

# Methodology for Estimating Network Distances of Gnutella Neighbors

Vinay Aggarwal<sup>1</sup>, Stefan Bender<sup>2</sup>, Anja Feldmann<sup>1</sup>, Arne Wichmann<sup>1</sup>

<sup>1</sup>Technische Universität München, Germany {vinay, anja, aw}@net.in.tum.de

<sup>2</sup>MPI für Informatik, Saarbrücken, Germany {sbender}@mpi-sb.mpg.de

**Abstract:** In this paper we ask the question how much does the neighborhood selection process of a P2P protocol such as Gnutella respect the underlying Internet topology.

## 1 Introduction

Network applications, such as IRC, Mbone, Usenet, etc. route data at the application layer, thus creating overlay networks. A common aspect of these applications is that these overlay networks are controlled by system administrators, who ensure that neighbor choices respect resource limitations to some degree. One may expect that this biases the neighborhood choices to respect network proximity. This is in contrast to another class of overlay networks, the popular peer-to-peer file-sharing systems. Here system-specific metrics or arbitrary choices govern the neighborhood selection process.

Accordingly we, in this paper, ask the question how much does such a selection process respect the underlying network topology, or put differently, how close is a P2P topology to the Internet topology. The answer can help us estimate the (in-)efficiency of using overlays. We investigate this question using Gnutella as our example overlay network.

Gnutella [Cl01], one of the first decentralized overlay file sharing networks, is based on agents, called *servents*. According to its original design each servent maintains connectivity with a set of others by sending `Ping` messages, which are answered using `Pong` messages. Search queries are flooded within the Gnutella-network using `Query` messages and answered by `Query Hits`. To limit flooding Gnutella uses TTLs and message IDs. Each answer message (`Query Hit/Pong`) takes the reverse path of the corresponding triggering message. Due to scalability problems in the original design, later versions of the Gnutella protocol [Li] introduced a hierarchy which elevates some servents to ultrapeers, while others become leaf nodes. Each leaf node connects to a small number of ultrapeers while each ultrapeer maintains a large number of neighbors, both ultrapeers and leafs. To further improve performance and to discourage abuse, the `Ping/Pong` protocol underwent semantic changes. Answers to `Ping` messages may be cached and too frequent `Pings` or repeated `Queries` may cause termination of the connection or may be ignored by the receiver.

These changes have vastly improved the scalability of the Gnutella network [Li]. Yet at the same time they pose a huge impediment to investigating the structure of Gnutella, which is based on the original semantic of the Ping/Pong protocol. Accordingly in this paper we investigate how to overcome these limitations to find neighbors in the Gnutella network and then explore their network distance in comparison to random node distances.

Fully distributed P2P networks, such as Gnutella [CI01], attracted an enormous interest after Napster was sued by the music industry in 2001. Traditionally networks such as Gnutella are mapped by crawlers [SGG02, RFI02]. The main component of a crawler is a client which maintains a list of known Gnutella servents. It connects to each servent on this list and uses the Ping/Pong protocol with large TTLs to discover other Gnutella servents, which are added to the list, as well as edges in the Gnutella network. This results in a snapshot of the network. The crawls, typically lasting a few hours, discovered about 400,000 [RFI02], 120,000 [Li] and 1,239,487 [SGG02] servents respectively. Overall [RFI02] asserts that Gnutella's virtual network topology does not match the Internet topology well. [SGG02] found considerable heterogeneity in Gnutella, and presented evidence of distinct client- or server-like behavior in servents.

The remainder of the paper is organized as follows: Section 2 outlines our methodology for exploring the Gnutella network topology. Next Section 3 presents some preliminary results and finally Section 4 concludes with a short summary.

## 2 Methodology for identifying edges in a P2P network

In order to study how close the P2P topology is to the Internet topology we first need to identify a representative set of edges in the P2P network. Then we need to find a comparable set of edges in the Internet and a metric suitable for comparison.

The most obvious way of finding edges in a P2P network is to create some by participating. Yet these are not representative. They are highly biased by the location and the software of the participant. Rather we want to identify edges in the P2P network where neither of the two nodes is controlled by us. We refer to any two nodes connected by an edge as neighbor servents and those not involving a node controlled by us as remote neighbor servents.

Due to the changed semantics of the Ping/Pong protocol the simple crawling approach outlined in the last section is no longer sufficient. As Pongs are cached and due to the rapid fluctuations in Gnutella networks<sup>1</sup> one cannot assume that answers to Pings with TTL equal to two (so called crawler pings) contain still active servents. They should, however, have been remote neighbor servents at some point. Note that leaf nodes are no longer reported in Pongs.

To cope with these complications we deploy a combination of active and passive techniques to explore the Gnutella network. From the passive technique (an ultrapeer servent) we gain a list of active servents. Using Querys with TTL value of two allows us to get a set of remote neighbor servents. Using Pings with TTL value of two results in a set of

---

<sup>1</sup>In our experiments, see Section 3, the median connection duration is 0.74/0.98 seconds respectively.

candidate servents. These are then contacted actively to further advance the network exploration. This approach allows us to discover edges in the Gnutella network that existed at some point but it does not guarantee that they still exist. In the future we plan to enhance our strategy to ensure that we have discovered active remote neighbors by connecting to both servents at the same time and issuing a query with a TTL of three which can only be answered by the crawler servent. The problem with this approach is that connecting to two servents at the same time is problematic due to the restrictions on the neighborhood size of each servent.

Our active approach consists of multiple client servents and a manager. The manager controls the clients in that it supplies each client with a Gnutella servent address (IP address/port number combination) to connect to. Should a client not respond within a reasonable time frame it is restarted. Each client tries to connect to its assigned servent. Depending on success, connection refusal, connection timeout, or Gnutella error message the client reports a different result to the manager. Based on this the manager reschedules the servent for retry. If the connection is rejected with a Gnutella error code it is indicative of an active servent that most likely has no open connection slots currently available. If the connection times out, the servent is either inactive or behind a firewall. If the connection is refused, it is either inactive or highly overloaded with connection requests. Accordingly servents that rejected connections are retried faster than those that refused them or did not respond.

When interacting with other servents, the client is pretending to be a long-running ultrapeer with an acceptable querying scheme. It processes incoming messages and has a non-intrusive Ping/Pong behavior. For example the client issues query/crawler pings only to those peers that have already responded with a Pong, Pings are issued only to those peers that send one themselves, and at the same rate. This seems to avoid bans. The client uses Query messages with a compiled list of catchwords such as *mp3*, *avi*, *rar*. One can expect queries to yield only a subset of neighbors due to the presence of “free-riders” [AH00].

Early experiments showed that the behavior of a client can have significant impact on the connection success rate. This has led to several changes that make the client more attractive (e.g., large X-Live-Since times, ultrapeer handshaking) as well as less predictable (e.g., initializing the timers that issue the Query and Ping messages with random values within a certain range).

To better understand the limitations of our approach and the behavior of both client and ultrapeers, we experimented with the prevalent tools in a test bed. It consisted of a small Gnutella network with servents based on GTK-Gnutella, LimeWire, BearShare, and GnuClaus. Interestingly only GTK-Gnutella provides a configuration parameter to elevate it to an ultrapeer. We also observed several compatibility issues. For example while the LimeWire servent allows other servents to establish TCP connections to it, it then rejects the Gnutella handshake with an error message. BearShare also discourages other vendors’ servents from connecting to it. We conclude that non-compliance and compatibility issues impose limitations on the success rate of our techniques.

Our passive approach consists of an ultrapeer on the basis of GTK-Gnutella [GM<sup>+</sup>03].

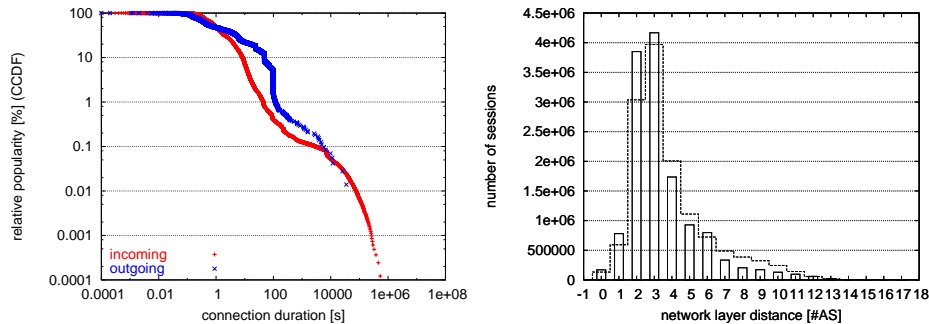


Figure 1: (a) CCDF of session duration distribution (b) Estimated number of autonomous systems between application layer distance one and two Gnutella servers (solid line) and random IP addresses (dashed line).

The goal is to have an ultrapeer that is just a normal node in the network, yet worthwhile to connect to. The ultrapeer shares a reasonable amount of data (100 randomly generated music files, totalling 300 MB in size), and maintains a maximum of 60 simultaneous connections to other servers. To derive various statistics the server is instrumented to log per-connection information which is augmented with a packet level trace.

Our combined active/passive approach integrates the crawler into the ultrapeer. Experiments with the unmodified and the modified ultrapeer confirmed that the changes did not alter the characteristics of incoming connections. Overall this allows us to reach a connection rate well above other known studies (e.g. [DZL02]) during the same time frame.

### 3 Results

Our initial characterization is based on a data set collected using the technique discussed in Section 2. The trace started on Oct 26, 2003 and lasted till Dec 3, 2003. During this time the ultrapeer logged 8,199,643 sessions of which 8,192,461 are incoming and 7,182 are outgoing. The dominance of the incoming connections indicates that the ultrapeer is quite popular, which is likely to reduce the bias in the sampled servers. The crawler discovered 14,101,399 remote neighbor servers.

Before exploring similarities of the P2P topology with the Internet topology we explore the variability of the Gnutella session durations. Figure 1(a) shows the complementary cumulative distribution function (CCDF) of the session duration of the above trace. It is apparent from the plot that most session durations are rather short. Indeed the median duration of incoming/outgoing sessions is 0.98/0.74 seconds. Only 5% of the incoming sessions lasted longer than 12.3 seconds. This implies that edges in the Gnutella network change rapidly. On the one hand this complicates any crawling attempts, on the other hand it affects the expected accuracy and value of any derived map.

Typical metrics for distances in the Internet are router hop counts and AS distances. Unfortunately, estimating the hop counts for any two random nodes is non-trivial [SMW02].

While difficult, estimating approximate AS distances is possible. We map IP addresses to AS numbers using BGP tables from Ripe [RI]. Using BGP tables and updates we derive an AS topology and the AS relationships [Ga00]. Based on this topology and the heuristic that a customer route is preferred to a peering route over an upstream, we estimate the AS distances. Figure 1(b) (solid line) shows a histogram of the estimated AS distances of the remote neighbor servers. The plot shows that the distances span a huge range with some clustering at distance 3 – 5. We note that the estimated AS distances for the direct neighbors have a significantly different distribution. The large values as well as the spread of AS values indicates that Gnutella does not bias its neighbor choices to correspond to network proximity. To further explore this the same Figure (dashed line) shows a histogram of the estimated AS distances of randomly chosen IP addresses. While the overall shape is quite similar there are some differences. This has to be expected since users need reasonable network connectivity to use the Gnutella network.

## 4 Summary

Exploring the Gnutella network topology is limited by the optimizations to the Gnutella protocol as well as the short session durations. Nevertheless we are able to identify a significant number of remote neighbor servers to approximate a representative set of edges in the P2P network. Our comparison to randomly selected pairs of IP addresses shows that neighbors in the Gnutella P2P network do not seem to significantly bias their neighbor choices towards network proximity.

## References

- [AH00] Adar, E. und Huberman, B.: Free riding on gnutella. Web Page. 2000. [http://www.firstmonday.dk/issues/issue5\\_10/adar/](http://www.firstmonday.dk/issues/issue5_10/adar/).
- [CI01] Clip2. The gnutella protocol specification v0.4. Web Page. 2001. [http://www9.limewire.com/developer/gnutella\\_protocol\\_0.4.pdf](http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf).
- [DZL02] Dhamdhere, A., Zegura, E., und Liston, R.: Determining characteristics of the gnutella network. In: *Technical Report, College of Computing, Georgia Tech*. 2002.
- [Ga00] Gao, L.: On inferring autonomous system relationships in the Internet. In: *Proc. IEEE Global Internet Symposium*. 2000.
- [GM<sup>+</sup>03] Grosse, Y., Manfredi, R., u. a. Gtk-gnutella - a clone of gnutella. Web Page. 2000-2003. <http://www.gtk-gnutella.sourceforge.net/>.
- [Li] Lime Wire. <http://www.limewire.com/>.
- [RFI02] Ripeanu, M., Foster, I., und Iamnitchi, A.: Mapping the gnutella network. In: *IEEE Internet Computing Journal*. 2002.
- [RI] RIPE's Routing Information Service raw data page. <http://data.ris.ripe.net/>.
- [SGG02] Saroiu, S., Gummadi, K., und Gribble, S.: A measurement study of peer-to-peer file sharing systems. In: *Multimedia Computing and Networking*. 2002.
- [SMW02] Spring, N., Mahajan, R., und Wetherall, D.: Measuring ISP topologies with Rocketfuel. In: *Proc. ACM SIGCOMM*. 2002.