

Free Riding on Gnutella Revisited: the Bell Tolls?

Daniel Hughes, Geoff Coulson, James Walkerdine
Computing Department, Lancaster University, Lancaster, UK

[hughesdr, geoff, walkerdi]@comp.lancs.ac.uk

'Freeriding' is a phenomenon in which peer-to-peer network users consume resources (e.g. download files) while not contributing any to the network. In this paper we analyse freeriding in the popular Gnutella network, revisiting a classic analysis from 2000.

1. Introduction

Since the release of Napster [1] in 1999, peer-to-peer computing has rapidly been growing in popularity. Napster is an example of a *semi-centralised* peer-to-peer system [2]. However, due to legal and scalability problems, semi-centralised peer-to-peer file sharing systems have largely been replaced by *fully-decentralised* file-sharing networks, such as Gnutella [3].

Peer-to-peer file-sharing networks like Gnutella embody a social dilemma wherein the behaviour of rational users is at odds with the common good. The dilemma for each individual is whether to contribute to the common good by sharing files, or to maximise their personal experience by 'freeriding' (i.e. downloading files while not contributing any to the network). As there is no personal benefit gained by uploading files (in fact it is inconvenient), it is 'rational' for users to free ride. However, significant numbers of free riders degrade the utility of the entire system. This is known as 'the tragedy of the digital commons' [6].

A frequently-cited paper from 2000 by Adar et al. [7] reports on an extensive study of user traffic on Gnutella. This study traced Gnutella user traffic over 24 hours in August 2000 and discovered that almost 70% of peers share no files at all, while 40% of all files are served by the top 1% of peers. The Adar study was important on two counts. First, it contradicted the then-orthodox view that user participation and hence communication in peer-to-peer file sharing systems is symmetrical. Second, it suggested a number of anti-free-riding techniques (discussed in section 4.2) that could be used to discourage free-riding.

The findings of the Adar study have been cited in over a hundred research papers and are still used in the most recent research [8] [9]. However, four years is a long time in the field of peer-to-peer research, a field which itself is only five years old. We therefore decided to provide an up-to-date snapshot of Gnutella usage. We hypothesised that freeriding has increased significantly since 2000, driven by factors such as increasing copyright enforcement and increasing restriction of access to peer-to-peer services. To verify this hypothesis, we revisited and extended the experiments performed in the Adar paper. We found that freeriding has increased significantly. Furthermore, as discussed in section 6, we believe that a '*meta* tragedy of commons' has now emerged, wherein rational Gnutella developers choose not to implement anti-freeriding measures in order to maximise their share of the communal Gnutella user-base.

The remainder of this paper is structured as follows: Section 2 provides background on the core Gnutella protocol and section 3 summarises Adar's findings as a base line for the present study. Subsequently, section 4 discusses changes that have occurred to the Gnutella protocol and its user environment since 2000 and section 5 describes our experiments and findings. Finally, section 6 discusses the future of the Gnutella network if observed trends continue, and section 7 concludes.

2. The Gnutella Protocol

Gnutella is an open protocol which supports peer-to-peer resource discovery. The protocol builds an unstructured decentralised overlay network [2]¹ in which each host is required to forward both resource discovery and network maintenance messages. The protocol uses just five message types as follows:

- **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 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** is a mechanism to support downloads from firewalled peers.

In addition, the protocol is structured into three phases: connecting to the network; discovering resources; and transferring resources.

Connecting to the Gnutella Network: Acquisition of an initial host address, used to bootstrap network entry occurs outside of the Gnutella protocol; typically via a 'GWebCache' [11]. A newly-arriving peer connects to a peer discovered in this way by initiating TCP connections to that host. Subsequently, further peers are discovered by sending a **PING** message to directly connected peers. These are broadcast or 'flooded' by each peer to all neighbour peers. All peers that receive a **PING** should respond with a **PONG** which is forwarded back along the path of the incoming **PING** to the originating peer. **PONG** messages contain the network address and port on which the sending peer is listening for incoming Gnutella connections; and information regarding the amount of data and the number files this peer is making available to the network. To avoid swamping the network, all messages are tagged with a 'time-to-live' (TTL) value, typically 7. Peers decrement the TTL value of messages as they are routed, discarding messages when the TTL equals 0.

Discovering Resources: Peers listen for incoming **QUERY** messages, and contribute to their broadcast across the network by flooding them to each of their neighbours. If a peer is able to satisfy a **QUERY**, it responds by sending 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. **QUERYHIT**s also include the speed of the peer's connection, and a set of 'hits' (matching file-names) which satisfy this **QUERY**.

Transferring Resources: File transfer occurs outside of the Gnutella protocol. When a requesting peer receives a **QUERYHIT** message, it can attempt to initiate a direct download, from the target peer (whose port and IP address were specified in the **QUERYHIT** message) via HTTP. However, if the target peer is behind a firewall, the requesting peer can instead send a **PUSH** message to the target, containing details of the file requested and the network address and port to which the file should be pushed. On receiving a **PUSH**, the target peer establishes the HTTP connection and pushes the file to the requesting peer.

3. The Gnutella Network in 2000

Adar [7] studied Gnutella user traffic over a 24 hour weekend period and reported three main findings: *i*) there was a significant amount of freeriding, *ii*) freeriding is uniform across different IP domains and

¹ This represents the base Gnutella protocol (Gnutella v0.4) which was the version that was prevalent when the Adar study was carried out. Since then it has changed significantly; see section 4.

connection speeds, and *iii*) a peer can effectively be a free rider even if it shares many files. We consider the three findings in more detail below.

3.1 Finding 1: A significant number of Gnutella users are free riders

To gauge the prevalence of freeriding, Adar analysed **PONG** and **QUERYHIT** messages. The quantitative finding was that *66% of peers were found to share no files at all, while 73% were found to share 10 or fewer files*. In addition, Adar observed that a very small proportion of the peers are responsible for the vast majority of the ‘work’: *the top 1% of sharing peers were responsible for 47% of all QUERYHITs, while the top 25% of these peers provided 98% of the QUERYHITs*.

3.2 Finding 2: Freeriding is uniform across IP domains and connection speeds

In an attempt to characterise free riders, Adar performed two analyses. The first analysed logged **PONGs** to determine if freeriding was more prevalent among peers in particular IP domains. It was discovered that the answer was no—there is 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** generated by each domain against the number of peers in that domain. This showed a similarly linear relationship between the number of peers in a domain and the number of files being served in that domain. The conclusion was *that freeriding is uniformly distributed across IP domains*.

Adar speculated that domains can be considered to represent bandwidth equivalency classes, (e.g. peers on rr.com are typically connected via cable links, while peers on aol.com are typically connected by narrowband links). From this he further concluded *that freeriding is uniformly distributed across connection speeds*.

3.3 Finding 3: A peer can effectively be a free rider even if it shares many files

Finally, Adar compared the number of files advertised by peers with the number of **QUERYHITs** issued by those same peers. It was found that there are numerous instances of peers which claim to share many files but which actually respond to very few **QUERY** messages. While these peers offer files, their offerings are so unpopular with the general Gnutella community that they are de-facto freeriding. Furthermore, it was found that the range of files that are popular among Gnutella users is actually quite narrow. In fact, *1% of search terms were found to account for 37% of the total queries issued, while the top 25% of search terms accounted for 75% of all QUERY traffic*.

3.4 Critique

The Adar study is a comprehensive analysis of the Gnutella network, based upon a statistically sound sample of Gnutella user traffic. The study, however, makes two assumptions which are not sufficiently validated:

Firstly, it assumes that domains can be considered as ‘bandwidth equivalency classes’, and based upon this assumption it concludes that the tendency of a node to free-ride is unrelated to connection speed. To verify this hypothesis, we specifically analysed the relationship between the connection speed reported by nodes in **PONG** messages and the number of files that node is making available to the network. This is described in section 5.2.

Secondly, while the Adar study briefly describes the concentration of Gnutella queries, the experiments used to gather this information are not explained in detail, limiting the usefulness of the results. We used natural language processing tools [14] to perform a detailed analysis of **QUERY** traffic. This analysis is described in section 5.3.

3.5 Related Work

Since 2000, a number of other studies of the Gnutella network have been performed, notably those by Saroiu [20] and Blake [21]. However, these studies differ significantly from Adar’s work in their focus:

While the Adar study focused specifically on the behaviour of users in the context of sharing files, the Saroiu work attempts to provide a more complete characterisation of the peers which compose file sharing networks by considering factors such as the bottleneck bandwidth between hosts, IP-level latencies, and the frequency of host disconnection and reconnection. Saroiu used a crawler to gather information about the Gnutella network by aggressively broadcasting **PING** messages and logging the meta-data contained 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 themselves and also by direct inspection, through which it was discovered that a significant number of peers misreport their bandwidth. Along with Gnutella, Saroiu also performed an analysis of the Napster file sharing network, although this is not of relevance in the context of the present work.

The Blake study also analyses Gnutella traffic and was performed in 2003. As one might expect, it discovered significantly different network characteristics to the Adar and Saroiu studies. In more detail, this study examines the extent to which peer-to-peer systems can provide large-scale reliable storage and, as such, focuses upon churn rate and the capabilities of hosts. As with Adar and Saroiu, Blake traces Gnutella traffic; however, unlike these studies, Blake does not perform a detailed analysis of the sharing behaviour of users.

Work by Gummadi et al [22] analyses a long-term trace of traffic on the Kazaa file sharing network, and attempts to model the workload of this system with the aim of informing caching schemes. This work is important in the wider context of understanding the work-load of large-scale file sharing systems; however, as with the Blake study, it does not specifically address the freeriding issue. Similarly, work by Chu et al [23] studies the popularity and availability of files available on Gnutella (and Napster) over a period of one month but does not specifically address freeriding.

Generally, studies such as the above are important because they provide information about real-world peer-to-peer workloads which can be used more effectively to evaluate peer-to-peer systems. Those studies that consider the sharing behaviour of users all corroborate Adar's basic observation that freeriding is a significant problem on Gnutella and, by implication, other anonymous decentralised peer-to-peer file sharing systems. Unlike the Adar study or the present work, none of these studies considers the *causes* of freeriding, or analyses the demographics of freeriders any detail.

4. Developments Since 2000

4.1 Changes to the Gnutella protocol

Since 2000, significant changes to the Gnutella protocol have been adopted—primarily to improve its scalability. Such changes been proposed by a loose coalition of developers working on the most popular Gnutella clients, and through the Gnutella RFCs [15]. The most significant changes since 2000 (i.e. from version 0.4 [3] to version 0.6 [11]) are as follows:

- Ultra-peers and the Query Routing Protocol (QRP);
- **PONG** caching;
- Support for rich queries

These are described below.

4.1.1 Ultra-peers and the Query Routing Protocol

Gnutella version 0.4 suffered from two scalability problems: *i*) the use of flooding tended to unduly swamp the network, and *ii*) the use of 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 [10].

To alleviate these problems, Gnutella 0.6 introduces a new scheme which uses the concepts of *ultra-peers* and *leaf-nodes* to create a hierarchically structured Gnutella network. Peers with faster connections may

elect to become ultra-peers, maintaining many connections to the Gnutella network simultaneously (and hence routing more traffic), while those with limited resources may join the network as leaf-nodes, maintaining only a small number of connections to the network and typically not accept incoming connections (which is the role of ultra-peers). As only ultra-peers (typically) respond to incoming **PING** messages, this arrangement significantly reduces the level of **PING** and **PONG** traffic on the network.

Ultra-peers also proxy for leaf-nodes, only forwarding queries to leaf-nodes if it appears that the node can answer. This is supported by the Query Routing Protocol (QRP) which specifies that each leaf-node should upload a vector to directly connected ultra-peers containing the names of the files that the leaf-node is sharing. Incoming queries are then filtered by the ultra-peer, so that leaf-nodes only receive queries when their vector contains a matching file name.

4.1.2 Pong caching

PING and **PONG** traffic formed a significant proportion of traffic on Gnutella 0.4. To reduce this bandwidth consumption, *caching schemes* for **PONG** messages have been introduced. These include the Gnutella **PONG** cache implementation and the **PING** reduction scheme [11].

These schemes share a number of commonalities. Peers store an array of n **PONG** messages which is periodically refreshed. When such a peer receives an incoming **PING** message, it responds with a number of **PONG** messages (typically 10) from its cache, rather than forwarding the **PING**. Returned **PONG**s are chosen such that they represent peers distributed across the network. The effect of **PONG** caching is two fold: *i*) it reduces the volume of traffic on the network; and *ii*) as each cache carries **PONG**s from peers across the network, responses tend to be more representative.

4.1.3 Support for rich queries

Gnutella 0.4 does not support searches based on meta-data or search by universal resource names. HUGE, the Hash/URN Gnutella extension [11], allows peers to discover resources based upon universal resource names (URNs) within standard **QUERY** / **QUERYHIT** messages. A peer wishing to retrieve URN data places a prefix of the required data in a standard **QUERY** message (this is ignored by peers that do not support the HUGE protocol). This allows for search by SHA1 hash value, which is used to support downloads of the same file from multiple sources simultaneously (swarming) [11].

LimeWire's meta-information search protocol allows meta-data to be associated with queries and responses through the use of extensible XML data which contains separate schemas for different media types. This enables rich queries and, potentially, more accurate matching of search terms.

The use of URNs in both schemes would not be expected to affect the observed volume of search or response traffic or the degree of freeriding.

4.2 The development of anti-free-riding schemes

Since 2000, a number of anti-free-riding mechanisms have been developed. Many of these schemes [4] [5] [16] are based on the notion of incentives. The Fasttrack network [4], for example, implements a reward-based scheme which ranks users based on the number of files they successfully uploaded to the network.

Bittorrent [5] 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.

Work by Karakaya et al. [8] suggests a *punitive* approach to discouraging free-riding. In this scheme, three levels of punishment are applied based upon the severity of freeriding observed:

1. At the least punitive level, the scheme limits the propagation of messages sent from peers which download more than they upload.
2. At the second level, searches generated by free-riding peers may be ignored.

3. At the third level, malicious or unproductive peers may be disconnected from the network.

MMAPS [17] seeks to establish a marketplace which allows users to trade resources in a peer-to-peer environment. The marketplace is one situation in which social dilemmas produce cooperation in the real world. Such an environment can potentially be used to discourage free-riding, as users must upload files in order to gain download 'credits'.

In a specifically Gnutella context, Adar suggests a number of ways to 'patch' Gnutella against freeriding, including automatic caching of content as implemented in AGnuS [19] and enforced sharing of downloaded files. Interestingly, Gnutella itself has never implemented either these measures or any of those described above. Possible reasons for this are discussed in section 6.

4.3 Changes in Gnutella usage patterns

A number of important changes affecting Gnutella *users* have occurred since 2000 that may significantly impact freeriding:

- Copyright enforcement activities such as legal action against users sharing copyrighted files and anti-piracy advertising campaigns;
- Blocking of peer-to-peer services by ISPs;
- Access technology developments.

4.3.1 The effect of copyright enforcement

Effectively enforcing copyright law on systems such as Gnutella is said to be problematic because of the expense involved in prosecuting a significant proportion of the user community. However, to the extent that freeriding becomes prevalent, enforcement of copyright law through prosecution becomes increasingly feasible (as few users share any files). Furthermore, the effects of copyright enforcement are not limited to those who are prosecuted; this activity also increases the fear of prosecution and hence the perceived risk involved in sharing files. In fact, it is likely that the probability of any user sharing files is inversely proportional to a function of the perceived detriment caused, forming the basis of a positive feedback loop, as visualised in Figure 1.

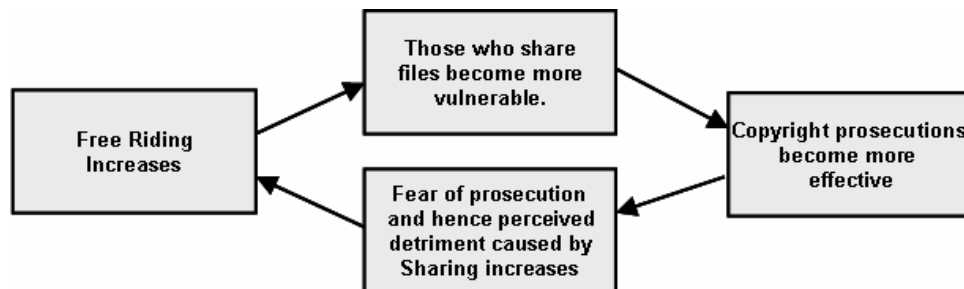


Figure 1 – Copyright Enforcement Activity as a Positive Feedback System

Our analysis of **QUERY** and **QUERYHIT** messages on Gnutella (see section 5.3) reveals that a significant volume of queries are targeted at 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 however, targeting peers sharing copyrighted media also causes problems for those who wish to download public domain material. Currently public domain and copyrighted media are predominantly served by the same small set of servers. If these servers are removed, the volume of public domain material available will be drastically reduced along with the volume of copyrighted material.

4.3.2 Blocking of peer-to-peer services

The blocking of access to peer-to-peer services by ISPs is widespread. There are two reasons for this: *i*) file sharing traffic is potentially disruptive for other network services; and *ii*) ISPs themselves come under legal

attack for copyright violation. For example, the threat of legal action against academic institutions has been effective in persuading such institutions to restrict access to peer-to-peer services.

The blocking of peer-to-peer services also occurs as a by-product of the increasing use of firewalls and network address translation (NAT). It is possible for peers behind a NAT to share files, using the 'push' mechanism described in section 2. However, if both the sharing peer and the downloading peer are using NAT, file-transfer is impossible [3]. In fact, as the number of NAT based peers on the network increase, the number of such cases grows drastically (see figure 2).

We observed 10% of peers reporting NAT addresses in **PING** messages, twice that observed by Adar in 2000. When only 10% of peers report NAT addresses, 1% of file-transfers are impossible; however, as the number of peers using NAT rises to 50%, the proportion of impossible transfers rises to a much more significant 25%. The increasing use of NAT is therefore a worrying trend for the Gnutella network.

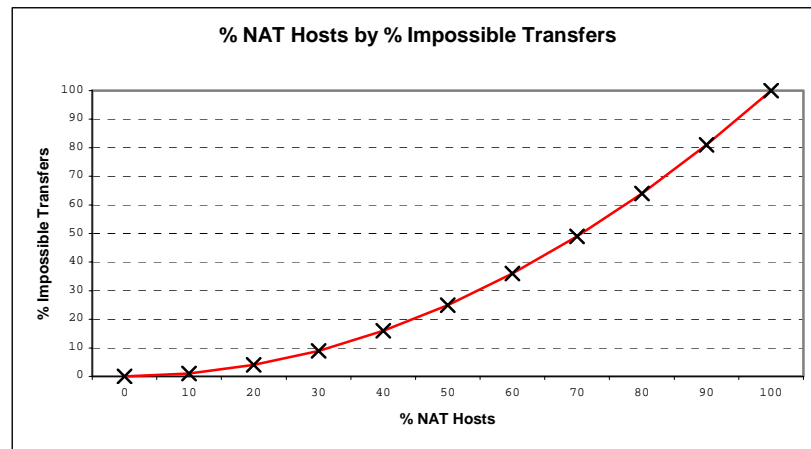


Figure 2 – The Effect of NAT on Peer-to-Peer Transfer Capability

5. Experimental Results

As each Gnutella peer participates in routing network messages, and these messages subsume all interactions on the network, monitoring experiments can be performed simply by deploying a modified peer onto the Gnutella network to log samples of these messages. To this end we developed a specialised peer based on the JTella base classes [12]. These classes and associated tools are available on Lancaster's peer-to-peer website [13].

The protocol modifications discussed in section 4.1 should not affect our experiments, with the exception of those that involve **PONG** logging: as only ultra-peers accept incoming connections, we expect to log fewer **PONG** messages on the Gnutella 0.6 network; therefore a longer trace will be required for accurate statistics than was used in the original Adar experiments. Accordingly, we performed a 1 week monitoring session, the results of which were verified by three additional 24 hour weekend traces. We maximised our sample base by connecting to the network as an ultra-peer, and maintaining a large number of connections to both ultra-peers, and to leaf-nodes.

5.1 Adar's finding 1: A significant number of Gnutella users are free riders

Our results indicate that 85% of peers share no files, and that 86% share 10 or fewer files (as opposed to Adar's original finding that 66% of peers share no files, while 73% of peers share 10 or fewer). Therefore freeriding has increased significantly since 2000. Figure 3a illustrates that the vast majority of files are shared by a small proportion of peers, while most peers share no files at all.

These results are broken down in Table 1 to show separate figures for each of the three traces we carried out. The consistency of these traces gives 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)	
	Percentage	Count	Percentage	Count	Percentage	Count
Peers sharing no files	85%	9618	88%	1104	86%	1363
Sharing 10 or less files	86%	9731	91%	1142	89%	1411

We found that the top 1% of sharing peers provide 50% of all **QUERYHITs**, while the top 25% provide 98% (as opposed to Adar’s finding that the top 1% of peers sending **QUERYHIT** messages were responsible for 47% of all **QUERYHITs**, while the top 25% of peers provided 98%). Figure 3b shows a rank ordering of peers based upon the number of **QUERYHITs** issued over a 24 hour period. While these results confirm Adar’s overall findings, we observe that the situation has become more extreme than in 2000. We hypothesise that this is due to increasing copyright enforcement activities (as discussed in section 4.3.1), and to a lesser extent the blocking of peer-to-peer file services and the increasing use of NAT (as discussed in section 4.3.2).

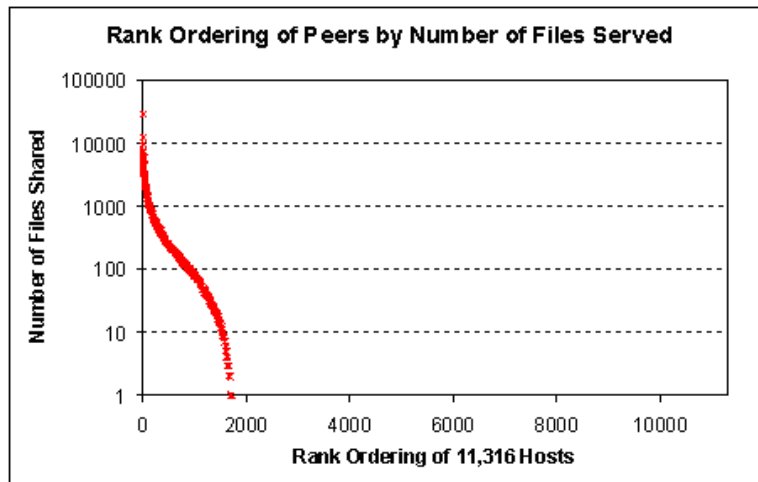


Figure 3a – Rank Ordering of Peers by Number of Files Served

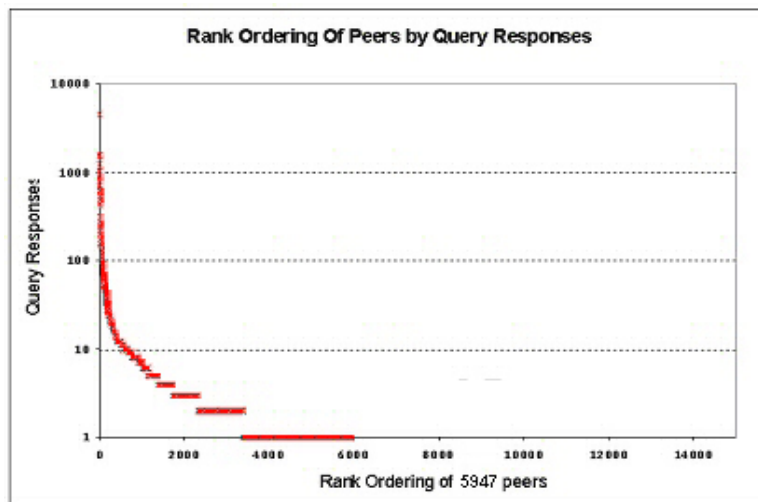


Figure 3b – Rank Ordering of Peers by **QUERYHITs**

5.2 Adar's finding 2: Freeriding is uniform across IP domains and connection speeds

To determine if freeriding is uniform across domains, we resolved our data into domains and top-level domains (discarding addresses which could not be easily resolved).

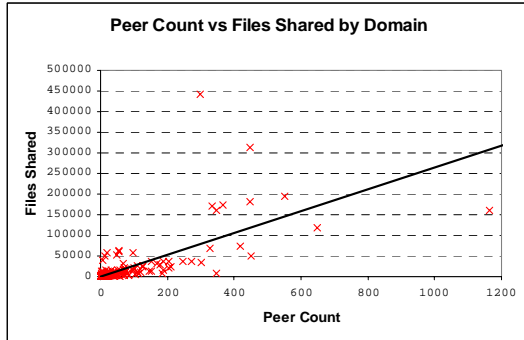


Figure 4a – Analysis by Domain

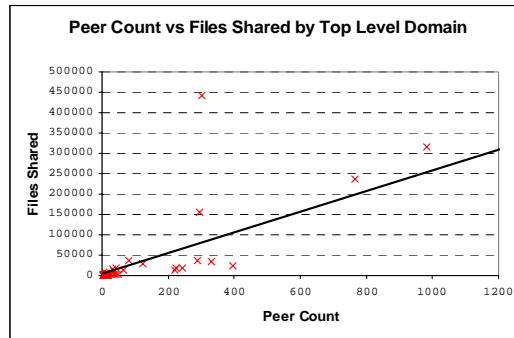


Figure 4b – Analysis by Top Level Domain

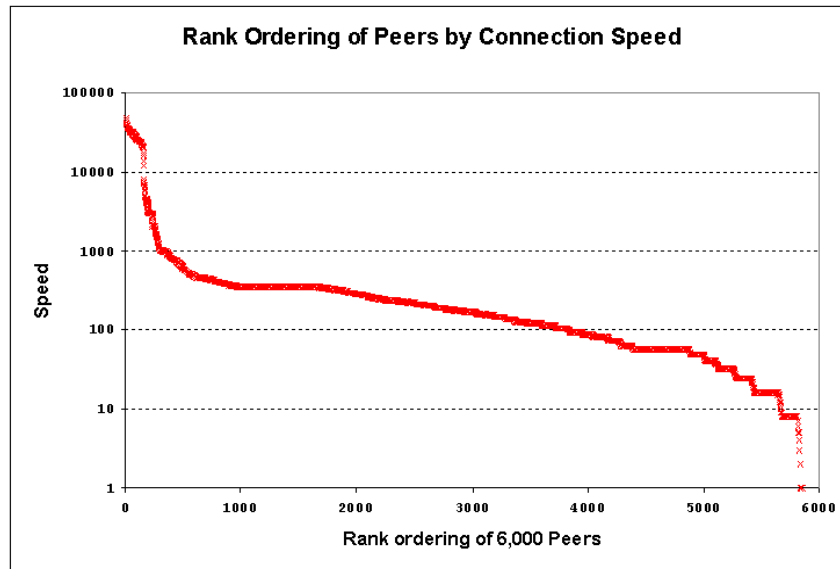


Figure 4c – Rank Ordering of Peers by Connection Speed

The results, shown in figures 4a and 4b, confirm Adar's findings, showing a reasonably linear relationship between the number of peers in a domain and the number of files available from that domain. This confirms that there is no evidence that freeriding is more prevalent in some domains than others.

We then considered Adar's hypothesis that freeriding is uniformly distributed across connection speeds: Adar's hypothesis was based on an assumption that domains could be considered to proxy for bandwidth. To investigate this more thoroughly, we first analysed the reported speed of peers on the network. Figure 4c shows a rank ordering of 6,000 peers based on connection speed gathered over 24 hours.

We then divided peers into bandwidth equivalency classes based upon their reported speed and plotted this against the average number of **QUERYHIT** messages generated in a 24 hour period by nodes in each bandwidth class (see figure 5). It can be seen that the number of **QUERYHITS** generated by peers is not independent of host speed as hypothesised by Adar, but varies across bandwidth classes. As one would expect, users on single-line ISDN links generate more **QUERYHITS** on average than users on dial up links, while users on dual-line ISDN, Cable and ADSL links generate even more **QUERYHITS**. Counter-

intuitively, however, users on T1 or better connections typically generated fewer **QUERYHITS**, as was also observed by Saroiu [20]. A small minority of users (2%) also report their bandwidth as unknown.

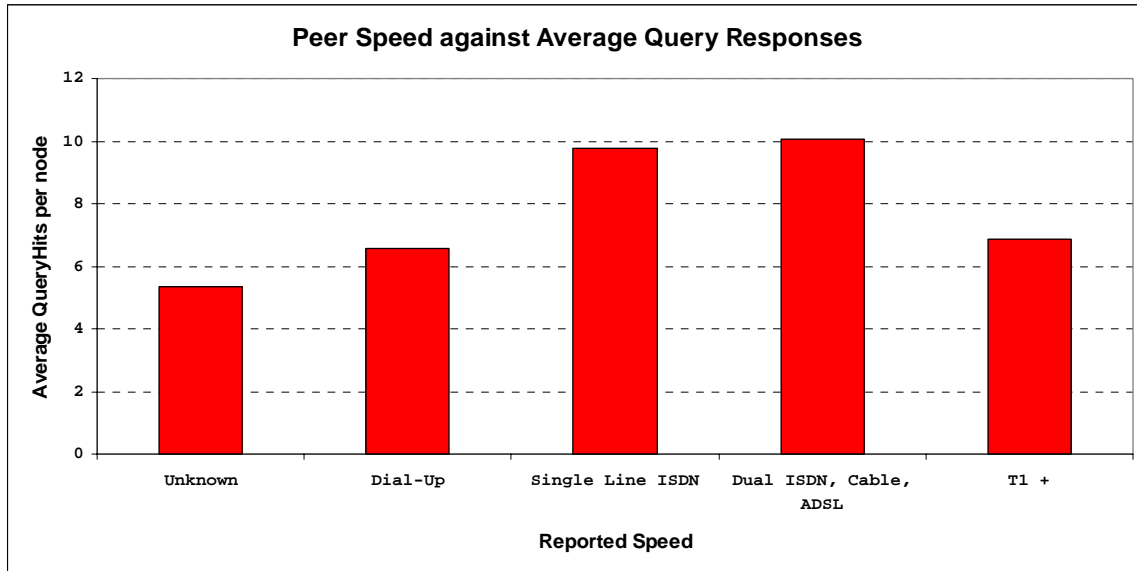


Figure 5 – Peer Speed against *QUERYHITS*

Our results thus contradict Adar’s assumptive hypothesis that freeriding is uniform across connection speeds. This may be due to a lack of detail in the initial investigation, which did not directly compare connection speed to the number of **QUERYHITS** generated, or to changes in user behaviour. However, it is important to consider that reported bandwidth is somewhat unreliable. Saroiu [20] discovered that up to 30% of nodes that report a low connection speed (single line ISDN or lower) actually have significantly *higher* bandwidth, and that 10% 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.

5.3 Adar’s finding 3: A peer can effectively be a free rider even if it shares many files

Adar found that the number of **QUERYHITS** generated by a peer is not proportional to the number of files the peer offers, as the bulk of queries are concentrated on particular topics, and that only a small number of peers share popular files.

To investigate this hypothesis, 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% of peers accounted for only 10% of **QUERY** messages while 25% of peers accounted for 73% of **QUERY** traffic (as opposed to Adar’s finding that 1% of search terms were found to account for 37% of total queries, while the top 25% of search terms accounted for 75% of **QUERY** traffic). This may indicate that Gnutella users search for a broader range of material now than was observed in 2000, however, the difference may also be due to the different experimental methods used in our studies. As Adar does not describe the methods used to analyse the popularity of search terms, this is difficult to assess. A breakdown of **QUERY** popularity is provided in table 2.

Table 2 – Popularity of Search Terms

The Top	Total Occurrences	As a Percentage of the whole
1%	62215	10%
5%	129755	23%
10%	165121	48%
15%	186237	61%
20%	198763	68%
25%	208683	73%

Figure 6 shows a rank ordering of search terms. As can be seen, this approximates a Zipf distribution (confirming the results reported in [15]).

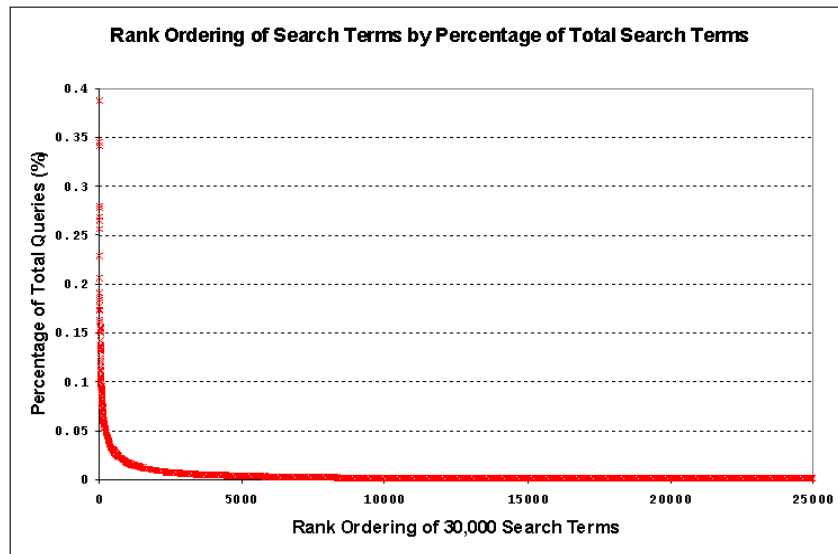


Figure 6 – Rank Ordering of Search Terms by % of Total Search Terms

6. Fixing Gnutella: a Meta-Tragedy of the Commons?

Adar observed that it is hard to provoke spontaneous cooperation in anonymous groups, and suggested that systems which rely on spontaneous cooperation may be rendered useless by the ‘tragedy of the digital commons’. This is echoed by other work, including [21] which calls for the implementation of incentive schemes. Our experiments demonstrate that freeriding has increased significantly since 2000. In addition, we believe that a positive feedback regime is in operation, the effect of which, together with the increasing use of NAT is to progressively increase freeriding on Gnutella. If left unchecked, the logical conclusion of both trends will be the collapse of the Gnutella network.

Given the significant problems caused by freeriding, it would seem that modifying Gnutella to discourage freeriding behaviour as recommended by Adar, Blake and others should be a high priority; yet no such modifications to the protocol have been made, despite significant research on incentive schemes as discussed in section 4.2. The Gnutella developer community has instituted other large-scale revisions to the Gnutella protocol, such as the ultra-peer scheme for scalability that was discussed in section 4.1, yet they have not taken action to address the problem of freeriding, the implications of which threaten to render questions over Gnutella’s scalability moot. We hypothesise that lack of action on the part of the Gnutella

developer community is due to a ‘meta-tragedy of the commons’ arising from a confluence of the following three factors:

1. Modifications to Gnutella are proposed by a loose coalition of developers working on popular Gnutella clients;
2. Clients that implement schemes to encourage sharing make it difficult for users to freeride;
3. Most users are free-riders.

The implication of these factors is that there is little motivation for client developers to introduce anti-freeriding measures, as these would likely not be taken up by the user community—their introduction would result in a significant number of users migrating to clients which do not implement anti-freeriding schemes.

Gnutella developers benefit from individuals using their clients. In the case of commercial clients this may be through direct payment. In the case of open-source clients, the benefit may be a large user-community able to contribute to the development effort. In either case, users can be considered a common ‘resource’ which is shared between, and indeed competed for by, Gnutella developers. This competition is demonstrated, for example, by the large-scale advertising of commercial Gnutella clients [24] [25].

The dilemma for developers is whether to address the problem of freeriding by patching their clients and therefore reducing their user-base, or to maximise their user base by not implementing incentive measures. It is thus ‘rational’ for developers wishing to create successful clients *not* to implement such measures even though this degrades the performance of the network as a whole and, in the longer term would reduce the total number of Gnutella users who (would likely migrate to other peer-to-peer file sharing networks such as [4] or [5]).

It therefore appears that *two* ‘tragedies’ afflict Gnutella. The first is the ‘tragedy of the digital commons’ which leads rational users to free-ride in order to maximise their download efficiency, the second is a ‘meta tragedy’ that leads Gnutella developers to choose not to update their clients with incentive measures in order to maximise their user-base.

7. Conclusions

Gnutella remains unique among peer-to-peer file sharing systems, both in being completely open and in having a large, established and studied user base. Valuable lessons can be drawn from the falling level of participation observed since 2000 and from the lack of response to this problem from Gnutella host developers. One theoretical way to address the ‘meta’ tragedy of the commons would be to introduce a central body to enforce the implementation of necessary changes to the protocol; however, this seems fundamentally at odds with the peer-to-peer philosophy. Yet, finding an effective balance between maintaining the open nature of the protocol and effectively managing its evolution could prove critical to the long-term survival of the Gnutella protocol.

8. References

- [01] “*Napster*”, <http://www.napster.com>
- [02] “*Dependability Properties of P2P Architectures*”, Walkerdine, J., Melville, L., Sommerville, I. proceedings of the 2nd IEEE International Conference on Peer-to-Peer computing (P2P’02). Linkoping, Sweden, September, 2003.
- [03] “*The Gnutella Protocol Specification v0.4*”. http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf, 2000.
- [04] “*FastTrack*”. <http://www.fasttrack.nu/>.
- [05] “*Bittorrent*”. bitconjurer.org/BitTorrent/protocol.html.

- [06] “*The Tragedy of the Commons*”. Hardin, G., *Science* 162, 1243-1248, 231-266. 1968.
- [07] “*Free Riding on Gnutella*”. Adar, E., Huberman, B., *First Monday*, October 2000. <http://www.firstmonday.dk/issues/issue5.10>.
- [08] “*A distributed and measurement-based framework against freeriding in peer-to-peer networks*” Karakaya, M., Korpeoglu, I., Ulusoy, O., proceedings of the 4th IEEE International Conference on Peer-to-Peer computing (P2P’03). Zurich, Switzerland, September, 2003.
- [09] “*A Framework for Developing Reflective and Dynamic Peer-to-Peer Networks (RaDP2P)*” Hughes, D., Coulson, G., Warren, I., proceedings of the 4th IEEE International Conference on Peer-to-Peer computing (P2P’04). Zurich, Switzerland, August, 2004.
- [10] “*Why Gnutella Can’t Scale, No Really*” Ritter, J, <http://www.darkridge.com/~jpr5/doc/gnutella.html>.
- [11] “*The Gnutella Protocol Specification v0.6*”. <http://rfc-gnutella.sourceforge.net/>.
- [12] “*JTella*”. <http://jtella.sourceforge.net/>.
- [13] “*Lancaster’s Peer-to-Peer Website*” <http://polo.lancs.ac.uk/p2p>.
- [14] “*Comparing Corpora Using Frequency Polling*” Rayson, P., Garside, R., proceedings of the workshop on Comparing Corpora. October 2000, Hong Kong.
- [15] “*The Popularity of Gnutella Queries and its implications for scalability*” Sripanidkulchai, K., <http://www-2.cs.cmu.edu/~kunwadee/research/p2p/gnutella.html>.
- [16] “*Direct Connect*” <http://www.neo-modus.com>.
- [17] “*The MMAPS project*” <http://www.mmapps.org/>.
- [18] “*From Selfish Nodes to Cooperative Networks – Emergent Link-based incentives in Peer-to-Peer Networks*” Hales, D, proceedings of the 4th IEEE International Conference on Peer-to-Peer computing (P2P’04). Zurich, Switzerland, August, 2004.
- [19] “*AGnuS – The Altruistic Gnutella Server*”, Hughes, D., Warren, I., Coulson, G., proceedings of the 3rd IEEE International Conference on Peer-to-Peer computing (P2P’03). Linköping, Sweden, September, 2003.
- [20] “*Measuring and analyzing the characteristics of Napster and Gnutella hosts*”, Saroiu, S., Gummadi, P., and Gribble S., *Multimedia Systems*, volume 9, number 2, 170-184 (2003), Springer-Verlag.
- [21] “*High Availability, Scalable Storage, Dynamic Peer Networks: Pick Two*” Blake, C., Rodrigo, R., In the proceedings of the Ninth Workshop on Hot Topics in Operating Systems (HotOS-IX), Lihue, Hawaii, May 2003.
- [22] “*Measurement, Modelling and Analysis of a Peer-to-Peer File-Sharing Workload*”, by Gummadi, K., Dunn, R., Saroiu, S., Gribble, S., Levy, H., Zahorjan, J., Proceedings of the 19th ACM Symposium of Operating Systems Principles (SOSP), Bolton Landing, NY, October 2003.
- [23] “*Availability and Locality Measurements of Peer-to-Peer File Systems*”, Chu, J., Labonte, K., Levine, B., in the Proceedings of ITCOM: Scalability and Traffic Control in IP Networks II Conferences July 2002.
- [24] *Lime Wire* www.limewire.com
- [25] *Bear Share* www.bearshare.com