IEEE DISTRIBUTED SYSTEMS ONLINE 1541-4922 © 2005 Published by the IEEE Computer Society
Vol. 6, No. 6; June 2005

Free Riding on Gnutella Revisited: The Bell Tolls?

Daniel Hughes, *Lancaster University*
Geoff Coulson, *Lancaster University*
James Walkerdine, *Lancaster University*

A new analysis of free riding on the Gnutella network updates data from 2000 and points to an increasing downgrade in the network's overall performance and the emergence of a "metatragedy" of the commons.

Individuals who use peer-to-peer (P2P) file-sharing networks such as Gnutella[1] face a social dilemma. They must decide whether to contribute to the common good by sharing files or maximize their personal experience by free riding, downloading files while not contributing any to the network. Individuals gain no personal benefits from uploading files (in fact, it's inconvenient), so it's "rational" for users to free ride. However, significant numbers of free riders degrade the entire system's utility, creating a "tragedy of the digital commons."[2]

Eytan Adar and Bernardo Huberman published an extensive study that traced Gnutella August 2000 user traffic over 24 hours.[3] Their study contradicted the then-orthodox view that user participation and hence communication in P2P file-sharing systems is symmetrical. It also suggested a number of techniques that developers could use to discourage free riding.

Over 100 research papers have cited this report, even in recent publications. However, four years is a long time for P2P research, a field that is only five years old. So, we decided to revisit and expand the 2000 study of Gnutella usage. We found that free riding has increased significantly since 2000. Furthermore, we believe that a "*meta*tragedy of the commons" has now emerged, wherein, to maximize their share of the communal Gnutella user base, rational Gnutella developers choose not to implement anti-free riding measures.

The 2000 study

The 2000 Gnutella study reported three main findings: a significant amount of free riding exists, free riding is uniform across different IP domains and connection speeds, and a peer can effectively be a free rider even if it shares many files.

A significant amount of free-riding occurs

To gauge the prevalence of free riding, the 2000 study analyzed `Pong` and `QueryHit` messages (see the "Gnutella 0.4" sidebar for more information about Gnutella messages). The study found that 66 percent of peers shared no files at all, while 73 percent shared 10 or fewer files. Additionally, Adar and Huberman observed that a very small proportion of the peers are responsible for the vast majority of the sharing: the top 1 percent of sharing peers accounted for 47 percent of all `QueryHits`, and the top 25 percent of these peers provided 98 percent of the `QueryHits`.

Free-riding is uniform across domains

To characterize free riders, the study performed two analyses. The first analyzed logged `Pongs` to determine if free riding occurred more among peers in particular IP domains. The answer was no. The report noted a linear relationship between the number of peers in a domain and the number of files that the domain as a whole claims to make available to the network. The second analysis plotted the number of `QueryHits` each domain generated against that domain's number of peers. This showed a similarly linear relationship between the number of peers in a domain and the number of files served. So, the study concluded that free riding was uniformly distributed across IP domains.

The study speculated that domains could represent bandwidth equivalency classes (for example, cable links typically connect peers on rr.com (http://rr.com) while narrowband links typically connect peers on aol.com, http://aol.com). From this, the study further concluded that free riding was uniformly distributed across connection speeds.

Peers that share files may still free ride

The 2000 study compared the number of files that peers advertised with the number of `QueryHits` they issued. This comparison showed numerous instances of peers claiming to share many files actually responding to very few `Query` messages. Although these peers offered files, their offerings were so unpopular with the general Gnutella community that they were de facto free riding. Furthermore, the study noted that the range of popular files was actually quite narrow. One percent of search terms accounted for 37 percent of total queries issued, while the top 25 percent of search terms accounted for 75 percent of all `Query` traffic.

Critique

The 2000 study offers a comprehensive analysis of the Gnutella network, based on a statistically sound sample of Gnutella user traffic. (See the "Related Work in P2P" sidebar for more information about other studies.) However, it makes two assumptions without sufficient validation:

First, we looked at its assumption that domains could represent bandwidth equivalency classes and the conclusion that a node's tendency to free ride is unrelated to connection speed. To verify this hypothesis, we analyzed the relationship between the connection speed nodes reported in `Pong` messages and the number of files the node makes available to the network.

Second, although the 2000 study briefly describes the concentration of Gnutella queries, it doesn't explain in detail the experiments used to gather this information, limiting the results' usefulness. We used natural-language processing tools[4] to perform a detailed analysis of `Query` traffic.

Changes to Gnutella

Since 2000 (see the "Gnutella 0.4" sidebar), the Gnutella protocol has adopted significant changes—primarily to improve its scalability. A loose coalition of developers working on the most popular Gnutella clients proposed these changes through the Gnutella Request for Comments.[5] The most significant changes since 2000 (that is, from version 0.4[2] to 0.6[6]) are as follows:

- ultrapeers and the Query Routing Protocol (QRP),
- `Pong` caching, and
- support for rich queries.

Ultrapeers and the QRP

Gnutella 0.4 had two scalability problems: flooding tended to unduly swamp the network, and TTL values in messages (introduced to alleviate the flooding effect) tended to reduce the number of peers that any given search reached before the TTL mechanism terminated the search, usually around 10,000.[7]

To alleviate these problems, Gnutella 0.6 introduces a new scheme that uses *ultrapeers* and *leaf nodes* to create a hierarchically structured Gnutella network. Peers with faster connections may elect to become ultrapeers, maintaining many connections to the Gnutella network simultaneously (and hence routing more traffic). Those peers with limited resources may join the network as leaf nodes, maintaining only a small number of network connections and typically not accepting incoming connections (which is the role of ultrapeers). As only ultrapeers typically respond to incoming `Ping` messages, this arrangement significantly reduces the network's level of `Ping` and `Pong` traffic.

Ultrapeers also proxy for leaf nodes, only forwarding queries to leaf nodes if it appears that the node can answer. This is supported by the QRP, which specifies that each leaf node should upload a vector to directly connected ultrapeers containing the file names that the leaf node is sharing. The ultrapeer filters incoming queries, so that leaf nodes only receive queries when their vector contains a matching file name.

`Pong` caching

`Ping` and `Pong` traffic comprised a significant proportion of traffic on Gnutella 0.4. To reduce this bandwidth consumption, Gnutellla 0.6 introduces *caching schemes* for `Pong` messages. These include the Gnutella `Pong` cache implementation and the `Ping` reduction scheme.[6]

These schemes share several commonalities. Peers store a periodically refreshed array of *n*`Pong` messages.

When such a peer receives an incoming `Ping` message, it responds with several `Pong` messages (typically 10) from its cache rather than forwarding the `Ping`. The peer chooses returned `Pongs` to represent peers distributed across the network. `Pong`-caching reduces the network's volume of traffic, and because each cache carries `Pongs` from peers across the network, responses tend to be more representative.

Support for rich queries

Gnutella 0.4 doesn't support searches based on metadata or search by universal resource names. HUGE, the Hash/URN Gnutella extension,[6] lets peers discover resources based on universal resource names within standard `Query` and `QueryHit` messages. A peer wishing to retrieve URN data places a prefix of the required data in a standard `Query` message (ignored by peers that don't support the HUGE protocol). This lets peers search by SHA1 (Secure Hash Algorithm 1) hash value, which supports *swarming,* downloads of the same file from multiple sources simultaneously.[6]

LimeWire's (http://www.limewire.com) meta-information search protocol lets peers associate metadata with queries and responses through extensible XML data, which contains separate schemas for different media types. This enables rich queries and, potentially, more accurate search term matching.

Using URNs in both schemes shouldn't affect the observed volume of search, response traffic, or amount of free riding.

Changes in usage patterns

Since 2000, several important changes affecting Gnutella users have occurred that could significantly impact free riding:

- copyright enforcement activities such as legal action against users sharing copyrighted files and antipiracy advertising campaigns,
- blocking of P2P services by ISPs, and
- access technology developments.

Copyright enforcement

Effective copyright law enforcement on systems such as Gnutella poses problems because of the expense involved in prosecuting a significant proportion of the user community. However, to the extent that free riding becomes prevalent, copyright law enforcement through prosecution becomes increasingly feasible (because few users share any files). Furthermore, copyright enforcement affects other people beyond those prosecuted; this activity increases the fear of prosecution and hence the perceived risk involved in sharing files. In fact, the probability of any user sharing files is likely inversely proportional to a function of the perceived detriment caused, forming the basis of a positive feedback loop (see figure 1).

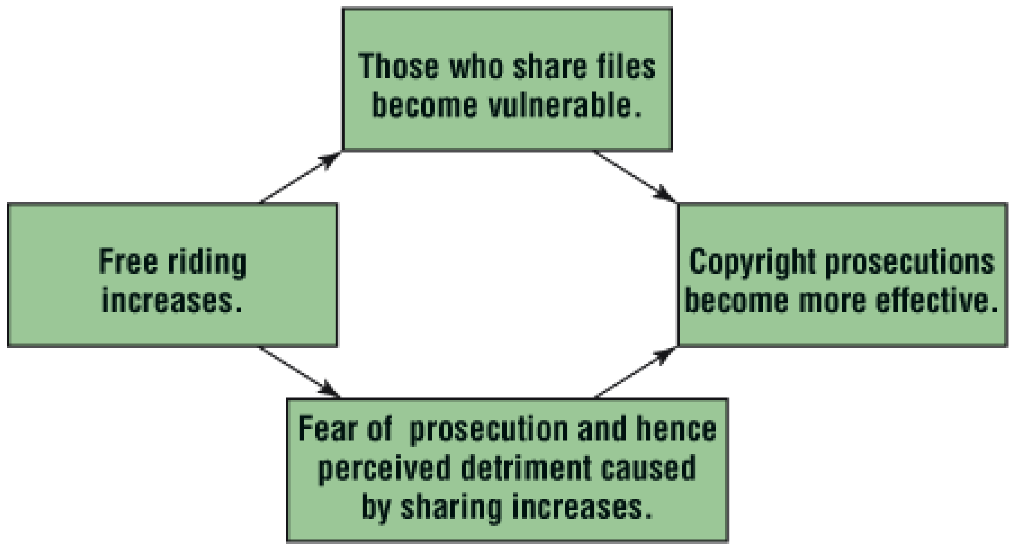


Figure 1. Copyright enforcement activity as a positive feedback system.

Our analysis of `Query` and `QueryHit` messages on Gnutella (see the "Our experimental results" section) reveals that a significant volume of queries target copyrighted materials and that a similar proportion of responses refer to copyrighted files. This, together with analysis of figure 1 indicates that the future looks bleak for those who wish to use Gnutella for sharing copyrighted media. Ironically, targeting peers sharing copyrighted media also causes problems for those who wish to download public-domain material. Currently, the same small set of servers predominantly serve public-domain and copyrighted media. If you remove these servers, it will drastically reduce the volume of public-domain and copyrighted material available.

Blocking P2P services

Many ISPs block access to P2P services because file-sharing traffic is potentially disruptive for other network services, and ISPs themselves come under legal attack for copyright violation. For example, the threat of legal action against academic institutions effectively persuaded such institutions to restrict access to P2P services.

P2P service blocking also occurs as a by-product of the increasing use of firewalls and network address translation. Peers behind a NAT can share files using the `Push` mechanism. However, if both the sharing peer and the downloading peer use NAT, file transfer is impossible.[1] In fact, as the number of NAT-based peers on the network increases, the number of such cases grows drastically (see figure 2).

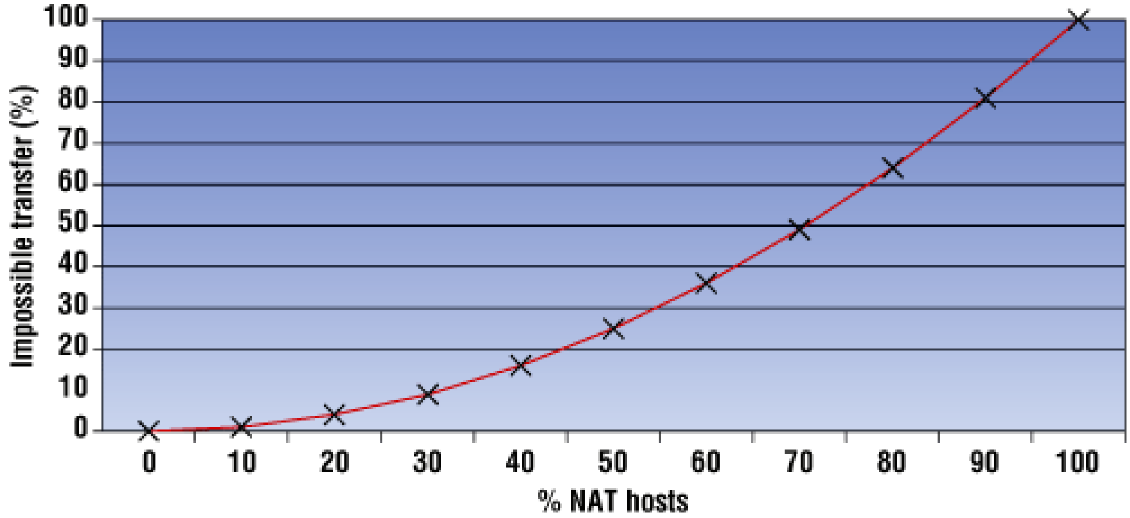


Figure 2. The effect of NAT (network address translation) on peer-to-peer transfer capability.

We observed 10 percent of peers reporting NAT addresses in `Ping` messages, twice that observed by the 2000 study. When only 10 percent of peers report NAT addresses, 1 percent of file transfers are impossible; however, as the number of peers using NAT rises to 50 percent, the proportion of impossible transfers rises to a much more significant 25 percent. So, the increasing use of NAT poses a worrying trend for the Gnutella network.

Anti-free riding schemes

Since 2000, the P2P research community has created several anti-free riding mechanisms, based on incentives. The Fasttrack network, for example, implements a reward-based scheme that ranks users according to the number of files they successfully upload to the network.

Bittorrent (http://bitconjurer.org/BitTorrent/protocol.html) takes enforced participation further, making uploading an intrinsic part of the protocol. In this scheme, download speed is throttled such that users providing more upload bandwidth receive faster downloads.

Another work suggests a punitive approach to discouraging free riding.[8] In this scheme, the system applies three punishment levels based on the free riding's observed severity:

- At the least punitive level, the scheme limits the propagation of messages sent from peers that download more than they upload.
- At the second level, the system may ignore searches that free riding peers generate.
- At the third level, the system may disconnect malicious or unproductive peers from the network.

MMAPS[9] seeks to establish a marketplace that lets users trade resources in a P2P environment. The marketplace is one situation in which social dilemmas produce cooperation in the real world. Such an environment discourages free riding because users must upload files to gain download credits.

In a specifically Gnutella context, the 2000 study suggested several ways to patch Gnutella against free riding, including automatic content caching as implemented in AGnuS[10] and enforced sharing of downloaded files. Interestingly, Gnutella has not implemented either of these measures or any of the others described.

Our experimental results

As each Gnutella peer participates in routing network messages and these messages subsume all network interactions, you can perform monitoring experiments simply by deploying a modified peer onto the Gnutella network to log samples of these messages. To this end, we developed a specialized peer based on the JTella (http://jtella.sourceforge.net) base classes. You can access these classes and associated tools on Lancaster University's P2P Web site (http://polo.lancs.ac.uk/p2p).

Protocol modifications such as those discussed in the previous section shouldn't affect our experiments, with the exception of those that involve `Pong` logging. Because only ultrapeers accept incoming connections, we expect to log fewer `Pong` messages on the Gnutella 0.6 network, so we'll need a longer trace for accurate statistics than Adar and Huberman used in their experiments. Accordingly, we performed a one-week monitoring session and verified the results with three additional 24-hour weekend traces. We maximized our sample base by connecting to the network as an ultrapeer and maintaining a large number of connections to both ultrapeers and leaf nodes. We compared our results directly against the 2000 study's findings.

Finding 1

Our results indicate that 85 percent of peers share no files and that 86 percent share 10 or fewer files. The 2000 study found that 66 percent of peers share no files, and 73 percent of peers share 10 or fewer, so free riding has increased significantly since then. Figure 3a illustrates that a small proportion of peers shares the vast majority of files and that most peers share no files.

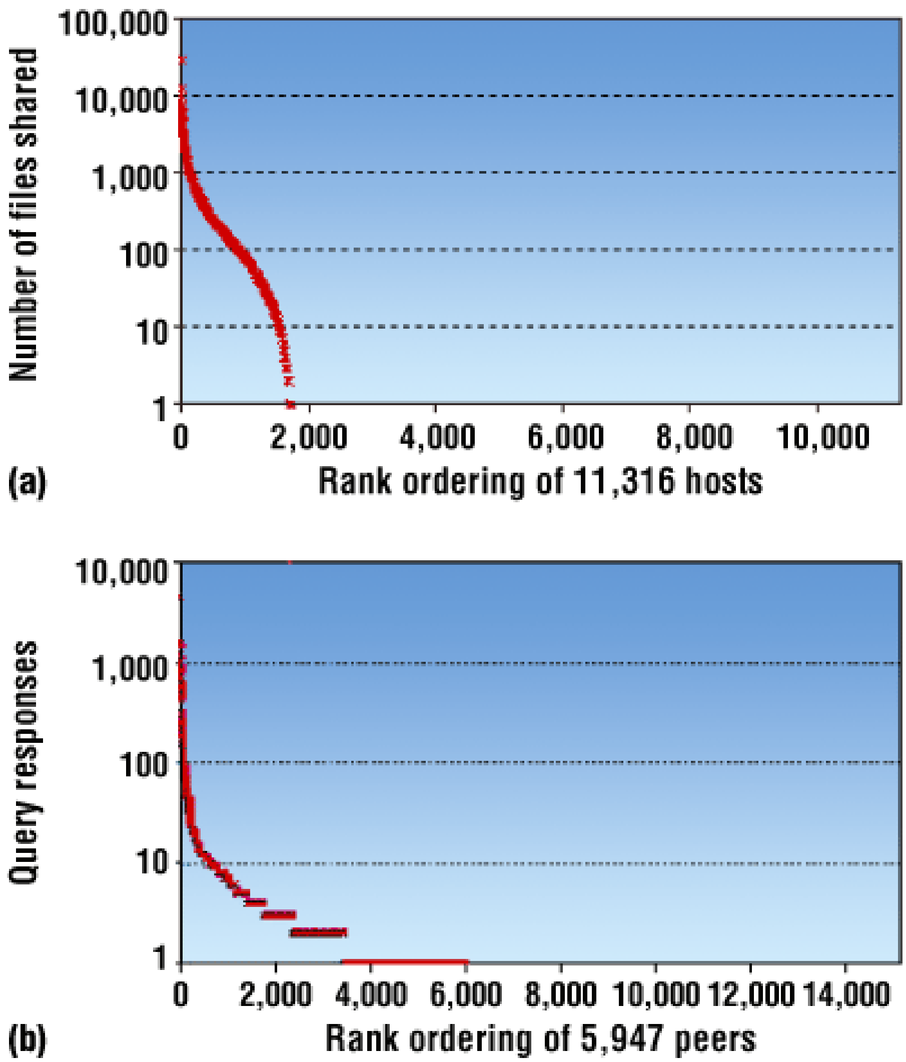


Figure 3. Rank ordering of peers (a) by number of files served and (b) by `QueryHits`.

Table 1 shows these results as separate figures for each of the three traces we carried out. The consistency of these traces gives us confidence that the results are typical and repeatable.

Table 1. Consistency of the three traces.

	Trace 1 (1 week)	Trace 2 (24 hours)	Trace 3 (24 hours)
Number of peers sharing no files	9,618 (85%)	1,104 (88%)	1,363 (86%)
Number of peers sharing 10 or fewer files	9,731 (86%)	1,142 (91%)	1,411 (89%)

We found that the top 1 percent of sharing peers provide 50 percent of all `QueryHits`, and the top 25 percent provide 98 percent. The 2000 study indicated that the top 1 percent of peers sending `QueryHit` messages were responsible for 47 percent of all `QueryHits`, and the top 25 percent of peers provided 98 percent. Figure 3b shows a rank ordering of peers based on the number of `QueryHits` issued over 24 hours. Although these results confirm Adar's overall findings, we observe that the situation has become more extreme. We hypothesize that this is due to increasing copyright enforcement activities and, to a lesser extent, P2P file-service blocking and the increasing use of NAT.

Finding 2

To determine if free riding is uniform across domains, we resolved our data into domains and top-level domains (discarding addresses that we couldn't easily resolve). Figures 4a and 4b show the results, which confirm Adar's findings, showing a reasonably linear relationship between a domain's number of peers and number of files available. This confirms that there is no evidence of free riding being more prevalent in some domains than others.

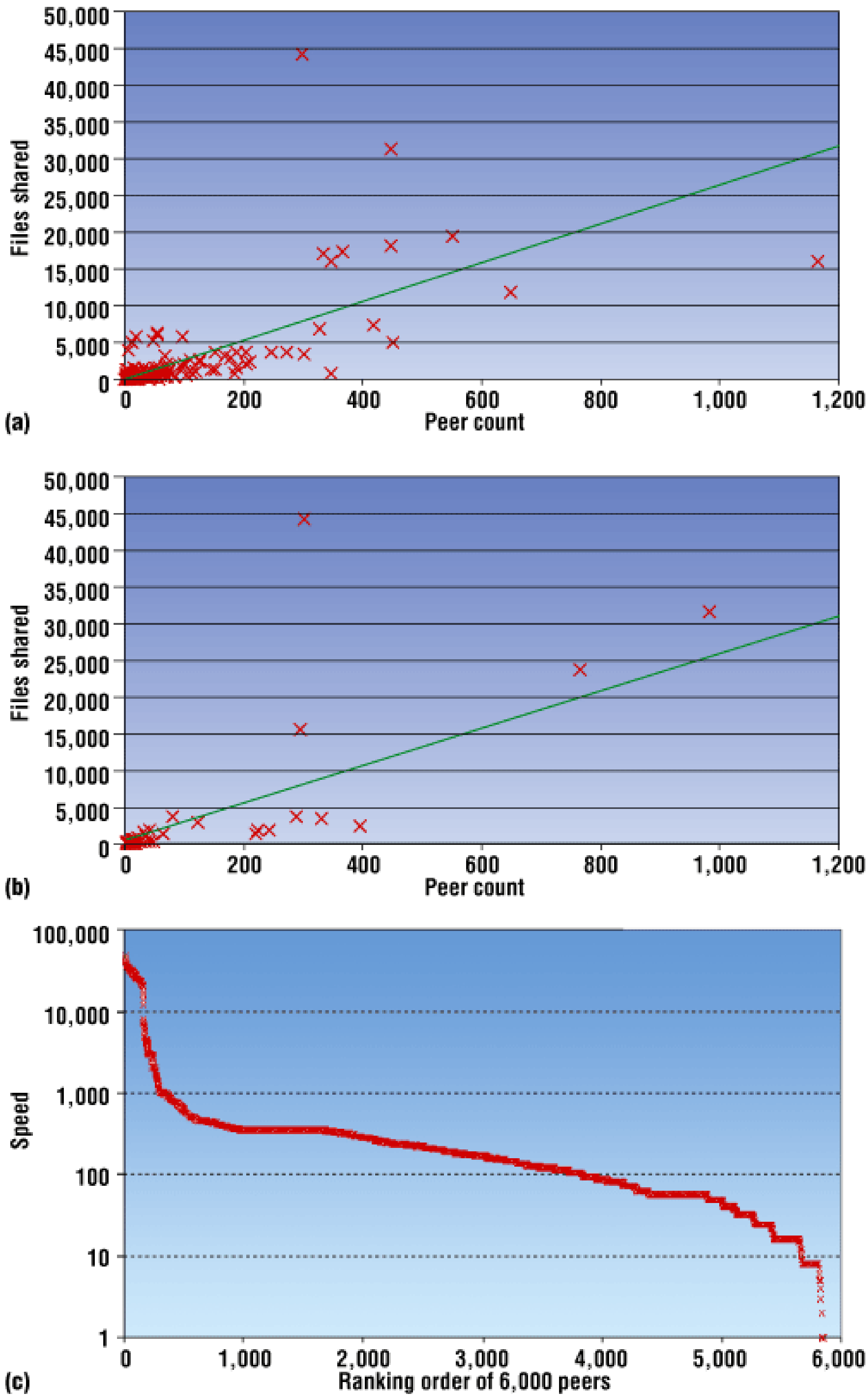

50,000
45,000
40,000
35,000
30,000
25,000
20,000
15,000
10,000
5,000
0
Files shared
0
200
400
600
800
1,000
1,200
Peer count
(a)
50,000
45,000
40,000
35,000
30,000
25,000
20,000
15,000
10,000
5,000
0
Files shared
0
200
400
600
800
1,000
1,200
Peer count
(b)
100,000
10,000
1,000
100
10
1
Speed
0
1,000
2,000
3,000
4,000
5,000
6,000
Ranking order of 6,000 peers
(c)

Figure 4. Analysis (a) by domain, (b) by top-level domain, and (c) by connection speed.

The 2000 study based the hypothesis that free riding is uniformly distributed across connection speeds on an assumption that domains could proxy for bandwidth. To investigate this more thoroughly, we first analyzed the network's reported peer speed. Figure 4c shows a rank ordering of 6,000 peers according to connection speed over 24 hours.

We then divided peers into bandwidth equivalency classes based on their reported speed and plotted this against the average number of `QueryHit` messages generated over 24 hours by nodes in each bandwidth class (see figure 5). The number of `QueryHits` that peers generated isn't independent of host speed as hypothesized by Adar, but varies across bandwidth classes. As you might expect, users on single-line ISDN links generate more `QueryHits` on average than users on dial-up links, and users on dual-line ISDN, cable, and ADSL links generate even more `QueryHits`. Counterintuitively, however, users on T1 or better connections typically generated fewer `QueryHits`, as Saroiu also observed.[11] A minority of users (2 percent) also report their bandwidth as unknown.

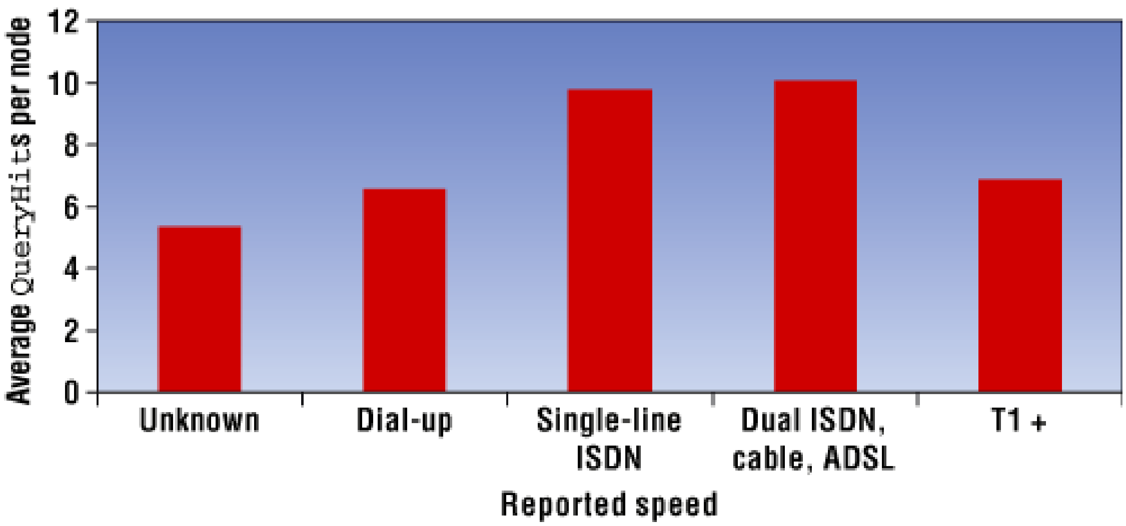


Figure 5. Peer speed against `QueryHits`.

So, our results don't support the idea that free riding is uniform across connection speeds. The initial investigation might have lacked detail; it didn't directly compare connection speed to the number of `QueryHits` generated or to changes in user behavior. However, reported bandwidth is somewhat unreliable. Saroiu discovered that up to 30 percent of nodes that report a low connection speed (single-line ISDN or lower) actually have significantly higher bandwidth and that 10 percent of nodes that report a high connection speed (T1+) actually have significantly lower bandwidth. However, Saroiu's direct measurements of peer connection speed also corroborate our findings.

Finding 3

The 2000 study found that the number of `QueryHits` a peer generates isn't proportional to the number of files the peer offers, as the bulk of queries concentrate on particular topics and only a small number of peers share popular files.

To investigate this, we recorded 25,000 search terms and performed word frequency analysis on them, isolating common phrases such as "star trek" as single items and removing stop words such as "and" and "the." We found that the top 1 percent of peers accounted for only 10 percent of `Query` messages and that 25 percent of peers accounted for 73 percent of `Query` traffic. The 2000 study found that 1 percent of search terms accounted for 37 percent of total queries, and the top 25 percent of search terms accounted for 75 percent of `Query` traffic. This might indicate that Gnutella users search for a broader range of material now than was observed in 2000; however, we could also attribute the difference to the different experimental methods used in our studies. Adar didn't describe the methods used to analyze search term popularity, so this is difficult to assess. Table 2 breaks down `Query` popularity.

Table 2. Search term popularity.

The top percentages	Total occurrences	As a percentage of the whole
1	62,215	10
5	129,755	23
10	165,121	48
15	186,237	61
20	198,763	68
25	208,683	73

Figure 6 shows a rank ordering of search terms. As you can see, this approximates a Zipf distribution (confirming the results reported elsewhere.[5])

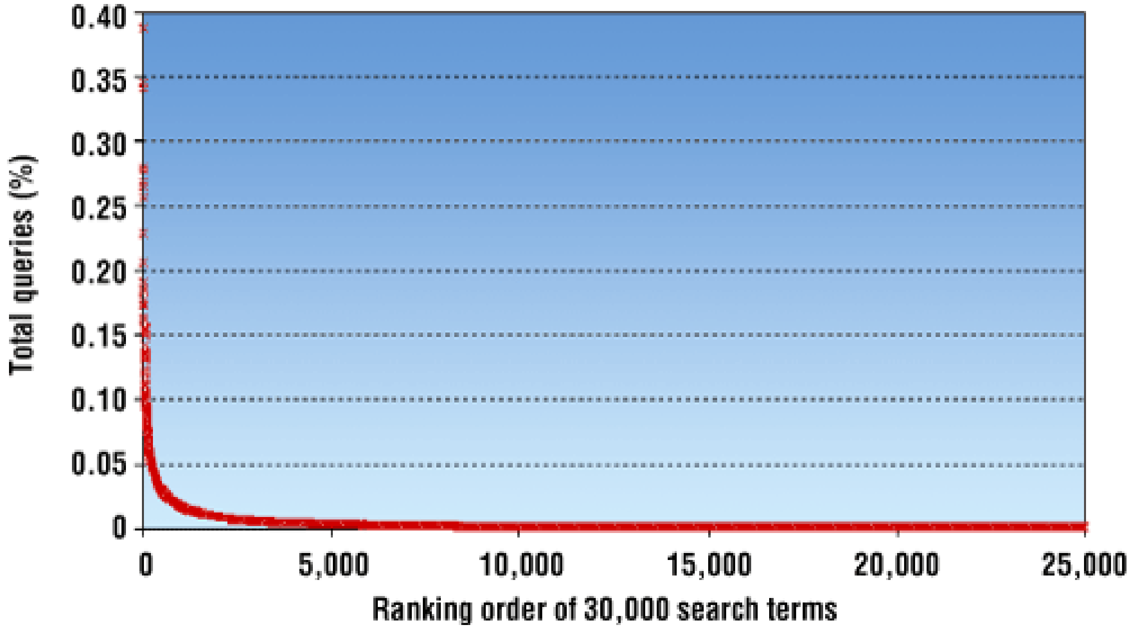


Figure 6. Rank ordering of search terms by percent of total search terms.

Fixing Gnutella

The 2000 study observed the difficulty of provoking spontaneous cooperation in anonymous groups and suggested that the "tragedy of the digital commons" might render useless systems that rely on spontaneous cooperation. Other work echoes this idea, including one that calls for the implementation of incentive schemes.[12] Our experiments show that free riding has increased significantly since 2000. Additionally, we believe that a positive feedback regime is in operation, the effect of which, along with the increasing use of NAT, is to progressively increase free riding on Gnutella. If left unchecked, the logical conclusion of both trends will be the Gnutella network's collapse.

Given the significant problems caused by free riding, modifying Gnutella to discourage free riding behavior (as recommended by Adar, Blake, and others) should be a high priority. However, no such modifications to the protocol have been made, despite significant research on incentive schemes. The Gnutella developer community has instituted other large-scale revisions to the Gnutella protocol, such as the ultrapeer scheme for scalability, but they haven't addressed the problem of free riding. We hypothesize that a metatragedy of the commons contributes to the Gnutella developer community's lack of action, arising from a confluence of the following factors:

- a loose coalition of developers working on popular Gnutella clients proposes Gnutella modifications;
- clients that implement schemes to encourage sharing make it difficult for users to free ride; and
- most users are free riders.

Consequently, client developers have little motivation to introduce anti-free riding measures because the user community would likely not adopt them—their introduction would result in a significant number of users migrating to clients that don't have anti-free riding schemes.

Gnutella developers benefit from individuals using their clients—for example, direct payment from commercial clients or a large user community able to contribute to the development effort in the case of open source clients. Either way, Gnutella developers share and even compete for users as a common resource. For example, consider the large-scale advertising of commercial Gnutella clients (such as LimeWire and BearShare, http://www.bearshare.com).

Developers must decide whether to address free riding by patching their clients and therefore reducing their user base or to maximize their user base by not implementing incentive measures. Thus, it's "rational" for developers wishing to create successful clients to not implement such measures even though this degrades the network's overall performance and, in the longer term, would reduce the total number of Gnutella users who would likely migrate to other P2P file-sharing networks.

So, two "tragedies" afflict Gnutella. First, the tragedy of the digital commons leads rational users to free ride to maximize their download efficiency. Second, a metatragedy leads Gnutella developers, who wish to maximize their user base, to not update their clients with incentive measures.

Conclusion

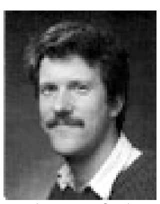
Gnutella remains unique among P2P file-sharing systems, both in being completely open and in having a large, established, and studied user base. We can draw valuable lessons from the falling participation level observed since 2000 and from Gnutella host developers' lack of response to this problem. In theory, you could address the metatragedy of the commons by introducing a central body to enforce implementation of necessary protocol changes. This seems fundamentally at odds with the P2P philosophy. Yet, finding an effective balance between maintaining the protocol's open nature and effectively managing its evolution could prove critical to the Gnutella protocol's long-term survival.

References

1. *The Gnutella Protocol Specification v0.4,*2000, http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf.
2. G. Hardin, "The Tragedy of the Commons," http://www.sciencemag.org/cgi/content/full/162/3859/1243, *Science,* vol. 162, no. 3859, 1968, pp. 1243–1248.
3. E. Adar and B. Huberman, "Free Riding on Gnutella," http://www.firstmonday.dk/issues/issue5_10/adar/index.html, *First Monday,* Oct. 2000.
4. P. Rayson and R. Garside, "Comparing Corpora Using Frequency Polling,"*Proc. Workshop Comparing Corpora,* Assoc. for Computational Linguistics, Oct. 2000, pp. 1–6.
5. K. Sripanidkulchai, "The Popularity of Gnutella Queries and Its Implications on Scalability," http://www-2.cs.cmu.edu/~kunwadee/research/p2p/gnutella.html.
6. *Gnutella 0.6,* RFC Gnutella 0.6, June 2002, http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html.
7. J. Ritter, "Why Gnutella Can't Scale, No Really,"Feb. 2001,

http://www.darkridge.com/~jpr5/doc/gnutella.html.
8. M. Karakaya, I. Korpeoglu, and O. Ulusoy, "A Distributed and Measurement-Based Framework against Free Riding in Peer-to-Peer Networks," http://doi.ieeecomputersociety.org/10.1109/PTP.2004.1334963 *Proc. 4th IEEE Int'l Conf. Peer-to-Peer Computing* (P2P 04), IEEE CS Press, 2004, pp. 276–277.
9. B. Strulo, A. Smith, and J. Farr, "An Architecture for Peer-to-Peer Economies," http://doi.ieeecomputersociety.org/10.1109/PTP.2003.1231528, *Proc. 3rd IEEE Int'l Conf. Peer-to-Peer Computing* (P2P 03), IEEE CS Press, 2003, p. 208.
10. D. Hughes, I. Warren, and G. Coulson, "AGnuS: The Altruistic Gnutella Server," http://csdl.computer.org/comp/proceedings/p2p/2003/2023/00/20230202abs.htm *Proc. 3rd IEEE Int'l Conf. Peer-to-Peer Computing* (P2P 03), IEEE CS Press, 2003, p. 202.
11. S. Saroiu, P. Gummadi, and S. Gribble, "Measuring and Analyzing the Characteristics of Napster and Gnutella Hosts," *Multimedia Systems,* vol. 9, no. 2, Springer-Verlag, 2003, pp. 170–184.
12. C. Blake and R. Rodrigues, "High Availability, Scalable Storage, Dynamic Peer Networks: Pick Two," http://www.usenix.org/events/hotos03/tech/full_papers/blake/blake_html *Proc. 9th Workshop Hot Topics in Operating Systems* (HotOS-IX), Usenix, 2003.

Daniel Hughes is a PhD research student at Lancaster University's Computing Department. His research interests include distributed systems and peer-to-peer systems. He received his master's degree in distributed interactive systems from Lancaster University. Contact him at Computing Dept., InfoLab 21, South Dr., Lancaster Univ., Lancaster LA1 4WA, UK; hughesdr@comp.lancs.ac.uk.



Geoff Coulson is a professor of computer science at Lancaster University's Computing Department. His main research interests are next-generation middleware and programmable networking. He received his PhD in computer science from Lancaster University. He is a member of the ACM and British Computer Society. Contact him at Computing Dept., InfoLab 21, South Dr., Lancaster Univ., Lancaster LA1 4WA, UK; geoff@comp.lancs.ac.uk.

James Walkerdine is a research associate at Lancaster University's Computing Department. His research interests include cooperative systems, information management, human-computer interaction, and peer-to-peer systems. He received his PhD in computer science from Lancaster University. He is a member of the British Computer Society. Contact him at Computing Dept., InfoLab 21, South Dr., Lancaster Univ., Lancaster LA1 4WA, UK; walkerdi@comp.lancs.ac.uk.

Cite this article: Daniel Hughes, Geoff Coulson, and James Walkerdine, "Free Riding on Gnutella Revisited: The Bell Tolls?" *IEEE Distributed Systems Online,* vol. 6, no. 6, 2005.

Gnutella 0.4

Gnutella is an open protocol that supports peer-to-peer resource discovery. The protocol builds an unstructured, decentralized overlay network,[1] in which each host must forward both resource discovery and network-maintenance messages. In this sidebar, we first discuss the base Gnutella protocol, Gnutella 0.4, which was prevalent when the 2000 study took place. In the main text, we discuss version 0.6, which includes significant changes. The protocol uses five message types:

- `Ping` is used in peer discovery. A peer receiving a `Ping` responds with a `Pong` message.
- `Pong` is a response to a `Ping`. It contains the responding peer's address and the amount of data it serves.
- `Query` is a search message. If a peer receiving a `Query` has matching data, it generates a `QueryHit`.
- `QueryHit` is a response to a `Query`. It contains information required to acquire the requested data.
- `Push` enables the support of downloads from firewalled peers.

Additionally, the protocol has three phases: connecting to the network, discovering resources, and transferring resources.

Network connection

Acquiring an initial host address, used to bootstrap network entry, occurs outside the Gnutella protocol, typically via a GWebCache (Gnutella Web-Caching System).[2] A newly arriving peer connects to a peer discovered in this way by initiating TCP connections to that host. Subsequently, a peer discovers further peers by sending a `Ping` message to directly connected peers. Each peer broadcasts, or *floods*, these messages to all neighboring peers. All peers that receive a `Ping` should respond with a `Pong`, which is forwarded back along the incoming `Ping`'s path to the originating peer. `Pong` messages contain the network address and port on which the sending peer is listening for incoming Gnutella connections as well as information from the peer regarding the amount of data and number of files available to the network. To avoid swamping the network, peers tag all messages with a *time-to-live* value, typically 7. Peers decrement a message's TTL value as it's routed, discarding a message when its TTL equals 0.

Resource discovery

Peers listen for incoming `Query` messages and broadcast them across the network by flooding them to each of their neighbors. If a peer can satisfy a `Query`, it sends a `QueryHit` message back along the same path. `QueryHit` messages contain the network address and port on which the responding peer is listening for HTTP file-transfer connections. `QueryHits` also include the peer's connection speed and a set of *hits* (matching file names) that satisfy this `Query`.

Resource transfer

File transfer occurs outside the Gnutella protocol. When a requesting peer receives a `QueryHit` message, it can attempt to initiate a direct download from the target peer (the `QueryHit` message specifies its port and IP address) via HTTP. However, if the target peer is behind a firewall, the requesting peer can instead send a `Push` message to the target, containing the requested file's details and the network address and port to push it to. On receiving a `Push`, the target peer establishes the HTTP connection and pushes the file to the requesting peer.

References

1. J. Walkerdine, L. Melville, and I. Sommerville, "Dependability Properties of P2P Architectures,"*Proc. 2nd IEEE Int'l Conf. Peer-to-Peer Computing* (P2P 02), IEEE CS Press, 2002, pp. 173–174.
2. *Gnutella 0.6,* RFC Gnutella 0.6, June 2002, http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html.

Related Work in P2P

Since 2000, many others have studied the Gnutella network—notably Stefan Saroiu and his colleagues[1] and Charles Blake and Rodrigo Rodrigues.[2] However, these studies differ significantly from the 2000 study in their focus.

Although Eytan Adar and Bernardo Huberman[3] focused specifically on user behavior in the context of sharing files, the Saroiu work attempts to more completely characterize the peers that constitute file-sharing networks by considering factors such as the bottleneck bandwidth between hosts, IP-level latencies, and the frequency of host disconnection and reconnection. In May 2001, Saroiu used a crawler to gather information about the Gnutella network by aggressively broadcasting `Ping` messages and logging the metadata in the resulting `Pong` messages. Unlike the Adar study, which assumes that domains represent bandwidth equivalency classes, Saroiu explicitly measures host bandwidth both as reported by the hosts and also by direct inspection, through which Saroiu discovered that a significant number of peers misreport their bandwidth.

The Blake study took place in 2003 and also analyzes Gnutella traffic. As you might expect, due to the deployment of the Gnutella 0.6 protocol, it discovered significantly different network characteristics from the Adar and Saroiu studies. This study examines in detail the extent to which peer-to-peer (P2P) systems can provide large-scale, reliable storage and, as such, focuses on churn rate and host capabilities. As with Adar and Saroiu, Blake traces Gnutella traffic; but unlike these studies, Blake doesn't perform a detailed analysis of users' sharing behavior. Our work focuses on a detailed analysis of a user's file-sharing behavior, rather than the low-level network factors that the Saroiu and Blake traces primarily addressed.

Other work[4] analyzes a long-term trace of traffic on the Kazaa file-sharing network and attempts to model this system's workload to inform caching schemes. This work helps you to understand the workload of large-scale, file-sharing systems; however, like the Blake study, it doesn't specifically address free riding. Similarly, another work studies file popularity and availability on Gnutella (and Napster) over one month but doesn't specifically address free riding.[5]

Generally, studies such as the ones we've mentioned provide information about real-world P2P workloads that you can use to more effectively evaluate P2P systems. Those studies that consider users' sharing behavior corroborate the Adar study's basic observation that free riding is a significant problem on Gnutella and, by implication, on other anonymous, decentralized P2P file-sharing systems. Unlike the Adar study or our work, these studies don't consider free riding's causes or analyze free rider demographics.

References

1. S. Saroiu, P. Gummadi, and S. Gribble, "Measuring and Analyzing the Characteristics of Napster and Gnutella Hosts,"*Multimedia Systems*, vol. 9, no. 2, Springer-Verlag, 2003, pp. 170–184.
2. C. Blake and R. Rodrigues, "High Availability, Scalable Storage, Dynamic Peer Networks: Pick Two," http://www.usenix.org/events/hotos03/tech/full_papers/blake/blake_html *Proc. 9th Workshop Hot Topics in Operating Systems* (HotOS-IX), Usenix, 2003.
3. E. Adar and B. Huberman, "Free Riding on Gnutella," http://www.firstmonday.dk/issues/issue5_10/adar/index.html *First Monday*, Oct. 2000.
4. K. Gummadi et al., "Measurement, Modelling and Analysis of a Peer-to-Peer File-Sharing Workload,"*Proc. 19th ACM Symp. Operating Systems Principles* (SOSP), ACM, 2003, pp. 19–22.
5. J. Chu, K. Labonte, and B. Levine, "Availability and Locality Measurements of Peer-to-Peer File Systems,"*Proc. ITCom: Scalability and Traffic Control in IP Networks II Conf.,* SPIE, vol. 4, 868, 2002.