An Enhanced Gnutella P2P Protocol: A Search Perspective

S. H. Kwok and K. Y. Chan

*Department of Information and Systems Management
Hong Kong University of Science and Technology
Clear Water Bay, Hong Kong
{jkwok, frankcky}@ust.hk*

Abstract

Although Gnutella is the most popular Peer-to-Peer protocol for file sharing, drawbacks are often found in searching and locating files. They include that (1) a limited number of hosts being searched leads to a limited number of hits; (2) the search result is not accurate due to the search by filename; and (3) an inefficient use of computer resource – spending too much resource on handling other peers' queries. In order to rectify these drawbacks, this paper proposes N-phase Search and Passive Search mechanisms. Compared with the traditional search mechanism, N-phase Search increases the number of received query hits by 55.41% and increases the host coverage by 41.23%. For the Passive Search, the average search time is reduced by 15.33%.

1. Introduction

Peer-to-peer (P2P) networking is undoubtedly one of the most touted topics in the Internet. The popularity achieved by applications like Napster and Gnutella provides testimony to the potentials of P2P application. Currently, P2P applications are mainly used for music file sharing. However, Gnutella's uses and potential could go beyond file sharing, for example knowledge reuse management [4]. Gnutella has the potential and the capability to completely revolutionize the current way we interact with the Internet contents and applications.

Gnutella-based P2P systems possess three distinct features: decentralization, anonymity, and autonomy. The decentralization implies a networking architecture without a centralized server control. The absence of the central control overcomes a single or multiple points of failure of P2P program. The decentralization also induces the dynamic re-configurability characteristic of network, in which an "arbitrary" network is always formed and the network keeps changing all the time. This characteristic is due to the preferences of participating P2P programs that can select preferable peers for network connection and refuse connection requests from un-trusted peers. This introduces the property of autonomy that applies peer's preference in network forming and searching. The property of anonymity is an important property to use

P2P applications; no one knows the identity of a P2P user in sharing, searching and downloading files.

Directions for improvement of searching in the Gnutella network would be to enhance the query flooding mechanism and to employ the proxy cache mechanisms used on the Web [5]. In the latest Gnutella specification version 1.1 [12], we identify several potential improvements on the Gnutella protocol.

2. Gnutella Protocol

The Gnutella protocol [1] is an open, decentralized group membership and search protocol, mainly used for file sharing. The term Gnutella also designates the virtual network of Internet accessible hosts running Gnutella-speaking applications and a number of smaller, and often private, disconnected networks.

Gnutella nodes, called servents by developers, perform tasks normally associated with both servers and clients. They provide client-side interfaces through which users can issue queries and view search results, accept queries from other servents, check for matches against their local data set, and respond with corresponding results. These nodes are also responsible for managing the background traffic that spreads the information used to maintain network integrity.

In order to join the system a new node/servent initially connects to one of several known host catchers that contain a list of Gnutella hosts (e.g. public.bearshare.net). Once attaching to the network (e.g., having one or more open connections with nodes already in the network), nodes exchange messages and interact with each other. Messages can be broadcasted (i.e., sent to all nodes with which the sender has open TCP connections) or simply back-propagated (i.e., sent on a specific connection on the reverse of the path taken by an initial, broadcasted, message). Several features of the protocol facilitate this broadcast/back-propagation mechanism. First, each message has a randomly generated identifier. Second, each node keeps a short memory of the recently routed messages, used to prevent re-broadcasting and to implement back-propagation. Third, messages are flagged with time-to-live (TTL) and "hops passed" fields.

To become a member of the network, a servent has to open one or many connections with nodes that are already in the network. In the dynamic environment where Gnutella operates, nodes often join and leave and network connections are unreliable. To cope with this environment, after joining the network, a node periodically checks its neighbors to discover other participating nodes. Using this information, a disconnected node can always reconnect to the network. Nodes decide where to connect in the network based only on local information; thus, the entire network forms a dynamic, self-organizing network of independent entities. This virtual, application-level network has Gnutella servents at its nodes and open TCP connections as its links.

The Gnutella protocol defines five different types of packets, namely QUERY, HIT, PING, PONG, and PUSH.

1. QUERY – requests for sharable files matching the given criteria.
2. HIT- responds to a query by returning a list of sharable files matching the given criteria and the IP addresses of their providers. There may be multiple responders to a query due to the nature of distributed network.
3. PING - requests the transitive closure of connected nodes to identify themselves.
4. PONG - responds by a node upon receiving a PING, the responding node provides its IP address and the number of sharable files it contains.
5. PUSH - requests a file provider to contact the requester. This provides a simple mechanism to attempt to get through the firewalls.

To search for files in the Gnutella network, the Gnutella servent sends a QUERY packet that contains the query string. The neighboring nodes will receive this QUERY packet and they will take the query string as search criteria to check their sharable local files that match the criteria. If the P2P servent finds relevant files in its local resource, it will generate one or more HIT packets containing information about the relevant sharable files. In the current Gnutella protocol, the servents only check the filenames of the sharable files.

Figure 1 illustrates the query-response operation. Assume Servent A requests for files with the keyword “mp3”. The query string containing the keyword will be sent to all of its neighbors, say one of them is Servent B. On receiving the request from Servent A, Servent B will automatically forward the query string to its neighbors too if the time-to-live (TTL) counter is still non-zero after the decrement by one. The request initiated by Servent A is recorded to avoid any duplicate requests from the same Servent because it is possible that the same QUERY string will return to Servent B for files matching “MP3”. After passing back anything that might be of interest to

Servent A, Servent B passes the request on to its peer, Servent C, and Servent B keeps a record that Servent A has made the request. If Servent C has files matching the request, it passes the information to Servent B. Servent B checks its records of request to see which request the HIT matches. If the returning information matches with the QUERY from Servent A, Servent B passes the information to Servent A.

Figure 1. The request-response operation in Gnutella protocol

3. Existing Search problems and solutions

In the Gnutella network, a node will forward all incoming “Ping” and “Query” to all of its directly connected nodes, except the one that delivered the incoming “Ping” and “Query”. Also the node will decrement a descriptor header’s TTL field, and increment its Hops field, before it forwards the descriptor to any directly connected nodes. If, after decrementing the header’s TTL field, the TTL field is found to be zero, the descriptor is not forwarded along any connection. This searching mechanism limits the “radius” traveled by the messages by TTL. For file that is widely available in the network, a small TTL is sufficient. For file that is rare in the Gnutella network, it is hard to find the desired files without distributing the queries widely. Such a searching mechanism limits the scalability of the Gnutella network. As a result, there will be no “Query Hit” response, even the requested file is present in the network. Increasing the value of TTL in the “Query” message can be one of the methods to increase the horizon that can be reached by the message; however, this approach will increase the amount of network traffics in an exponential order. For example, a node is connecting to 4 nodes, and each of these nodes is connecting to another 4 nodes, so for $TTL=2$, the initial node is able to reach $4+4*4 = 20$ nodes. If the TTL is increased to 3, the initial node will be able to reach $4+4*4*4 = 68$ nodes.

Several solutions to this problem were proposed by different researchers [7, 8]. Those solutions are based on a highly structured network design in which the network can provide exact location of a specific file. However, in the Gnutella network, the nodes turnover rate is very high, it is estimated that half of the nodes participating in

received in a 10-minute period and the average number of distinct hosts from which “Query Hit” messages returned.

Table 1a. Experimental results of N-Phase Search (Number of Query Hits)

N-phase Search (16 Dec 2003)			
Result set	Outermost nodes	Traditional	N-Phase
1	25	2264	3816
2	23	2551	3227
3	34	1015	2139
4	21	1253	3577
5	22	5690	5815
6	30	3677	3805
7	21	1911	5924
8	28	3793	5753
9	23	3777	6242
10	25	1862	6237
11	25	2225	2576
12	23	3139	5528
13	27	1604	2946
14	26	3018	5290
15	18	2380	3846
16	15	1198	1301
17	16	2275	2698
18	13	4035	4924
19	14	1905	2722
20	28	3814	4600
Average number of query hits		2669.3	4148.3

Table 1b. Experimental results of N-Phase Search (Host Coverage)

N-phase Search (16 Dec 2003)			
Result set	Outermost nodes	Traditional	N-Phase
1	25	210	239
2	23	197	195
3	34	131	204
4	21	150	224
5	22	181	231
6	30	165	220
7	21	139	249
8	28	114	268
9	23	186	303
10	25	163	326

11	25	218	222
12	23	181	247
13	27	19	166
14	26	194	257
15	18	163	262
16	15	146	167
17	16	219	236
18	13	156	216
19	14	142	182
20	28	227	248
Average number of hosts		165.05	233.1

We completed 20 sets of traditional and N-phase file searching. For each sample set, the search session lasted for 10 minutes. The content of “Query Hit” messages and the total number of “Query Hit” messages were recorded. Then the number of distinct hosts from which the “Query Hit” messages returned was measured by extracting the location information, IP address, from the “Query Hit” messages.

The average number of “Query Hit” messages received for traditional search was 2669.3 while the average number of “Query Hit” messages for N-phase Search was 4148.3. The average number of distinct hosts covered for traditional search was 165.05, while that for N-phase Search was 233.1. The average number of “Query Hit” messages received in a 10-minute period by using N-phase Search increased by 55.41%, while the host coverage increased by 41.23%. The results are presented in Table 1a and Table 1b. For reference, the number of outmost hosts involved in N-phase Search was also shown in the tables.

5.2. Passive search

In order to evaluate the effectiveness of the passive searching mechanism, we conducted another set of experiments with the procedures similar to the previous ones. The searching effectiveness was reflected by the total time used to return 1000 “Query Hit” messages with a certain searching criterion using both the traditional search and the Passive search representatively. The average time to get 1000 “Query Hit” messages for traditional search was 46.00 seconds while that for Passive Search was 38.95 seconds. The average time saved in getting 1000 “Query Hit” messages by using Passive Search was 15.33%. The results are shown in Table 2.

Table 2. Experimental results of Passive Search

Passive Search (28 Sep 2003)		
Result set	Traditional	Passive
1	43	23
2	32	23
3	34	39
4	19	53
5	48	24
6	32	26
7	41	30
8	58	26
9	30	24
10	30	21
11	31	32
12	33	20
13	61	14
14	36	23
15	33	27
16	62	84
17	41	95
18	112	24
19	44	28
20	100	143
Average time - 1000 query hits (seconds)	46	38.95

6. Future Development

This section discusses further improvements on Gnutella protocol.

6.1. MP3 Meta-data Search Mechanism

Searching for MP3 is the most popular query in P2P network. The current query matching mechanism in the Gnutella network is based on the search keywords input by the users and the file names of the files shared by the nodes. If the filename of a file matches the search keyword, the application will forward a "Query Hit" message to announce the availability of the file.

Metadata are data about other data and objects. Metadata are used to describe digitized and non-digitized resources located in a distributed system in a networked environment [9, 10]. In a MP3 file, metadata are saved in ID3v1 or ID3v2 format [2]. The metadata is a fix-sized 128-byte tag that would reside at the end of the MP3 file. It includes fields of title, artist, album, year, genre and

comment. Since the metadata record the specificity of the song, these fields could be treated as valuable criteria for searching. Performing MP3 metadata searching can increase the accuracy of finding a particular song.

For the "Query" matching mechanism of the Gnutella protocol, since the matching is only based on the filename, it is possible that files actually contain the same data but with different files names. For example, a user can save or rename any file according to the user's own convention and preference. The use of MP3 metadata search can improve this scenario, because modifying the metadata may not be as easy as modifying the filename of a MP3 file. Moreover, the accuracy of results can also be increased by providing more criteria for searching. Some users would like to search MP3 songs of a particular album, or songs of a particular artist. They cannot get relevant results if they apply the traditional search mechanism because there are so many songs that are alike in the Gnutella network. However, as most of the MP3 songs record common fields like artist, album, year, genre, etc, they can again be treated as a valuable source for searching. Searching by more than one criterion in metadata can therefore increase the accuracy of the results.

We have prototyped the metadata search with XML messages. Search criteria are formatted in XML documents and forwarded to other nodes in the network. These XML feature can be further modified to include more enhanced messages, for example, some application monitoring instructions or network management commands. This will be the one of the future directions of this research project.

6.2. Partial Downloading Mechanism

Slow file download speed and disconnection of the node in the middle of the download process is always a concern in the Gnutella network. Like normal HTTP download, the Gnutella node will resume the previous incomplete downloads by sending the "resuming message". However, this resuming is only possible when the "disconnected" node connects to network again. In order to speed up the download speed and minimize the chance for incomplete download, we propose a partial downloading mechanism that allows the peer to download a single file from different hosts simultaneously by segmenting the file into different parts. The node will download file parts from different hosts and combine the parts into a single file when all the partial downloads are completed. Therefore, even one of the uploading nodes disconnects or the transmission rate of one of the uploading node is very slow, the downloading node can also download from other hosts simultaneously.

To evaluate this feature, we compared the effectiveness of normal downloading and partial

downloading process. The file downloading effectiveness depends on the bandwidth for file transfer. The effectiveness can be justified by measuring the total time used to download one file with the same name and size, using normal and partial downloading at the same time. We provide three uploading hosts, which stored the desired file. We control the upload hosts by randomly disconnecting the uploading hosts from the network. Moreover, the uploading speed of the uploading hosts is limited to 6000 bytes per second. The difference between average time to get one file (the size of the file used in the simulation is 4M, which is similar to the size of a MP3 file) downloaded between normal download and partial download is 800 seconds. In most of the simulation samples, the time to download by partial downloading is still at least one third shorter than by non-partial downloading.

7. Conclusions

Several drawbacks of Gnutella protocol have been identified. We proposed mechanisms to enhance the search feature of Gnutella protocol. The proposed mechanisms include N-phase and Passive search mechanisms. These mechanisms were derived from the current Gnutella protocol and therefore they can adapt to existing Gnutella protocol without causing problems. We conducted experiments to evaluate the performance of the proposed mechanisms, compared with the traditional search mechanism. The experimental results show that while using N-phase Search, the number of query hits received increases by 55.41% and the host coverage increases by 41.23%, and while using Passive Search, the average time saved in getting 1000 query hits is 15.33%.

8. Acknowledgments

The work described in this paper was partially supported by grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. HKUST6256/03E), the Teaching Development Grant (TDG) (Ref. HKUST-1-F), and the Sino Software Research Institute (SSRI) (Ref. SSRI01/02.BM01). We would like to thank Jack Wong, Elsa Wong, Derek Hau, Alice Chiang, Henry Lin, Ivan Ng, and Jennifer Chan for the implementation of the P2P prototype.

9. References

[1] Anonymous, The Gnutella Protocol Specification v0.4, <http://www.securitytechnet.com/resource/hot-topic/p2p/GnutellaProtocol04.pdf>, accessed on 5 April 2003.

[2] Anonymous, ID3v2, <http://www.id3.org/>, accessed on 5 April 2003.

[3] A. Fiat and J. Saia, "Censorship Resistant Peer-to-Peer Content Addressable Networks," presented at 13th ACM-SIAM Symposium on Discrete Algorithms, 2002.

[4] S. H. Kwok, "Decentralized Knowledge Reuse With Peer-to-Peer Technology," presented at Proceedings of the First Workshop on e-Business (Web2002), 2002.

[5] R. Matei, A. Iamnitchi, and P. Foster, "Mapping the Gnutella network," *IEEE Internet Computing*, vol. 6, pp. 50-7, 2002.

[6] K. McCrary and B. Waters, JTella, <http://jtella.sourceforge.net/>, accessed on 5 April 2003.

[7] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content-addressable network," *ACM Computer Communication Review*, vol. 31, pp. 161-172, 2001.

[8] A. Rowstron and P. Druschel, "Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility," *ACM Operating Systems Review*, vol. 35, pp. 188-201, 2001.

[9] D. Rusch-Feja, Metadata: Standards for Retrieving WWW Documents (and Other Digitized and Non-Digitized Resources), <http://www.eso.org/gen-fac/libraries/lisa3/ruschfejad.html>, accessed on 5 April 2003.

[10] J. Saia, A. Fiat, S. Gribble, A. Karlin, and S. Saroiu, Dynamically Fault-Tolerant Content Addressable Networks, <http://www.cs.rice.edu/Conferences/IPTPS02/180.pdf>, accessed on 5 March 2003.

[11] S. Saroiu, P. K. Gummadi, and S. D. Gribble, "Measurement study of peer-to-peer file sharing systems," presented at Spie - the International Society for Optical Engineering, 2001.

[12] M. Starmer, Peer-to-Peer File Sharing Protocols (version 1.1), http://final-score.wiserhosting.com/demont/CSCI3002/PeerToPeerFileSharing_1.1.htm#_Toc25384292, accessed on 5 April 2003.