

Characterization of Data on the Gnutella Peer-to-Peer Network

Jeffrey Miller

Department of Computer Science
University of Southern California
Jeffrey.Miller@usc.edu

Abstract - In this paper I present a characterization of the different types of files for which users search on the Gnutella network. I classify the searches into five different categories: audio files, video files, image files, documents and program files, and other files that cannot be classified as one of the other four types. I show that audio files are the most abundant type of file for which users search, followed closely by video files. Relatively few documents and program files are queried, and queries for image files are almost negligible. I also classify the percentage of queries that are x-rated and show that it constitutes a significant portion of the total queries. In addition to queries, I also quantify the distribution of distinct files shared by hosts and duplicate files that are shared by more than one host.

I. INTRODUCTION

Peer-to-peer programs have become a popular means of file exchange, which can partly be attributed to the ease of use of programs such as Morpheus [19], Kazaa [15], Bearshare [2], Limewire [16], and Freenet [9], among many others. A peer-to-peer network is defined by the protocol it uses, with Gnutella being one such protocol. The Gnutella protocol is an open protocol, which allows any programmer to create a server to communicate on the network. Three of the more popular commercial Gnutella applications are Morpheus, Bearshare, and Limewire, with Limewire reporting more than 100,000 hosts on the Gnutella network in April of 2003 [17].

The Gnutella peer-to-peer network has been analyzed with respect to many aspects of the network. The overall network has been observed with respect to the number of nodes, the number of different types of messages transmitted, and the number of links maintained by nodes based on how much data the node is sharing [23]. The bandwidth compared to the latency of nodes, in addition to determining a topology of the network, was studied in [25]. Ways to improve searching has been extensively studied using Search/Index Links [4], locally-organized lookup services [10], YAPPERS [11], routing indices [7], and iterative deepening, directed breadth-first search, and local indices [29]. Issues in scalability have been addressed [5,20,28], as well as the security implications associated with open networks [6,8,14,24].

Despite the vast amount of research being conducted in peer-to-peer systems, no one has yet performed a study of the specific content being searched for and shared in a peer-to-peer network. In this study, I have captured all of the queries that one Gnutella client received and categorized them based

on whether the query was searching for an audio file, a video file, an image file, a document or program file, or for data that was not able to be classified into one of the other four categories. In addition, I classified the queries into those that were in search of x-rated material and those that were not.

Realizing that the success of a peer-to-peer network does not predominantly rely on what data is being queried but on whether the data being queried is actually found in the network, I also created a program that attempts to capture all of the files that each host is sharing based on multiple queries sent by my application. Although other studies have shown that approximately 26% of the users on the Gnutella network are “free-riders” [1,25] (meaning that they do not share any files), of the remaining 74%, how many hosts are sharing files that are already shared by other users? The previous studies concluded that 74% of the nodes share 100% of the files, and 7% of the nodes together share more files than the other 93% of the nodes combined [25]. In my study, I have expanded on the above statistics to include how many of the files being shared are distinct and how many are duplicates. It can be reasonably assumed that there are a significant number of duplicate files on the Gnutella network, for peer-to-peer applications support simultaneous downloading of files from different nodes that share the same file [2,15,16,19]. This feature has no benefit if there is only one copy of a file to download. I conclude that approximately 58% of the files being shared are actually distinct.

II. BACKGROUND AND SETUP

I chose to perform the experiments on the Gnutella network for three main reasons. First, the Gnutella network provides an open test bed for my application in which no experimental setup is needed. Thus, the statistics are not based on a simulation of the network but rather on live snapshots. Second, the Gnutella protocol is an open protocol that enables a programmer to write his own server that will operate on the Gnutella network. In fact, there are also open implementations of the Gnutella protocol, notably a Java implementation called JTella [13] and a C implementation called GTK-Gnutella [12], among others. Based on my own experience and the portability of Java code, I used the JTella implementation as a basis, though I still had to write my own application on top of the protocol code. Third, since the Gnutella protocol floods the network with all queries, it is feasible to create a server to capture the queries for analysis.

In Gnutella, when a node wants to initiate a query, it sends the query to all of its neighbors, each of which forwards the query to all of their neighbors, and so on, until the maximum number of hops has been reached (or the time-to-live, or TTL, equals 0) [3,27]. The Gnutella protocol utilizes this controlled-flooding mechanism for queries, which allows me to capture all of the queries that are forwarded through a host on the network. However, in this paper I am attempting to characterize all of the queries that are transmitted on the Gnutella network, so the question then becomes whether the queries one node receives is an accurate representation of the queries transmitted on the entire network. Because the Gnutella network exhibits a power-law configuration scheme, it has been proven that 95% of any two nodes are no more than 7 hops away [21]. This means that each node will receive 95% of all of the queries on the network, which I believe is an accurate representation of all of the queries being transmitted on the network at a specific instant in time.

For this study, I created two different Gnutella host applications. The first application, called QueryCapture, was responsible for connecting to an initial set of Gnutella servents and capturing all of the queries received. The second application, called QueryHitCapture, was responsible for crawling the network, sending the necessary queries to servents to determine the files being shared at each host, and capturing the query hits when each host responded.

After I collected the data from the QueryCapture and QueryHitCapture applications I analyzed it using a statistical program for each host application. For QueryCapture, the statistical program, called QueryCaptureStats, was responsible for determining which queries were in search of audio files, video files, image files, documents and program files, and other files that could not be classified as one of the other four types. It was also responsible for determining which queries were searching for x-rated material and which queries were not. For more details on this application, refer to section III. For QueryHitCapture, the statistical program, called QueryHitCaptureStats, was responsible for determining which files were distinct and which files were shared by more than one host. Section IV contains more details on this application.

III. QUERIED FILE STATISTICS

The QueryCapture application connected with other servents on the Gnutella network and captured all of the queries for which it was asked to search. The application behaved exactly as a Gnutella servent behaves that shares no files. Every query it received was written out to a file, and if the TTL was not equal to 0, the query was forwarded to all of its outgoing links. After each execution of QueryCapture, QueryCaptureStats was executed to determine the percentage of the types of files for which users search, which is explained in section III.A, and the percentage of x-rated content for which users search, which is explained in section III.B.

QueryCapture was executed several times in the months of February, March, and April of 2003 for periods of time between 15 minutes and 48 hours. Table I provides a list of

TABLE I. DURATION AND TOTAL QUERIES CAPTURED ON DATES QueryCapture APPLICATION WAS EXECUTED

Exec #	Date	Duration (hh:mm:ss)	Total Queries
1	2-16-2003	23:06:23	2,179,182
2	2-19-2003	00:32:56	27,031
3	2-25-2003	00:15:48	8,205
4	3-10-2003	15:42:04	932,537
5	3-22-2003	00:29:13	17,740
6	4-2-2003	42:02:23	3,910,734
7	4-2-2003	00:40:12	50,920
8	4-15-2003	00:44:40	44,539
9	4-15-2003	00:18:32	13,170
10	4-16-2003	05:10:25	235,719
11	4-17-2003	00:32:00	47,741
12	4-17-2003	30:47:15	3,761,097
Total		120:21:51	11,228,615

all of the executions of QueryCapture with the duration and number of queries received on each execution.

The queries transmitted on the Gnutella network can be sent in one of four formats. First, the query could be transmitted as the exact string for which the user wanted to search. An example of this type of query is the following:

```
beatles - i feel fine.mp3
```

Second, the query could be transmitted as the exact string for which the user wanted to search, appended with the SHA1 encoding [26] of the query. An example of this type of query is the following:

```
beatles - i feel fine.mp3
urn:sha1:3KROPWG3MTVPNBB3HS4ATCAXVKMCDZCB
```

Third, the query could be transmitted with *only* the SHA1 encoding of the query. An example of this type of query is the following:

```
urn:sha1:3KROPWG3MTVPNBB3HS4ATCAXVKMCDZCB
```

Lastly, the query could be transmitted as an XML string. Morpheus is one such Gnutella application that transmits queries as formatted XML when the user is performing an advanced search for file metadata information, such as title, artist, or album of an audio file. An example of this type of query is the following:

```
beatles s:xml:<o name="AU">
<p><q>Title</q><r>i feel fine</r></p>
<p><q>Artist</q><r>beatles</r></p></o>
```

The use of the SHA1 hash algorithm [26] for encoding the queries before they are sent presented a small obstacle for the analysis. The queries encoded with the SHA1 hash algorithm are transmitted with the "urn:sha1:" prefix, followed by a string of 32 ASCII characters. A peer-to-peer application would match a SHA1 value by hashing all of the filenames the user is sharing and then comparing the hashed filename values with the SHA1 value sent in the query. Because hash algorithms inherently present a one-way mapping of keys to

TABLE II. QUERY STATISTICS CORRESPONDING TO EXECUTIONS PRESENTED IN TABLE I FOR AUDIO, VIDEO, IMAGE, DOCUMENTS AND PROGRAMS, AND UNCLASSIFIABLE QUERIES

Execution #	Audio	Video	Image	Docs/ Programs	Other	SHA-1	Total Queries
1	341,481	349,859	17,862	33,096	525,185	911,699	2,179,182
2	3,508	2,740	87	397	6,722	13,577	27,031
3	1,237	917	156	51	2,481	3,363	8,205
4	8,989	683	507	566	457,366	464,426	932,537
5	1,024	1,122	18	172	5,072	10,332	17,740
6	429,411	395,443	25,448	37,432	1,105,959	1,917,041	3,910,734
7	2,703	3,821	138	258	11,936	32,064	50,920
8	3,620	4,438	108	449	12,672	23,252	44,539
9	1,164	1,209	16	99	4,018	6,664	13,170
10	21,356	22,407	462	1,735	57,108	132,651	235,719
11	4,754	4,195	144	302	11,699	26,647	47,741
12	413,608	388,059	21,483	45,289	1,067,022	1,825,636	3,761,097
Total	1,232,855	1,174,893	66,429	119,846	3,267,240	5,367,352	11,228,615

TABLE III. MAPPING OF MEDIA TYPES TO WELL-KNOWN FILE EXTENSIONS

Media Type	Well-Known File Extensions
Audio	mp3, wma, ogg
Video	mpg, mpeg, avi, wmv, qt, rm, ram, asf, divx, mov
Image	jpg, gif, png, bmp
Document / Program	exe, vbs, bat, iso, pdf, zip, tar, gz, doc, xls, ppt, mdb, txt, csv

values, decoding a hashed value is computationally infeasible. However, I was able to decode some of the queries that came with only the SHA1 hash value *if* they happened to match one of the SHA1 values that were transmitted along with a user’s search query.

The percentage of total queries received by QueryCapture over all executions encoded with SHA1 was 50.35%, with a standard deviation of 8.88%. I was able to decode 3.38% of these queries, with a standard deviation of 2.32%. However, this still left 47.96% of the queries transmitted on the Gnutella network as those that I was not able to decode, and therefore my analysis is based on the 52.04% of the queries (5,861,263 queries) that I was able to determine the string for which the user was searching.

A. File Type Statistics

The QueryCaptureStats program performed the analysis of the queries received based on a string-matching algorithm. It can be assumed that a query containing the substring “mp3” is searching for an audio file. Keeping this assumption in mind, there are many well-known file types that can be associated with a specific form of media. Table III provides a mapping of file type to well-known file extensions that were used in my program to characterize the queries as audio files, video files, image files, documents and program files, and other files that did not fall into one of these four categories.

One of the features of searching in a peer-to-peer network is that a user can query based on a set of keywords, but file type is irrelevant. Because of this, there were a significant

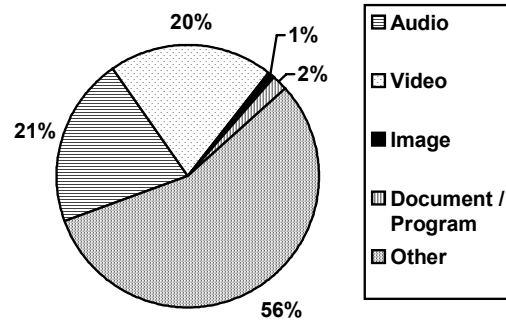


Figure I. Percentage of each type of file queried on the Gnutella network based on the 5,861,263 queries received by QueryCapture

TABLE IV. STANDARD DEVIATIONS BASED ON THE 12 EXECUTIONS PRESENTED IN TABLE II

File Type	Standard Deviation
Audio	6.93%
Video	6.46%
Image	0.84%
Document / Program	0.76%
Other	13.86%

number of files that could not be classified as any of the above four types. For example, if a user searches for “Beatles”, it is impossible for the application to know if the user is searching for an audio file, a video file, an image, or even a document or program about the Beatles. The user needed to provide more information than just “Beatles” for me to be able to classify the search as one looking for a specific type of file. Fig. I provides a diagram detailing the percentage of queries out of the 5,861,263 queries captured by QueryCapture in search of the different file types. Table II shows the actual number of queries that were sent in search of the different file types for each execution of QueryCapture corresponding to the executions in Table I.

As can be seen, there are a significant percentage of queries that cannot be classified as searching for audio files, video files, image files, or documents and program files. However, it is encouraging that I was able to classify

TABLE V. QUERY STATISTICS CORRESPONDING TO EXECUTIONS PRESENTED IN TABLE I FOR X-RATED AND NON-X-RATED QUERIES

Execution #	X-Rated	Non-X-Rated	SHA-1	Total Queries
1	258,548	1,008,935	911,699	2,179,182
2	2,036	11,418	13,577	27,031
3	722	4,120	3,363	8,205
4	41,379	426,732	464,426	932,537
5	791	6,617	10,332	17,740
6	305,035	1,688,658	1,917,041	3,910,734
7	3,188	15,668	32,064	50,920
8	3,090	18,197	23,252	44,539
9	996	5,510	6,664	13,170
10	12,745	90,323	132,651	235,719
11	3,275	17,819	26,647	47,741
12	257,416	1,678,045	1,825,636	3,761,097
Total	889,221	4,972,042	5,367,352	11,228,615

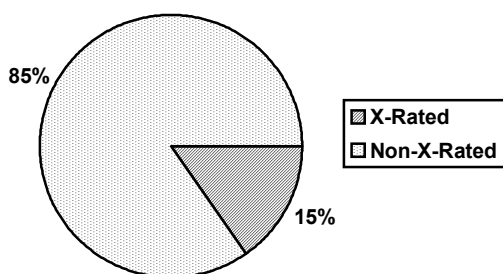


Figure II. Percentage of x-rated queries based on the 5,861,263 received by QueryCapture

approximately 50% of the queries into one of the four categories, since the information used to classify the files is not necessary for a user to provide.

As was expected, audio queries constitute the majority of queries on the Gnutella network, followed closely by video queries. Image files, documents, and program files are not queried significantly, though they should not be overlooked. In a 24-hour period, I captured almost 2.2 million queries – 2% of which is 44,000 queries for documents and programs.

The standard deviations for the percentages in Fig. I are given in Table IV. Interestingly enough, the standard deviation for audio files is slightly larger than that of video files. In some of the executions, there were even more queries for video files than audio files. This is due to the fact that the executions that were run for extended periods of time (i.e. more than 12 hours) tended to exhibit statistics more consistent with those depicted in Fig. I. However, the executions that were run for shorter amounts of time (i.e. less than 12 hours) had results that varied and did not necessarily remain consistent with the averages of the results.

Considering only the shorter queries, I can conclude that a snapshot of the network does not provide an accurate representation of what types of files are being queried over time. Just as a peer-to-peer network will eventually converge in its topology based on the dynamic nature of nodes joining and leaving the network [22], the percentages of types of files

being queried will also converge, which will ultimately reduce the standard deviations of the queries for each file type.

B. X-Rated Statistics

QueryCaptureStats was also responsible for determining which queries were searching for x-rated material, which was based on a string-matching algorithm. The application searched for 44 different words that can be deemed as x-rated, and the results were as expected. There were a significant number of x-rated queries, though the queries that were not x-rated far outweighed them. This result should follow from section III.A, which showed that the majority of files for which users search in the Gnutella network are audio files. Furthermore, the great majority of audio files are music files, which are not generally x-rated. Table V shows the actual number of queries that were deemed x-rated for each of the 12 executions listed in Table I. Fig. II shows the percentages of queries that were x-rated, which represents the average of all of the 12 executions listed in Table V.

The standard deviations on both the x-rated and the non-x-rated queries were 2.59%. This is relatively low, though it is still important to note that a snapshot of the network does not provide an accurate characterization of the queries that are actually transmitted over time.

Although I attempted to group the x-rated queries based on different times of the day, this was not feasible because of the different time zones in the worldwide Gnutella network. I would like to hypothesize that more x-rated queries are generated in the evening and at night since users probably do not search for x-rated material during the day while at work, but this was not possible for me to verify on the Gnutella network since I do not know from which time zone the query originated.

IV. SHARED FILE STATISTICS

The QueryHitsCapture application connected with other hosts on the Gnutella network and sent queries to all of its neighbors in search of each of the file extensions provided in Table III (i.e. *.mp3). All of the responses were captured and saved into a file that was analyzed by the QueryHitsCaptureStats program. It is important to note that

TABLE VI. SHARED FILE STATISTICS CORRESPONDING TO EXECUTIONS OF THE QueryHitsCapture APPLICATION

Exec #	Date	Distinct	Duplicate	Total
1	2-16-2003	4,874	3,247	8,121
2	2-19-2003	3,176	2,983	6,159
3	2-25-2003	3,922	2,502	6,424
4	3-22-2003	3,565	4,826	8,391
5	3-22-2003	4,113	3,117	7,230
6	4-2-2003	7,958	5,402	13,360
7	4-15-2003	6,416	4,057	10,473
8	4-15-2003	8,843	5,388	14,231
9	4-15-2003	6,884	3,558	10,442
Total		49,751	35,080	84,831

the number of responses returned was on the order of hundreds to thousands per minute, as was also observed in [18]. Any file name returned in the message corresponded to a file that was being shared by that host on the network.

QueryHitsCapture was run numerous times at different times of the day, with the number of files returned on each execution shown in Table VI. This was intended to provide a more accurate view of the network over an extended period of time, though after approximately 3 minutes, the query returned an insignificant number of results (if any), and the executions were usually halted after 5 minutes.

The QueryHitsCaptureStats program was responsible for determining what percentage of the files being shared were duplicates of a file being shared by another host and which files were distinct. Fig. III graphically displays the average of the results of all of the executions of QueryHitsCapture, corresponding to the data in Table VI.

As can be seen, 58% of the files being shared are distinct, leaving only 42% as files that are shared by more than one host. The standard deviation for both of these percentages was 7.01%, which is slightly high though justifiable. In the dynamic nature of the Gnutella network, hosts join and leave the network more frequently than a crawler of the network can portray. Different files are shared at different times, depending on the hosts that are on the network at that specific moment in time. Since my host application was run at many different times of the day, the specific files being shared varied. However, all of the executions provided statistics that converge upon 58% of the files being distinct, and I did not gather any statistics that significantly disagreed with this percentage.

V. CONCLUSION

In this paper, I provided a characterization of the different types of files for which users search on the Gnutella network, as captured during the months of February, March, and April of 2003. Approximately 48% of the queries are encoded using the SHA1 hash algorithm in such a manner that I cannot know the actual query string. Of the remaining 52%, with a very low standard deviation, I determined that 21% of all of the queries captured over all executions of the QueryCapture application were searching for audio files, 20% for video files, 2% for programs or document files, 1% for image files, and

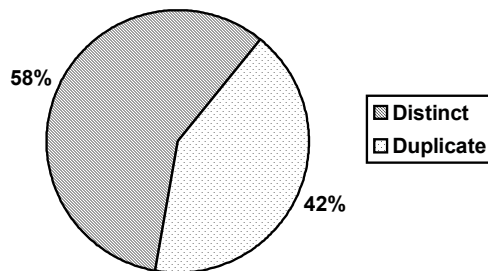


Figure III. Percentage of distinct and duplicate files shared on the Gnutella network based on the 84,831 files captured by the QueryHitsCapture

56% did not contain any keywords that enabled me to determine the file type the user desired.

In addition to classification of the queries on file type, I also performed an analysis on the percentage of queries in search of x-rated material, which was 15% of the total queries captured, and queries in search of non-x-rated material, which was 85%. Although the non-x-rated material definitely dominates the queries, the x-rated queries do account for a significant percentage of the queries on the network.

Instead of only focusing on the queries being transmitted, I also analyzed the percentage of files that are shared by more than one host at the same time. Although I expected to have a lower percentage of distinct files than I measured, the statistics gathered were from a set of snapshots of the network. The values I present in this paper show that the percentage of distinct files shared by hosts converge upon 58%, even though the percentage may be higher or lower given a specific snapshot of the network.

Though peer-to-peer systems are often defended on the basis that they represent a new mode of computing that may have great potential benefits, my study supports the conclusion that the primary benefit to date is the trading of audio and video files, which probably consist of copyrighted material. Though this may not come as a surprise, it may weaken the stridency of those computer researchers who defend these systems while ignoring their use.

REFERENCES

- [1] Eytan Adar, Bernardo A. Huberman. "Free Riding on Gnutella," *First Monday*, Volume 5, Number 10, October 2000.
- [2] Bearshare website. <http://www.bearshare.com>.
- [3] Clip2. "Gnutella Protocol Specification v0.4," http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf.
- [4] Brian F. Cooper, Hector Garcia-Molina. "Studying Search Networks with SIL," *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems*, February 2003.
- [5] Brian F. Cooper, Hector Garcia-Molina. "Modeling and Measuring Scalable Peer-to-Peer Search Networks," *Proceedings of ACM SIGCOMM*, 2001.
- [6] Brian F. Cooper, Mayank Bawa, Neil Daswani, Hector Garcia-Molina. "Protecting the PIPE from Malicious Peers," Technical Report, Stanford University, 2001.
- [7] Arturo Crespo, Hector Garcia-Molina. "Routing Indices for Peer-to-Peer Systems," *Proceedings of the 22nd International Conference on Distributed Computing Systems*, July 2002.

- [8] Neil Daswani, Hector Garcia-Molina. "Query-Flood Attacks in Gnutella," *Proceedings of the 9th ACM Conference on Computer and Communications Security*, 2002.
- [9] Freenet website. <http://www.freenetproject.org>.
- [10] Prasanna Ganesan, Qixiang Sun, Hector Garcia-Molina. "A Case for Locally-Organized Peer-to-Peer Lookup Services," *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems*, February 2003.
- [11] Prasanna Ganesan, Qixiang Sun, Hector Garcia-Molina. "YAPPERS: A Peer-to-Peer Lookup Service over Arbitrary Topology," *Proceedings of INFOCOM 2002*, July 2002.
- [12] GTK-Gnutella website. <http://gtk-gnutella.sourceforge.net>.
- [13] JTella website. <http://jtella.sourceforge.net>.
- [14] Sepandar D. Kamvar, Mario T. Schlosser, Hector Garcia-Molina. "The EigenTrust Algorithm for Reputation Management in P2P Networks," *Proceedings of the 12th International World Wide Web Conference*, May 2003.
- [15] Kazaa website. <http://www.kazaa.com>.
- [16] Limewire website. <http://www.limewire.com>.
- [17] Limewire website. http://www.limewire.com/historical_size.html.
- [18] Evangelos P. Markatos. "Tracing a large-scale Peer to Peer System: an hour in the life of Gnutella," *Proceedings of the 2nd IEEE/ACM International Symposium on Cluster Computing and the Grid*, 2002.
- [19] Morpheus website. <http://www.morpheus.com>.
- [20] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, Scott Shenker. "A Scalable Content-Addressable Network,"
- [21] Matei Ripeanu, Ian Foster. "Mapping the Gnutella Network: Macroscopic Properties of Large-Scale Peer-to-Peer Systems," *Proceedings of the 1st International Workshop on Peer-to-Peer Systems*, March 2002.
- [22] Matei Ripeanu, Ian Foster, Adriana Iamnitchi. "Mapping the Gnutella Network: Properties of Large-Scale Peer-to-Peer Systems and Implications for System Design," *IEEE Internet Computing Journal*, Volume 6(1), 2002.
- [23] Matei Ripeanu. "Peer-to-Peer Architecture Case Study: Gnutella Network," *Proceedings of the 1st International Conference on Peer-to-Peer Computing*, August 2001.
- [24] Jared Saia, Amos Fiat, Steve Gribble, Anna R. Karlin, Stefan Saroiu. "Dynamically Fault-Tolerant Content Addressable Networks," *Proceedings of the 1st International Conference on Peer-to-Peer Computing*, August 2001.
- [25] Stefan Saroiu, P. Krishna Gummadi, Steven D. Gribble. "A Measurement Study of Peer-to-Peer File Sharing Systems," *Proceedings of Multimedia Computing and Networking 2002*, January 2002.
- [26] SHA1 Hash Algorithm Specification. http://www.w3.org/TR/1998/REC-DSig-label/SHA1-1_0.
- [27] Anurag Singla, Christopher Rohrs. "Ultrapereers: Another Step Towards Gnutella Scalability," <http://rfc-gnutella.sourceforge.net/Proposals/Ultrapeer/Ultrapeers.htm>, 18 December 2001.
- [28] Kunwadee Sripanidkulchai. "The Popularity of Gnutella Queries and its Implications on Scalability," *Proceedings of O'Reilly's Peer-to-Peer and Web Services Conference*, 2001.
- [29] Beverly Yang, Hector Garcia-Molina. "Improving Search in Peer-to-Peer Networks," *Proceedings of the 22nd International Conference on Distributed Computing Systems*, July 2002.