A Study of Scalable Search Algorithm on Unstructured P2P System
(Work-in-Progress)

Fuyong Yuan
College of Information Scienceand Engineering, Yanshan University
Qinhuangdao, China
fyyuan@ysu.edu.cn

Jian Liu
College of Information Scienceand Engineering, Yanshan University
Qinhuangdao, China
chinafather@163.com

Chunxia Yin
College of Information Scienceand Engineering, Yanshan University
Qinhuangdao, China
lovelyfatbear@163.com

ABSTRACT
We proposed a search algorithm to unstructured P2P network, which consists of ranked neighbor caching, queryhit caching, and file replication to free riders. And the simulation results show that the algorithm can extend the search region but reduce the search traffic, and also balance the network load, so that acquires the whole network scalable.

Categories and Subject Descriptors
H.3.3 [**Information Search and Retrieval**]: Information Search and Retrieval –*Search process, Selection process*

General Terms
Algorithms, Experimentation

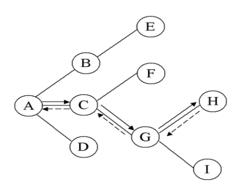
Keywords
flooding-based, Gnutella, search, unstructured P2P networks

1. INTRODUCTION
While the Gnutella model has managed to succeed thus far, in theory its scalability isn't well. The number of queries and the number of potential responses increases exponentially with each hop. And the Gnutella protocol itself does not provide a fault tolerance mechanism. The hope is that enough nodes will be connected to the network at a given time such that a query will propagate far enough to find a result. However, the distributed nature of the protocol does not guarantee this behavior. In fact, some studies have shown that only a small fraction of Gnutella users actually remain online long enough to respond to queries from other users. Additionally, many of the peers in Gnutella are free riders, which only waste traffic without contribution. To improve search efficiency and reduce unnecessary traffic in Gnutella, we propose an algorithm based on [1] and [2]. It can extend the search region but reduce the search traffic, and also balance the network load, so that acquire the Gnutella networks scalable. Ranked neighbor caching, queryhit caching, and file replication to free riders are three parts of our proposed algorithm.

INFOSCALE 2007, June 6-8, Suzhou, China

DOI 10.4108/infoscale.2007.197

2. PROPOSED ALGORITHM
2.1 Ranked Neighbor Caching
In ranked neighbor cache scheme, neighboring peers assign each other trust ranks. And the higher the rank for a peer *B* at its neighbor peer *A*, the more likely *A* would forward the query message to *B*. When *B* replies *A* with a valid queryhit message, *A* should add one to peer *B*'s rank value which must be initialized to zero and updated based on one-step feedback mechanism. Suppose *A* has *s* neighbors $N_1, N_2, \ldots, N_s$ ($s > 0$), and r_i indicates the rank peer *A* assigns to its neighbor N_i. Then, peer *A* will choose neighbor N_i as its query "receiver" with the forwarding probability: $p_i = (r_i + r)/\sum_{n=1}^{s}(r_n + r)$. The risk factor *r* must satisfy the constraint $r >- r_i$ (i = 1, … s) to ensure that the forwarding probabilities are positive for all neighbors [1]. In the scheme, every peer needs to cache not only its neighbors' ranks but also the probabilities of them. Figure 1 shows an example that the peer *A* searched a file in the network and found it at *H*.



Peer	Neighbor	Rank	Possibility
A	*B*	0→0	1/3→1/4
A	*C*	0→1	1/3→1/2
A	*D*	0→0	1/3→1/4
C	*F*	0→0	1/2→1/3
C	*G*	0→1	1/2→2/3
G	*H*	0→1	1/2→2/3
G	*I*	0→0	1/2→1/3

Figure 1. Search procedure of the scheme.

The scheme increases the scalability of the network by extending the search region and reducing the traffic. However, the traffic is tend to centralized to a few links which are connected to the high ranked peers. The reason is that the high ranked peers have more chance to send queryhit message, and then their rank turn to higher and higher. So we proposed following two schemes.

2.2 Queryhit Caching
In Gnutella, the queryhit message includes the information about the file, but not the address of the requester. Thus every peer

needs to cache not only queryhit message but also the direction of the queryhit message sent. Since queryhit caching scheme leads the query to the peer, which downloads the file from the high ranked peer, not to the high ranked peer, this scheme can reduce the load of the high ranked peer.

2.3 File Replication to Free Riders

When a file is transferred, if there are some free riders on the way to destination peer, the file is replicated to the free riders. After that, if these free riders receive request for that replicated file from other peers, they can reply and transfer the file directly. This scheme changes the free riders to file suppliers to reduce unnecessary traffic and also realize load balancing of the whole networks.

2.4 Process of the Whole Algorithm

Firstly, when a peer searches a file, the peer checks its queryhit cache. If the file information is cached in the queryhit cache, then the peer propagates the query along the connection where the peer owns the file. Secondly, the peer does not broadcast query to all of the neighbors, instead the peer only sends to one according to the ranked neighbor cache scheme. If the neighbor cached the file in file cache space, the peer downloads the file from that peer directly and quits the propagation. Otherwise the neighbor checks its queryhit cache. If the file information is cached in the queryhit cache, then the neighbor peer propagates the query along the connection where the peer owns the file. If there is no file information in queryhit cache, then the neighbor sends query to next neighbor according to the ranked neighbor cache scheme. At last, the peer downloads the file after finding it. Thirdly, queryhit was copied to peers in the search path according to the queryhit cache scheme. And file was replicated to free riders according to file replication to free riders scheme.

3. SIMULATION RESULTS

In this section, we design a Gnutella simulator employed a power-law like distribution topology with one thousand peers. Each peer generates 100 queries and issues one query per time slot, and peers may be normal peers or free riders in the network [2]. And, we suppose every query message passes *n* peers every time.

3.1 Traffic of the Network

We test the traffic of the simulator with the proposed algorithm, and compared with the classic flooding-based search. Figure 2 shows the throughput of the whole network per minute. The throughput of flooding-based search is 84.6 MB/min, ranked neighbor caching is only 12.6 MB/min. The ranked neighbor caching scheme generates much lower traffic than flooding-based search. And we also compared the value of *n* when the number of connections to each peer is 4 to 7 and TTL value is 7. With ranked neighbor caching scheme, query passes extremely fewer peers than flooding-based search. Therefore, the ranked neighbor caching scheme, presents good performance in traffic.

3.2 Load Balance

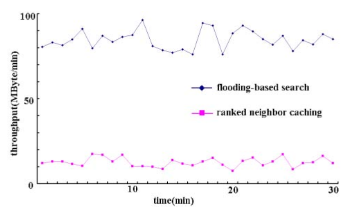
As shown in Figure 3(a), the query response rate of 1% high ranked peers is measured. With no cache, that is flooding based search, the average query response rate of is 90.2%, and with ranked neighbor caching scheme, the query response rate is 94.1%. However, with queryhit caching and file replication schemes, these peers only manage 64.2% of the query messages. Obviously, these two schemes are efficient to balance the load of network.

3.3 Success Rate

We simulate other two related algorithms for comparison, and name our proposed algorithm as P-search. One is random walk search, named as R-search, which means a peer forwards queries to a randomly chosen neighbor at each hop in the network [3]. Another is Max-Degree-biased search, named as M-search, which means a peer forwards queries to the highest-degree neighbor at each hop in the network [4]. The comparison for three algorithms' success rates as TTL varies is shown in Figure 3(b). P-search achieves a very high query success rate and a fast response time. Peers using P-search to forward queries are most likely to find the desired resources.

4. CONCLUSION

We proposed a search algorithm to unstructured P2P network, which include ranked neighbor caching, queryhit caching and file replication to free riders. Ranked neighbor caching is efficient to extend the search region and reduce the traffic effectively. And through other two schemes, the traffic cannot centralized in a few links, the free riders become useful peers, and they also balance the network load. Then, comparisons show that the algorithm achieves a high query success rate while greatly reduces traffic volumes. So, the network can be more scalable.



Connections	flooding-based(*n*)	ranked neighbor caching(*n*)
4	4298	17
5	27012	69
6	35896	276
7	51878	1032

Figure 2. Traffic of the network.

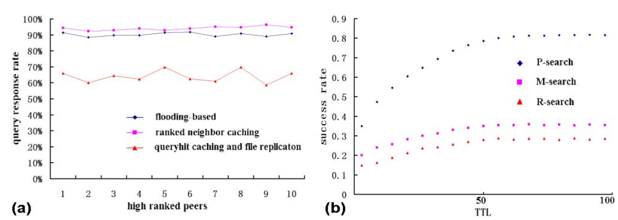


Figure 3. Comparisons of the performance.

5. REFERENCES

[1] H.ai Zhuge, X. Chen and X. Sun, Preferential Walk: towards Efficient and Scalable Search in Unstructured Peer-to-Peer Networks, *International World Wide Web Conference*, 2005.

[2] L. Quan, T. R. Eom, K. G. Lee, J. Jang and S.Y. Lee, Retrieval schemes for scalable unstructured P2P system, *ISICT*, 2004.

[3] C.Gkantsidis, M.Mihail and A.Saberi, Random walks in Peer-to-Peer networks, *INFOCOM*, HongKong, China, 2004.

[4] M.Ripeanu and I.Foster, Mapping the Gnutella network: macroscopic properties of large-scale Peer-to-Peer systems, *in IPTPS*, Cambridge, MA, 2002.