers were free riders in 2000, while a study by Saroiu *et al.* [10] found 25% were free riders, with 75% of peers sharing less than 100 files in 2002. A recent study also reported 68% were free riders in eDonkey [7].

Table 2 presents the range (minimum–maximum values) for the number of ultrapeers (first row), leaf peers

	Contacted Peers	Free Riders(%)	Files/Peer
Ultra	94,329–188,107	8.96–10.54	324–358
Leaf	253,164–395,682	10.81–15.85	347–406
Long-lived Ultra	69,441–147,435	9.27–10.78	320–354
Short-lived Ultra	24,888–40,672	8.08–9.77	335–371
Long-lived Leaf	166,241–240,969	11.39–16.93	369–437
Short-lived Leaf	86,923–154,713	9.93–13.79	316–358
Total	410,252–552,702	10.63–13.41	340–393

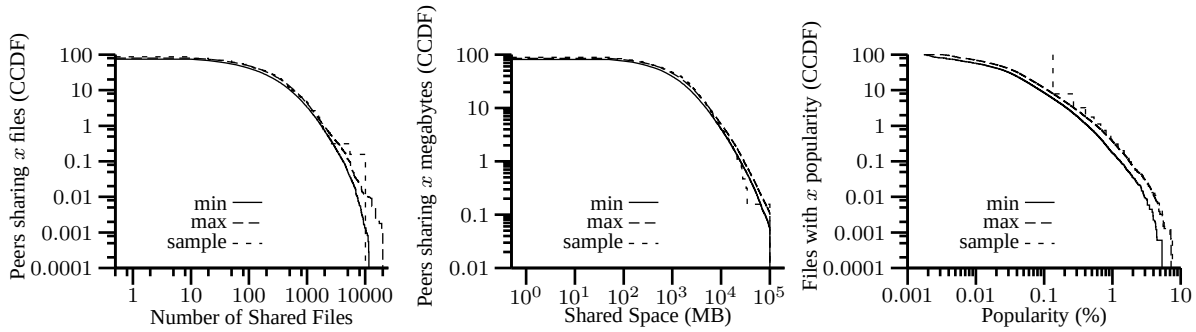
Table 2: Percentage of free-riders, its breakdown across different groups, and the mean number of shared files among peer

(second row) and total peers (last row) across all snapshots. We further divide ultrapeers (row 3 and 4) and leaf peers (row 5 and 6) into short-lived and long-lived based on their presence in the second topology crawl as we discussed in Section 3. These grouping reveals any potential differences in the free riding between ultrapeers and leaves as well as long- versus short-lived peers. For each one of the above groups, the corresponding row in Table 2 presents the range of the following properties across all snapshots *(i)* the number of contacted peers that provided their sharing list (2nd column), *(ii)* the percentage of free riders (3rd column), and *(iii)* the average number of shared files among peers in each group.

Table 2 shows several interesting points as follows. The ratio of free riders in Gnutella has significantly dropped from 25% in 2002 [10] to around 13% among all participating peers (*i.e.,* last row), and it is drastically lower than the 68% recently reported in eDonkey [7]. We speculate that several factors have contributed in the observed drop in the percentage of free riders including the increase in access link bandwidth for average Internet users and active marketing efforts by the Gnutella vendors encouraging their users to share. Table 2 reveals that the percentage of free riding among ultrapeers (9.0–10.5%) is somewhat lower than that in leaf peers (10.8–15.9%). However, on average leaf peers share more files. Since leaf peers constitute a larger portion of the total population (86–90%), their behavior has a greater impact on system performance. Long-lived peers appear to have a slightly higher percentage of free riders compared to short-lived peers. The correlation between lifetime and number of files shared is inconclusive. Long-lived leaf peers share more files than short-lived leaf peers, but long-lived and short-lived ultrapeers share about the same number of files.

4.2 Degree of Resource Sharing Among Cooperative Peers

We now focus our attention on cooperative peers and characterize their willingness to contribute their resources (*i.e.,* both files and storage space). During our analysis, we noticed that the sharing lists of many peers contain duplicate files. This occurs because most Gnutella clients simply put various folders with potentially duplicate



(a) Distribution of number of shared files across peers

(b) Distribution of number of shared bytes across peers

(c) Distribution of file popularity across unique files

Figure 3: Distributions of files and bytes

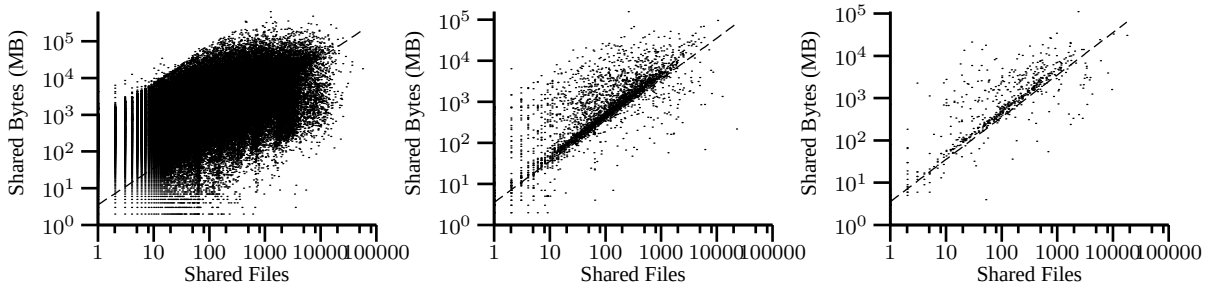
content under the sharing folder. We have excluded all the duplicate files from captured sharing lists (which account for around 10 million files or roughly 9% of all captured files) in all the reported analysis in this paper.

In the conference version of this paper [1], we depicted the number of shared files and shared bytes per peer as a histogram on log-log scale and concluded that both distributions resemble power-law due to their approximately linear shape. Presenting these distributions as histogram results in a "messy", difficult-to-interpret tail of the distribution, due to the small number of peers in that region. This in turn makes it difficult to properly examine the slope of distribution. Presenting the distribution as a Complementary Cumulative Distribution Function (CCDF) yields a smooth tail without losing any information while retaining the linear slope that a power-law distribution exhibits on a log-log scale (since $\int_x^\infty u^{-\alpha} du \propto x^{-\alpha}$). Toward this end, we compute the CCDF of the number of shared files and shared bytes per peer for individual snapshots. To present the variability of these distributions across all snapshots in a compact and clear fashion, we compute the MIN-CCDF and MAX-CCDF, defined as follows:

$$\text{MIN-CCDF}(x) = \min(\text{CCDF}_i(x) \text{ for all snapshots } i)$$

$$\text{MAX-CCDF}(x) = \max(\text{CCDF}_i(x) \text{ for all snapshots } i)$$

Figures 3(a) and 3(b) present the MIN-CCDF and MAX-CCDF for CCDF of shared files and shared bytes across all snapshots, *i.e.,* the CCDFs from all the snapshots fall in between the MIN-CCDF and MAX-CCDF curves in the corresponding graph. Figures 3(a) and 3(b) also include the CCDF for shared files and shared bytes per peer based on samples collected with `ion-sampler`, to verify whether the results from complete snapshots are significantly affected by distortion or not (as described in Section 3). Note that the MIN-CCDF and MAX-CCDF curves in both figures are very close which indicates that the the CCDF for shared files and



(a) All peers from a snapshot

(b) One in every 100 peers from a complete snapshot

(c) All peers captured via sampling

Figure 4: Scatter-plot of number of shared bytes as a function of number of shared files. Each point represents one peer. The reference line is 3.544 MB per file.

shared bytes per peer across all snapshots are very stable. Furthermore, the CCDFs from sampling are consistent with the results from complete snapshots, except in the tail of the distribution where the precision of sampling is low.

To verify our earlier conclusion, we recall that power-law distributions are *scale-free*. This means that the "slope" (α) of distribution is consistent over many orders of magnitude on a log-log scale. While the histograms presented in the conference version of this paper appear approximately linear [1], it is not possible to closely fit a line over the entire range of x values due to the subtle downward curvature. This curvature is more apparent in the CCDFs in Figures 3(a) and 3(b). Based on these considerations, we revise our earlier conclusions as follows: the distribution of the number of files and bytes shared by individual peers are heavily skewed but does not appear to follow a power-law distribution. Most peers share a moderate amount of files (or bytes) while a few peers contribute an enormous amount of content (or space). The median value for shared files is around 70 files while 0.01% of peers share more than 7,500 files. The median value of shared space is around 650 MB while 0.1% of peers contribute more than 85 GB.

Correlation between Contributed Files and Bytes: Saroiu *et al.* [10] reported a strong correlation between the number of shared files and the volume of shared data across Gnutella peers in 2002. Figure 4(a) shows this correlation as a scattered-plot across all cooperative peers in one snapshot (June 13$^{\text{th}}$, 2005). Each point in this figure represents the number of shared files versus the shared disk space for each cooperative peer. We observe a greater variability than what has been reported by Saroiu *et al.* in 2002. However, this may simply be due to the larger number of data points in our scatter-plot. To verify this issue, we sampled one out of every hundred data points in Figure 4(a) and the selected samples are shown in Figure 4(b). Figure 4(b) reveals a strong correlation between the number of files and the number of bytes contributed by each peer which was obscured in Figure 4(a). We performed a least-squares fit to a line through the origin (0 files = 0 bytes), finding

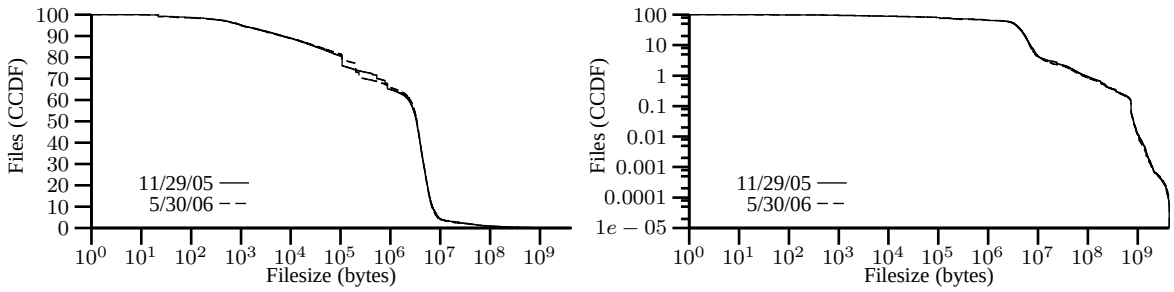


Figure 5: Distribution of filesize across all files

a slope of 3.544 MB per file. This is consistent with the size of a MP3 file that is a few minutes long (*i.e.,* a typical song). Figure 4(c) shows the correlation between the contributed bytes and files by each peer using the data gathered by `ion-sampler` on June 16th, 2006—more than a year after the presented snapshot data in Figure 4(a). Figure 4(c) fits to the same line and is consistent with the results from complete snapshots.

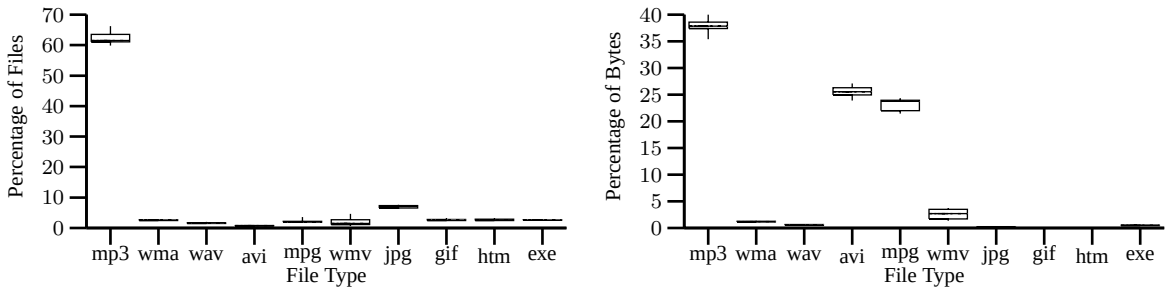
To gain a deeper insight about the correlation between contributed files and bytes, we derive the CCDF distribution of file size among all files in individual snapshots. Figure 5(a) depicts the CCDF distribution of file size in two snapshots that are six months apart in a linear-log scale. [5] Figure 5(a) shows that the distribution of file size has very little change between the two snapshots that are six months apart. This figure also indicates that 60% of all files are a few megabytes in size (*i.e.,* the graph has a knee between 1 MB and 10 MB). Furthermore, the majority of other files (around 40%) are smaller than 1 megabyte in size. Figure 5(b) shows the same CCDFs for file size distribution of two snapshots on a log-log scale. This figure illustrates that a very small portion of files (around 0.1%) have the size larger than 1 gigabyte.

4.3 File Popularity Distribution

The distribution of popularity for individual files throughout the system is an important property that shows the degree of similarity and thus availability of individual files among participating peers. Chu *et al.* [9] showed that the file popularity follows a log-quadratic distribution, which can be viewed as a second-order power-law distribution, among Gnutella peers in 2001. Furthermore, Fessant *et al.* [7] recently reported a Zipf distribution for the file popularity in eDonkey. However, none of these studies have captured a large number of peers.

Figure 3(c) shows the range of CCDF for file popularity as a function of its rank (in log-log scale) among cooperative peers across all snapshots. Each snapshot typically contains more than 800 terabytes worth of

[5] We were unable to process the file size distribution for all 50 snapshots. In the final version of the paper we will present the distribution of file size for all snapshots in a min-max fashion similar to Figure 3(a).



(a) Contributed files per file type

(b) Contributed bytes per file type

Figure 6: Breakdown of file types based on the contributed number of files or bytes by each file type

content in more than 100 million unique files based on information from 0.4 million peers, constituting 18.5% of identified peers. If we assume that unreachable peers have similar profiles, the volume of available content in the Gnutella network is around 4,400 terabytes. Figure 3(c) illustrates two points: *(i)* file popularity is heavily skewed, and *(ii)* the distribution of file popularity has remained very stable across the eleven month measurement period. A small number of files are very popular while for most files only few copies are available in the network. Again, we also present data collected using unbiased sampling, which closely matches the snapshot data except for very unpopular files where sampling has poor precision.

4.4 File Type Analysis

We have also examined the distribution of available files among Gnutella peers across different types of video and audio formats. This basically illustrates what types of content are available in the system and thus exchanged among peers. Chu *et al.* [9] conducted similar analysis for Gnutella peers in 2001 and reported that audio files constitute 67.2% of files and 79.2% of bytes. However, video files were significantly less popular and only contributed 2.1% of files and 19.1% of bytes. Using our snapshots, we analyze the various types of audio and video files based on file extensions.

Figure 6(a) and 6(b) depict the contribution of the most common file types in the total number of files and the total number of bytes across cooperative peers, respectively. As box-and-whisker plots, these figures present the variations of the contribution by different file types across all snapshots by showing the 75th percentile (the top of the box), 25th percentile (the bottom of the box), the median value (the line through the box), the maximum value (the top of the line) and the minimum value (the bottom line). The distribution of file types across all snapshots seems fairly consistent with respect to both files and bytes (*i.e.*, all boxes are small). Note that except for MP3 files, each file types individually make up at most a few percent of all files. Normally, MP3 audio files are significantly more popular than any other file type, making up two-thirds of all files and occupying more

than one third of all disk space in the system.

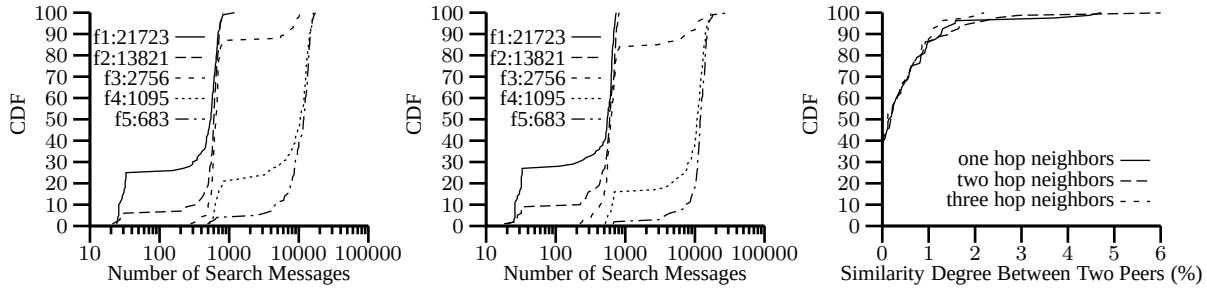
Although non-media files (jpg, gif, htm, exe, txt) are among the top ten most popular types, audio and video files collectively occupy more than 93% of bytes and make up more than 73% of all files in the system. These figures also reveal that video files (avi, mpg, wmv) are significantly larger than other file types in the system. The audio files (MP3, wma, wav) account for 67% of files and 40% of bytes whereas video files constitute around 6% of files but 52.5% of bytes among Gnutella peers (*i.e.*, most files are audio files, but most bytes are in video files). Comparing to the reported results by Chu at al. in 2001 [9], video files have become three times more popular and occupy almost three times more space in the system.

We are unable to apply our sampling technique for file type analysis because it samples random *peers* not random *files*. Consequently, analyzing the file data gathered with `ion-sampler` shows significant correlations based on the preferences of the peers collected. For example, in one instance we saw that 16% of files had the jpg extension, due to sampling two peers with large jpg archives.

5 Topological Analysis

In this section, we investigate whether copies of a given file are located at close-by peers (*i.e., topologically clustered*) or at randomly scattered peers throughout the overlay. Understanding this issue would be useful in both the design and evaluations of search techniques in file-sharing applications. There are two factors that affect the location of available copies of individual files throughout the overlay: *(i)* scoped search, and *(ii)* the dynamics of peer participation (or churn). To accommodate scalability, each searching peer often examines available files among nearby peers (*i.e.*, conducts scoped search). This suggests that a single copy of a file gradually diffuses outward from the original location to other nearby peers and some type of topological clustering might exist. However, due to the dynamics of peer participation (or churn), the location of a peer in the overlay changes over time and could prevent such clustering. The key question is which one of these two factors has the dominating effect. To answer this question, we generate an annotated snapshot of the overlay topology where each peer is annotated with its available files. Then, we explore topological clustering from the following two perspectives:

Per-File Perspective: We conduct trace-driven simulation of flood-based querying over our annotated overlay topology. Figure 7(a) depicts the CDF of the minimum number of query messages to find five copies of a target file from 100 randomly selected peers in the overlay. Fine lines in this figure correspond to five target files with different popularity from our static analysis. The abrupt steps in the lines indicates one-hop increases in the scope of the search. Note that each CDF has at most two steps. This implies that most of the 100 randomly



(a) True file locations; Number of query messages to reach 5 copies of a file

(b) Randomized file locations; Number of query messages to reach 5 copies of a file

(c) Similarity degree distribution for peers one-hop, two-hops, and three-hops away

Figure 7: Distribution of file popularity from a random subset of peers in each snapshot

selected peers find all 5 copies among other peers that are either n or $n+1$ hops away where n inversely depends on the log of the target file's popularity.

Clearly, more search messages are required for less popular files. If significant topological clustering exists, a few searches will complete using very few messages (*i.e.,* fewer hops) while most searches will require a much larger number of messages (*i.e.,* many more hops). However, the pattern of increase in the number of search messages for less popular files in Figure 7(a) indicates that files are randomly distributed.

To verify this conclusion, we randomized the placement of available files without changing the popularity, guaranteeing that no topological clustering exists in the overlay. Figure 7(b) shows the CDF of the number of required query messages to reach 5 copies of the same target files over the annotated topology with randomized file placement. The high similarity between Figures 7(b) and 7(a) supports our conclusion that no significant topological clustering exists in the locations of a given file throughout the overlay. It is worth noting that our annotated overlay topology is incomplete due to the unreachable peers behind firewalls and departed peers. This implies that the number of messages are likely to be smaller in practice. However, the unreachable peers should not significantly affect topological clustering. More specifically, if a file is available only in a certain region of the overlay, the clustering property is preserved even if information is absent for some fraction of the peers in that region.

Per-Peer Perspective: We also investigate the possibility of any topological clustering from per-peer's perspective by examining the degree of similarity between available content at a random peer and all its one-hop, two-hop, and three-hop neighbors. The similarity degree between two peers is defined as the number of common files divided by the smaller size of the sharing lists for the two peers. Figure 7(c) shows a separate CDF for the average similarity degrees of 100 randomly selected peers with their one-hop, two-hop and three-hop neighbors. If topological clustering were present, the similarity degree would decrease as the distance between comparing

peers increases. However, since the distributions are nearly identical, Figure 7(c) clearly illustrates that there is no correlation between the similarity degree and distance between two peers.

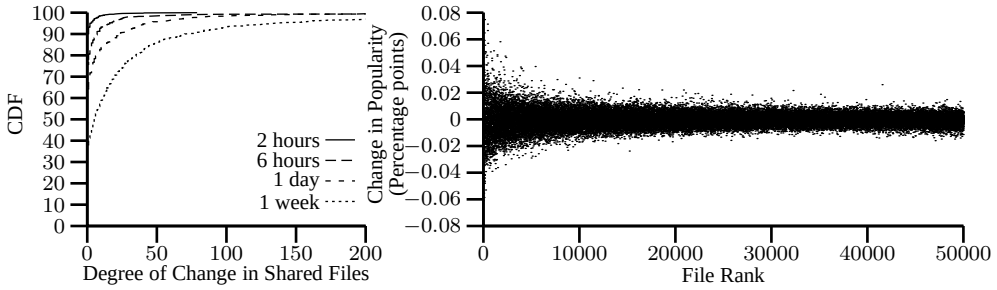
These results suggest that churn is the dominant factor in determining the distribution of files throughout the overlay. In our prior work, we observed that more than half of the Gnutella peers in a given snapshot will depart within 5 hours [24]. Any time a peer joins the overlay, it attaches itself as a leaf peer to several random ultrapeers. Furthermore, qualified leaf peers may become ultrapeers in order to maintain a proper ultrapeer-to-leaf ratio. Hence, the rapid changes in the overlay topology prevents formation of such topological clustering. This finding is important for two reasons: *(i)* Measurement studies may sample the list of files from random peers in any part of the overlay topology, and do not need to capture the entire network. *(ii)* Simulation studies may randomly distribute available files among participating peers regardless of location. However, the number of files per peer and the total number of copies of each file should follow appropriate distributions, examined in Section 4. While previous studies have frequently assumed these properties, to our knowledge, they had not previously been empirically verified.

6 Dynamic Analysis

In this section, we turn our attention to the dynamic properties of available files in Gnutella. More specifically, we investigate how various properties of available files change over time. Prior studies examined changes in the popularity of queries and exchanged files in P2P file-sharing systems (*e.g.*, [6, 25]). However, to our knowledge, no study has previously explored the dynamic characteristics of stored files in P2P systems. For these analysis, we leverage our three sets of daily snapshots that were captured once every two months, as well as several months of weekly snapshots. This dataset allows us to explore dynamic properties over short timescales (*i.e.*, hours and days) as well as long timescales (*i.e.*, months). We explore the following three dynamic properties of shared files by peers in Gnutella: *(i)* variations in shared files by individual peers, *(ii)* variations in popularity of individual files, *(iii)* long-term evolution of file popularity, and *(iv)* variations in Top-n files.

6.1 Variations in Shared Files by Individual Peers

Our goal is to determine how rapidly the available files at individual peers change with time. These dynamics show whether past information about the available files at individual peers can be reliably used in future searches. There are two types of change that can occur to the list of shared files at each peer. First, the user may add new files, either by downloading them from other peers or by manually adding them to the shared folder. Second,



(a) File List Change of Random 3000 Peers

(b) Change in popularity of files over a one day interval

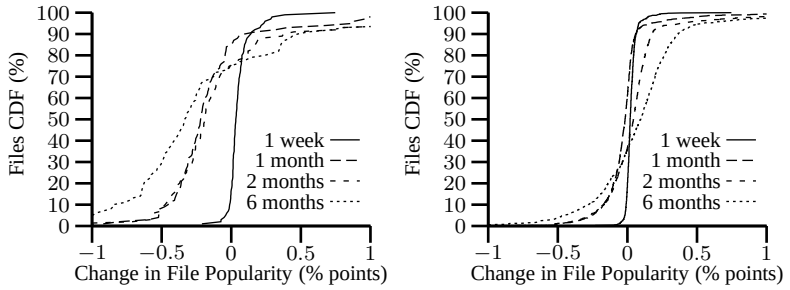
Figure 8: Changes in popularity

the user may remove files, by moving (or deleting) those files from the sharing folder. We note that dynamic IP addresses may introduce error into our results when a peer departs from the system and another peer joins the system with the same IP address. However, our prior study on churn revealed that such events are not common [24]. We define the total number of added and deleted files at a single peer as the *degree of change* to quantify both types of changes in shared files at each peer.

Figure 8(a) depicts the CDF of the degree of change for 3000 randomly selected peers over four different timescales: 2 hours, 6 hours, 1 day and one week. This figure reveals that 36% of monitored peers do not change their sharing lists over a one week interval. However, that number increases to 69%, 80% and 90% over one day, 6 hour and 2 hour intervals, respectively. On the other hand, 90% of peers change less than 10 files over a 6 hour period, less than 25 files over a one day period, and less than 80 files over a week. Given the average number of shared files by each peer (which is around 350 from Table 2), this result indicates that the variations of shared files by individual peers is rather small especially over several days. While this finding is rather intuitive and expected, Figure 8(a) allows us to quantify the distribution of the degree of change across different timescales. Finally, this result implies that caching information about the available files at other peers (especially over the timescale of few days) can be a highly effective bandwidth-saving strategy in peer-to-peer systems.

6.2 Variations in Popularity of Individual Files

We focus on the effect of changes in shared files by each peer on the popularity of individual files across the system. Understanding the dynamics of file popularity can be used to determine how often the popularity of available files should be sampled. To eliminate the effects of a varying peer population across different snapshots, we define the *popularity* of a file as the percentage of successfully contacted peers with the file. Given the random distribution of files among peers, the popularity can be interpreted as the probability of having that



(a) Popularity change for top 100 files over different timescales

(b) Popularity change for top 1000 files over different timescales

Figure 9: Changes in file popularity

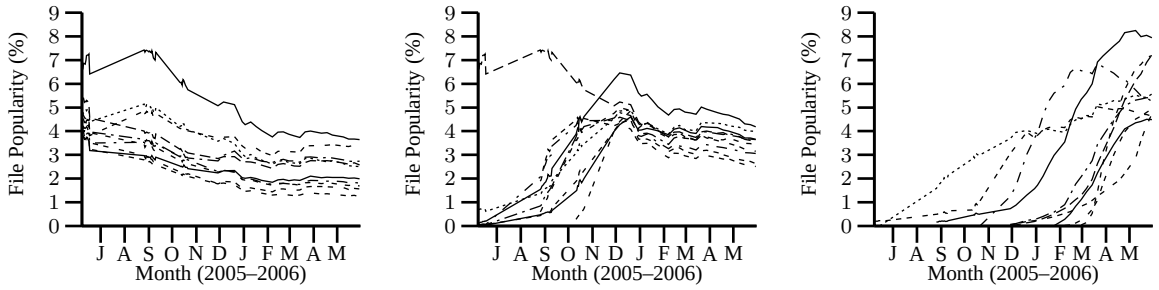
file at a random peer. We define the *change in popularity* of a given file over interval τ as the difference between its popularity at the beginning and the end of such an interval (in percentage points).

Figure 8(b) depicts the change in popularity of 50,000 randomly-selected files over a one day interval as a function of their popularity rank in a scattered-dot plot where each dot corresponds to a particular file. The population of the highest and lowest ranked files in this figure are roughly 28,000 and 130, respectively. This figure clearly demonstrates the effect of file popularity on the variations of its popularity over a one-day period. This figure shows that the most popular files (top 15,000) exhibit significantly larger variations (increase or decrease) in their popularity compared to the rest of the files. Note that the changes in popularity for most of the files are within 0.02 points. While the variations in file popularity rapidly drop with file rank among the top 15,000 files, it becomes relatively stable across the least popular files and remains below 0.01 points. In summary, a group of the most popular files experience wider variations in their popularity than unpopular files.

To study popularity dynamics in further detail, we focus on the top-100 and top-1000 most popular files in a particular snapshot and examine their popularity variations over different timescales. Figures 9(a) and 9(b) plot the CDF graph of the change in popularity for the top-100 and top-1000 files, respectively, for intervals of 1 week, 1 month, 2 months and 6 months. Figures 9(a) and 9(b) individually show that the range of variations in popularity of top-n files expanded with time. However, comparing these figures clearly illustrates that *(i)* for any given timescale, more popular files exhibit significantly larger variations in their popularity, and *(ii)* the popularity of more popular files changes more rapidly with time.

6.3 Long-term Evolution in Popularity

Another interesting question is whether the long-term evolution of file popularity follows a particular pattern, and what does that pattern look like? Intuitively, the popularity of a new file should gradually increase with



(a) 6/8/2005's Top 10 (b) 12/6/2005's Top 10 (c) 5/30/2006's Top 10

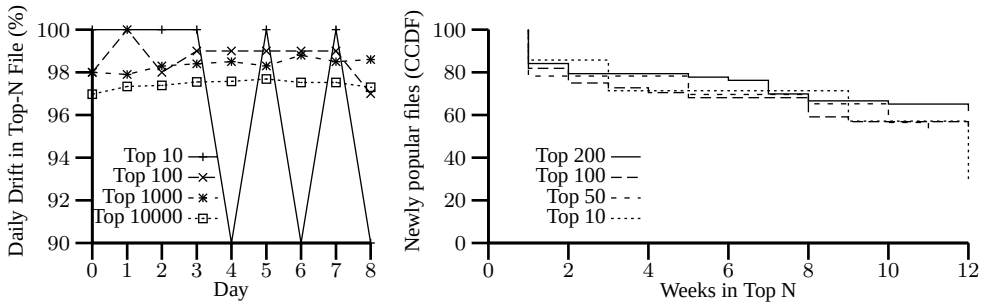
Figure 10: Changes in the identity of Top-N files

some average rate until it reaches its peak popularity. Then, it remains close to its peak popularity for a while before its popularity gradually drops with time. However, the rate of change in popularity, the peak popularity and the time that a file remains around its peak popularity could significantly vary across different files.

To examine the pattern of long-term evolution in file popularity we tracked the pattern of changes in the popularity of top-10 files in each snapshot across all other snapshots over our one-year measurement period. We examine the pattern of long-term evolution in Top-10 files for the first, middle and last snapshots during our measurement periods [6]. Figure 10(a) shows the evolution of the popularity in the Top-10 file in our first snapshot (captured in June 8, 2005) during the one year interval *after* they appear in the Top-10 list. Despite the fluctuations in the popularity of individual files, popularity of all files exhibit an overall downward trend. Figure 10(c) depicts the evolution of popularity for the Top-10 files in our last snapshot during the one year period *before* they appear in the Top-10 list. This figure shows that most files transition from very low popularity to Top-10 list in the span of a few months. Some of these files experienced a rather long periods of relative obscurity before making rapid gains in popularity. However, two files transitioned gradually from obscurity to popularity over the course of the year. Also note that two of the Top-10 files in the last snapshot appears to have passed their peak popularity. Figure 10(b) provides a third point of view, examining the pattern of evolution in the popularity of Top-10 files in the middle of our measurement window. This figure illustrates how the popularity of Top-10 files have changed during a six month period before and after they become Top-10. This figure shows that one of the Top-10 files in this snapshot was more popular at the beginning of our measurement and has been gradually losing popularity, while the other Top-10 files are newcomers that have been gaining popularity during the same period. Afterward, they all gradually lose their popularity over the following six month period.

In summary, these figures show that files rise in popularity at varying (and often rapid) rates until their reach

[6]The pattern of long-term evolution in the popularity of Top-10 files for all other snapshots are available online at http://mirage.cs.uoregon.edu/P2P/files/



(a) Drift in the popularity of top-N files (b) Distribution of how long files remain in the top-N

Figure 11: Distribution of how long files remain in the top-N

their peak popularity. Then they enter a period of slow decay where their popularity gradually drops. The Top-10 most popular files at each snapshot is composed of three groups of files: *(i)* files whose popularity is still rising towards their peak, *(ii)* files whose popularity is around their peak, and *(iii)* files whose popularity is dropping.

6.4 Variations in Top-N Files

We examine how the changes in file popularity affects the identity of the Top-n files in the system over both short and long timescales. In essence, this explores the rather subtle relationship between the popularity and the ranking of Top-N files. Figure 11(a) depicts the percentage of the top-N files on day $x-1$ (starting at 06/08/2005) that remain in the top-N files on day x (*i.e.*, daily drift in the top-N list) for four values of N. Note that the y-axis begins at 89%, indicating that the top-N list is highly stable from one day to the next. The changes in the Top-N list are more visible for smaller values of N mainly due to two reasons: *(i)* the sensitivity to the small number of files (one change in the list results in $\frac{1}{N}$ variations) coupled with *(ii)* the noisy variations in the popularity of individual files that can easily change a couple of files at the bottom of the list. The main conclusion from this figure is that the identity of the top-N list remains highly stable across consecutive days for different values of N. Its minor variations is inversely proportional to N.

We also investigate the stability the top-N list over longer time scales, by examining how long files remain in the top-N. We use the "create-based method" [10] to avoid bias towards files that remain in the top-N files for a short period of time. Toward this end, we divide our measurement window into two halves. For any file that enters the top-N during the first half of the window, we measure the number of weeks that a file remains in the top-N. This technique provides us with unbiased measurements for files that remain for up to one half of the window (12 weeks). We can count how many files remain in the top-N for more than 12 weeks, but cannot accurately compute their distribution.

Figure 11(b) depicts the CCDF for the percentage of changes in the top-N files over time for different values

of N. This figure shows that approximately 20% of files remain in the top-N for only one week. However, approximately 60% of files that remain in the top-N list for at least one week, will continue to remain in the top-N for at least 10 weeks. More interestingly, the percentage of changes in the top-N files over time does not appear to have a strong dependency to N. Note that over a given period, the number of changed files in the Top-N list is proportional to N. Therefore, the *percentage* of changes in the Top-N files is relatively similar for different values of N.

7 Related Work

Several measurement studies have examined different properties of P2P file-sharing networks including: *(i)* dynamics of peer participations (*i.e.,* churn) [24, 26], *(ii)* overlay topology structure [4, 5, 27, 28], *(iii)* query traffic [6], *(iv)* data traffic [25, 29, 30], and *(v)* shared files [7, 8]. We are aware of only two other studies that focus on the characteristics of files shared by users. First, Fessant *et al.* [7] examined characteristics of available files, using data collected from 12,000 eDonkey clients over a three day period in 2003. They showed that the popularity of files stored in file-sharing systems is heavily skewed, following a Zipf distribution. When two peers have 10 files in common, there's an 80% chance they have at least one more file in common. The probability is close to 100% if they have at least 50 files in common. Second, Liang *et al.* [8] recently analyzed the nature and magnitude of deliberately corrupted files ("pollution") in Kazaa. To combat P2P sharing of copyrighted content, some companies intentionally inject decoy files, which have the same file name as a popular song. Liang *et al.* developed a multi-threaded crawler that queries all 30,000 Kazaa super-nodes for seven popular songs over the course of one hour. They showed that the popularity of different versions of a song also follows a Zipf distribution. For most of the seven popular songs, over 50% of the copies are polluted.

A few other studies have examined the files shared by users as part of broader measurement studies on peer-to-peer systems. In 2001, Chu *et al.* [9] studied peer churn and the distribution of file popularity. They found that file popularity follows a log-quadratic distribution (which can be thought of as a second-order Zipf distribution). Saroiu *et al.* [10] examined many characteristics of peers in Napster and Gnutella, such as their bottleneck bandwidth, latency, uptime, and number of shared files in 2001. They found that the number of shared files was heavily skewed.

Our study differs from the few previous studies on the shared files in P2P systems in at least two ways. First, we used recent and accurate snapshots of the Gnutella network with a significantly larger population of peers (*i.e.,* more than a million concurrent peers). Second, we presented two properties of the shared files that have not

previously been studied: *(i)* the correlation between shared files among peers and the overlay topology structure, and *(ii)* the variations in popularity of shared files across participating peers over time.

Another group of studies passively captured P2P traffic at a router to characterize exchanged files among peers. Gummadi *et al.* [29] analyzed a 200-day trace of Kazaa traffic collected at the University of Washington, demonstrating that file transfers in Kazaa do not follow a Zipf distribution and argued that this difference is due to the "fetch-at-most-once" nature of downloads in file-sharing applications. Another analysis of Kazaa traffic was conducted by Leibowitz *et al.* [25] at a large Israeli ISP. They examined the changing popularity of exchanged files among peers and showed that the data-sharing graph exhibits small-world properties. Note that the pattern of exchanged files among peers affects the characteristics of shared files throughout the system, but is subject to shorter-term trends. In contrast, the files shared by a peer may be the result of transfers over the course of months or years, followed by a gradual pruning of unwanted files. In summary, these studies on exchanged files are closely related and complement our work.

8 Conclusion

This paper presented a measurement-based characterization of available files in the Gnutella file sharing application. We discussed the challenges in capturing an accurate snapshot of available files in P2P file-sharing applications, and then presented two new measurement methodologies to achieve this goal. We used our parallel crawl and our peer sampler to obtain fairly accurate snapshots of available files among peers in the Gnutella network along with the connectivity among peers. Using these snapshots, we conducted three types of analysis and provided a better understanding of the distribution, correlation and dynamics of available files throughout the system.

We plan to continue this work in the following directions: We continue to collect many more snapshots to characterize properties of Gnutella files over longer timescales. Furthermore, we plan to further examine the models that properly capture each characteristics of available files in Gnutella. These models can be used in simulation based evaluations of file sharing applications. Finally, we are in the process of making our dataset publicly available to other researchers.

References

[1] S. Zhao, D. Stutzbach, and R. Rejaie, "Characterizing Files in the Modern Gnutella Network: A Measurement Study," in *Multimedia Computing and Networking*, (San Jose, CA), Jan. 2006.

[2] "slyck.com." `http://www.slyck.com`, 2005.

[3] S. Saroiu, K. P. Gummadi, R. J. Dunn, S. D. Gribble, and H. M. Levy, "An analysis of Internet content delivery systems," in *Symposium on Operating Systems Design and Implementation*, pp. 315–327, 2002.

[4] D. Stutzbach, R. Rejaie, and S. Sen, "Characterizing Unstructured Overlay Topologies in Modern P2P File-Sharing Systems," in *Internet Measurement Conference*, pp. 49–62, (Berkeley, CA), Oct. 2005.

[5] 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.

[6] A. Klemm, C. Lindemann, M. Vernon, and O. P. Waldhorst, "Characterizing the Query Behavior in Peer-to-Peer File Sharing Systems," in *Internet Measurement Conference*, (Taormina, Italy), Oct. 2004.

[7] F. L. Fessant, S. Handurukande, A.-M. Kermarrec, and L. Massoulie, "Clustering in Peer-to-Peer File Sharing Workloads," in *International Workshop on Peer-to-Peer Systems*, 2004.

[8] J. Liang, R. Kumar, Y. Xi, and K. W. Ross, "Pollution in P2P File Sharing Systems," in *INFOCOM*, (Miami, FL), Mar. 2005.

[9] J. Chu, K. Labonte, and B. N. Levine, "Availability and Locality Measurements of Peer-to-Peer File Systems," in *ITCom: Scalability and Traffic Control in IP Networks II Conferences*, July 2002.

[10] S. Saroiu, P. K. Gummadi, and S. D. Gribble, "Measuring and Analyzing the Characteristics of Napster and Gnutella Hosts," *Multimedia Systems Journal* **9**, pp. 170–184, Aug. 2003.

[11] D. Stutzbach and R. Rejaie, "Capturing Accurate Snapshots of the Gnutella Network," in *Global Internet Symposium*, pp. 127–132, (Miami, FL), Mar. 2005.

[12] D. Stutzbach, R. Rejaie, N. Duffield, S. Sen, and W. Willinger, "Sampling Techniques for Large, Dynamic Graphs," in *Global Internet Symposium*, (Barcelona, Spain), Apr. 2006.

[13] D. Stutzbach, R. Rejaie, N. Duffield, S. Sen, and W. Willinger, "On Unbiased Sampling for Unstructured Peer-to-Peer Networks," Tech. Rep. CIS-TR-06-07, University of Oregon, 2006.

[14] A. Rasti, D. Stutzbach, and R. Rejaie, "On the Long-term Evolution of the Two-Tier Gnutella Overlay," in *Global Internet*, (Barcelona, Spain), Apr. 2006.

[15] A. Fisk, "Gnutella Dynamic Query Protocol v0.1." Gnutella Developer's Forum, May 2003.

[16] Gnutella Developer Forum, "Browse Host Extension." `http://www.the-gdf.org/wiki/index.php?title=Browse_Host_Extension`.

[17] "Gnutella Developer Forum." `http://www.the-gdf.org/`, 2005.

[18] D. Stutzbach and R. Rejaie, "Evaluating the Accuracy of Captured Snapshots by Peer-to-Peer Crawlers," in *Passive and Active Measurement Workshop, Extended Abstract*, pp. 353–357, (Boston, MA), Mar. 2005.

[19] S. Chib and E. Greenberg, "Understanding the Metropolis–Hastings Algorithm," *The Americian Statistician* **49**, pp. 327–335, Nov. 1995.

[20] W. Hastings, "Monte Carlo Sampling Methods Using Markov Chains and Their Applications," *Biometrika* **57**, pp. 97–109, 1970.

[21] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller, "Equations of State Calculations by Fast Computing Machines," *Journal of Chemical Physics* **21**, pp. 1087–1092, 1953.

[22] Free Peers, Inc., "BearShare Network Statistics." `http://www.bearshare.com/stats/`, Oct. 2005.

[23] E. Adar and B. A. Huberman, "Free riding on gnutella," *First Monday* **5**, Oct. 2000.

[24] D. Stutzbach and R. Rejaie, "Characterizing Churn in Peer-to-Peer Networks," Tech. Rep. CIS-TR-2005-03, University of Oregon, Eugene, OR, May 2005.

[25] N. Leibowitz, M. Ripeanu, and A. Wierzbicki, "Deconstructing the Kazaa Network," in *WIAPP*, 2003.

[26] R. Bhagwan, S. Savage, and G. Voelker, "Understanding Availability," in *International Workshop on Peer-to-Peer Systems*, 2003.

[27] D. Stutzbach and R. Rejaie, "Characterizing the Two-Tier Gnutella Topology," in *SIGMETRICS, Extended Abstract*, (Banff, AB, Canada), June 2005.

[28] L. A. Adamic, R. M. Lukose, B. Huberman, and A. R. Puniyani, "Search in Power-Law Networks," *Physical Review E* **64**(46135), 2001.

[29] K. P. Gummadi, R. J. Dunn, S. Saroiu, S. D. Gribble, H. M. Levy, and J. Zahorjan, “Measurement, Modeling, and Analysis of a Peer-to-Peer File-Sharing Workload,” in *SOSP*, 2003.

[30] S. Sen and J. Wang, “Analyzing Peer-To-Peer Traffic Across Large Networks,” *IEEE/ACM Transactions on Networking* **12**, pp. 219–232, Apr. 2004.