Two New Ways of Source Node Lookup in DHT Network

Xiangzhan Yu and Xingqi Shang

Harbin Institute of Technology, Harbin City, China
yxz@hit.edu.cn,shangxingqi@pact518.hit.edu.cn

Abstract

In this paper, through analyzing the process of publishing resource files in DHT network and from the viewpoint of making full use of the information stored in values nodes, we propose two methods to improve the lookup efficiency of source nodes: values-node-based lookup algorithm and source nodes exchange algorithm. The former firstly sends find_node message to those found values nodes, then sends get_peers message to nodes in the returned nodes-list so that increase the number of source nodes in fewer iterations. The latter organizes source nodes into a connected graph taking advantage of the information stored in values nodes, and based on this, with additional interactive messages we can improve the performance of looking up source nodes.

Keywords: *Peer-to-Peer; DHT network; source nodes lookup; values nodes*

1. Introduction

DHT [1] (Distributed Hash Table), a typical decentralized structured P2P networks, is highly tolerated to the failure of single point, extensible and without center. Therefore, it has been widely used in file sharing, streaming media services, distributed computing, distributed data storage and many other fields. There are a variety of DHT technologies at current, such as CAN [2] (Content Addressable Network), Chord [3], Pastry DHT [4], the P-Grid [5], Tapestry DHT [6] and KAD [7] (Kademlia), *etc*. KAD is widely used in BitTorrent (BT) and eMule, the most popular P2P file sharing systems. It effectively reduces the payload of the central index server (such as BT Tracker), increases the efficiency of the node query and improves the performance of the system. Usually in the P2P file-sharing system, a resource file is divided into a number of pieces. Each node can download different pieces in parallel from multiple source nodes, what is with file pieces, to increase speeds. After measurement, it is found that the more source nodes, the greater the likelihood of obtaining complete resources, the higher degree of download parallelism, and the faster the speed of download. Therefore, improving the performance of lookup source nodes and the comprehensiveness of the query results in DHT network is significance to improve the performance of P2P file-sharing system.

Researchers have been done a lot of study in improving query efficiency and the performance of DHT networks. In KAD [7], querying node use parallel search method to avoid interruptions of single path lookup caused by some offline nodes. The Bamboo [8] protocol proposed by SeanRhea and others can detect the failed routing nodes timely and accurately, and enhance the tolerance to the invalid routing table entries. Besides, it also introduces a network congestion mechanism avoiding that some nodes bear too much routing burden. Daniel Stutzbach [9] and others give proper degree of lookup parallelism based on actual testing of KAD network used by eMule. Raul Jimenez, *et al.*, [10], argue node query can be completed within one second by modifying the updating strategy of routing table in Mainline DHT network. Philipp Rösch, *et al.*, [11], described some necessary operations during query process to response to the dynamics of large-scale P2P network, and also discussed assessment strategies of query. These achievement reduce the side effect of the dynamics of node participation, or churn, on the search performance, speed up the query and to some extent, compensate for the defect of lack of node offline reporting mechanism in DHT network. However, due to the heterogeneous of DHT network and dynamic, the lookup algorithm used by existing BT clients is not efficient or can't find enough source nodes, which reduce the users' download speed for resources. For example, KAD protocol in BT network (called Mainline DHT) only uses values-list returned from values nodes (values nodes refer to the nodes which receive get_peers message and return values-list) to locate a source node, abandoning other information stored in values nodes. It results to waste time and valuable network bandwidth in query, and even cannot provide enough source nodes to download the whole file. Therefore, in this paper, taking Mainline DHT as an example, after analyze DHT protocol, from the viewpoint of making full use of the information stored in values nodes, we present two methods to improve the search efficiency for source nodes: values-node-based lookup algorithm and source nodes exchange algorithm. As far as we know, it is the first time to propose methods to search source nodes from this aspect. We also proved the effectiveness of these two methods in experiment, and they are applicable in other similar DHT network as well.

The paper is organized as follows: We introduce Mainline DHT protocol in Chapter 2. In Chapter 3 and Chapter 4, we propose respectively values-node-based lookup algorithm and source nodes exchange algorithm, and demonstrate their effectiveness in a simulated environment. We conclude our work in Chapter 5.

2. Introduction of Mainline DHT Protocol

In Mainline DHT network, each resource file is stored as a <key, value> pair. key is a unique hash value of the file and value is the location's information of file. All <key, value> pairs constitute resource storage table of the network. Each node is responsible for a local routing and stores the information of adjacent nodes. Nodes request resources to other nodes according to the key and provide others with the resources possessed of. In this paper, we use nodes, peers and users to refer the participants in DHT network alternatively.

2.1. Interactive Messages in Mainline DHT

In BT system, DHT nodes complete the interaction through the four types of messages: ping, find_node, get_peers and announce_peer. Each message carries the sender's ID at least and each ID is a 20-byte string [12]. Now we introduce these four types of messages.

ping: This message is used to detect whether a node is online. The query parameter is the ID of the requested node. The response message of ping includes the ID of the answering node.

find_node: This message is used to lookup a node P. There are two parameters: id (ID of the sender) and target (ID of P). The answering message of find_node has two parameters: id (ID of answering node) and nodes-list (information of K nodes closest to P in routing table of answering node, including ID, IP and listening port).

get_peers: If the ID of nodes returned from find_node is very close to the target source ID, Peers will send get_peers messages to these nodes query the source node of a certain resource. There are two parameters: id (ID of sending node) and info_hash (a 20-byte string, the info_hash of a certain resource). The answering message of get_peers has three parameters: a) id (a 20-byte string, ID of answering node); b) token(a string type, used to send next announce_peer message) ; c) if the queried node contains the downloaders' information of this resource, it is values-list(a list of BT nodes with the required resource); or it is nodes-list(a list of K nodes closest to the hash value in the routing table of queried node).

announce_peer: A peer uses announce_peer message to announce others with the resources information it has. There are four parameters: id (ID of sending node); info_hash (hash value of released resource Infohash); port (listening port of BT client); token(the returned value of last get_peers query). The answering message has only one parameter, ID of the answering node.

2.2. Resources Publishing Process in Mainline DHT

In BT system, when a node finishes downloading a file piece from other nodes, it will become resource provider and announce this resource to other nodes. In this way, peers share the files and cooperate in download files. In Mainline DHT network, a node publishes the resources it has in the following steps:

(1) Getting the info-hash of the resource. At first, the file owner (a source node called seeder) calculate a unique 20-byte resource ID (namely info-hash) of the resource to be publised with a certain hash method (SHA-1 usually), then publish this info-hash and the basic information (such as description, file size, *etc.*) of the resource on a website or other DHT nodes. Later on, other nodes can get its info-hash from the other source nodes or web sites.

(2) Finding nodes whose ID is close to resource ID. The seeder (or source node) looks up a set of nodes closest to the resource ID in its routing table and then sends find_node messages (target is the resource ID) to these nodes. Then it send find_node messages iteratively to returned results of find_node until getting the nodes who are close enough to the resource and send get_peers message to them.

(3) Send ping message to nodes returned by get_peers message to check if it is online.

(4) Send announce_peer message to node online in step (3). Nodes receiving the messages store info_hash and resource information and then become values nodes.

Because of different locations of nodes in Mainline DHT network, the values nodes we got in the release of resources are not identical. The more values nodes, the greater the likelihood to find more of the source nodes. From above process we can draw:

(1) The values nodes of a resource are close to its info_hash;

(2) The values nodes of the same resource are closer. Routing table of any values node may have multi values nodes' information. It is easy to find more other values nodes to iteratively query values nodes.

(3) The values nodes of a resource may store many source nodes' information of this resource.

2.3. Current Method to Find Source Nodes in Mainline DHT

In current Mainline DHT network, the way of searching for source nodes is as follows:

(1) Calculating/obtaining the info-hash of the target resource;

(2) Sending get_peers messages to K nodes closest to the info-hash selected from local routing table;

(3) Sending get_peers message iteratively to nodes in the returned nodes-list extracted from get_peers messages until closest enough;

(4) Get IP and port of source nodes from returned values- list extracted from get_peers messages;

Mainline DHT network is dynamic and lack of node-offline-reporting mechanisms, resulting in lower accuracy of routing and network connectivity, which seriously affects the efficiency of search algorithm. By analyzing the publishing process in 2.2, we can conclude that a values node may store information about many sources nodes and other values nodes. If we can search for more values nodes from found values nodes by sending iteratively get_peers message, or organize source nodes into one (or many) connected graph with the source nodes' information in values nodes, more source nodes for some μTorrents will be found in smaller searching time and bandwidth. In this paper, we present two approaches of improving lookup efficiency for popular resources and unpopular resources. Detailed description is respectively in Chapter 3 and Chapter 4.

3. Values-Node-Based Lookup Algorithm (VNLA)

According to the protocol specification of Mainline DHT, during the process of finding source nodes, nodes receiving get_peers messages will selectively reply to the requester with a nodes-list or values-list. Current BT clients only connect to those nodes returning values-list to exchange piece. From the announce process of resource in Mainline DHT network, we know that the values nodes of the same resource are close, and routing table of each values node may be likely to store information about other values nodes or source nodes. Theoretically speaking, if we send find_node or get_peers message to found values nodes, we could increase source nodes in less iterations.

3.1. Algorithm description

The values-node-based fast search algorithm is shown in Algorithm 1. And it works as follows: the request peer do the searching algorithm like current method described in 2.3 at begin. Once a values node is found, the request peer sends find_node message to it and sends iteratively get_peers messages to the nodes included in returned nodes-list, wishing that peers can find more values nodes in less iterations and thus find more source nodes.

Algorithm 1: values-node-based lookup algorithm
Input: info-hash of resource F
Output: list of source nodes sourcelist
Begin:

```
search for nodes close to info-hash of F in routing table of peer P, and put these nodes' information into peerlist
for each node Q in peerlist
do
    send get_peers message to Q
    if (return nodes-list) then
        update peerlist: add some nodes in returned nodes-list to peerlist.
    else
        update sourcelist: add source nodes in values-list to sourcelist.
        send find_node message to node Q
    update peerlist with returned nodes-list.
    end if
end for
```

End

The following example (as shown in Figure 1) describes the process of the algorithm to find the source nodes. Suppose in a Mainline DHT network, node P sends search requests, and S1, S2, S3 are three source nodes sharing resource F. P1, P2, P3 are values nodes. S1 sends a announce_peer message to P1, S2 sends a announce_peer message to P2, and S3 sends a announce_peer message to P3. It will search for resource F as following steps:

(1) P searches for nodes close to the info-hash of resource F in its local routing table and sends get_peers message to them;

(2) After several iterations, P sends get_peers messages to P2. P2 returns value- list to P;

(3) From values-list, P can find a source node S2, establish TCP connection with it and start downloading and uploading with S2;

(4) P sends find_node messages to values node P2. The target parameter is info_hash of F;

(5) P iteratively sends get_peers message to nodes in nodes-list returned by P2. Since P1, P2 and P3 are close in IDs, it will take less iteration to find P1 and P3 from P2;

(6) When receives information about S1, S3 from values- list returned from P1, P3, P establishes TCP connection with them and starts downloading.

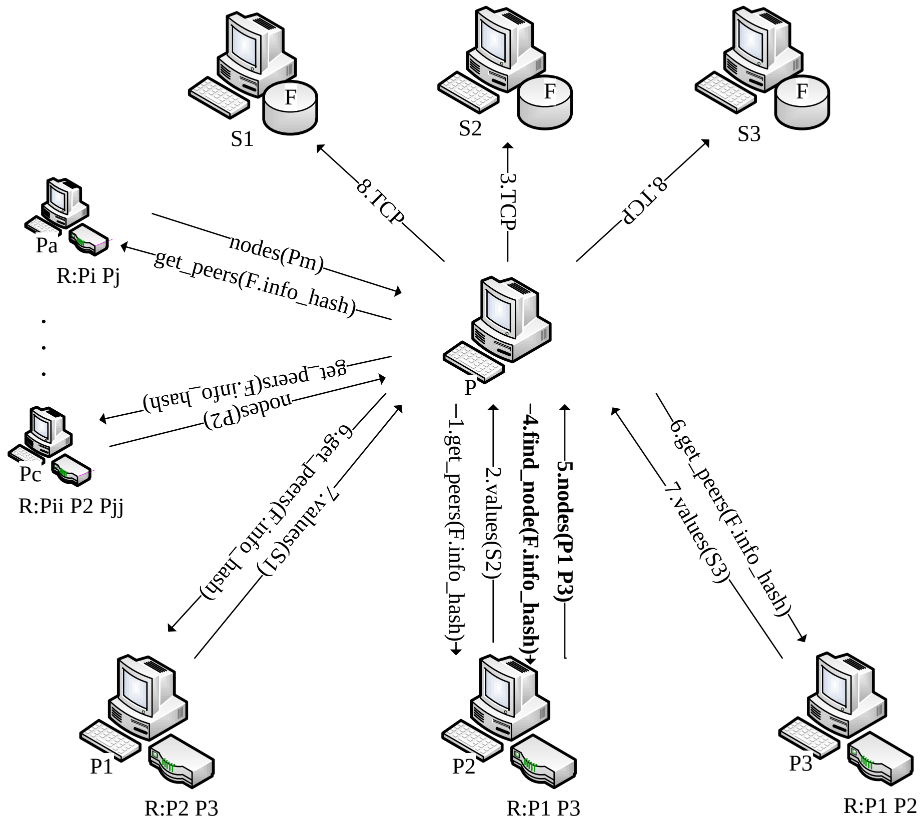

Figure 1. Process of Values-Node-Based Fast Search Algorithm

This algorithm is based on the characteristic of Mainline DHT protocol that the values nodes of the same resource are close in ID distance. The step (4) is added to existing method to find more source nodes. This method greatly reduces the iterations without additional message types, improves efficiency of lookup source nodes and save bandwidth.

3.2. Experimental verification

To verify the effectiveness of the improved algorithm, we perform experiments with popular μTorrents (number of nodes is greater than 100) and unpopular μTorrents (Number of nodes is less than 100) respectively. We run the BT client software (μTorrent) and our test client in actual Mainline DHT network. Both the two clients find source nodes of a same resource in the same number of iterations. The obtained experimental data are shown in the Table 1 and Table 2.

Table 1. Comparison of the Number of Source got from Traditional Method and Improved Method for Popular Seed

Seed	Iteration number N	Traditional source number	Improved source number	Increased percentage
1	1	23	42	82%
1	2	43	53	23%
1	3	274	288	5%
2	1	504	1287	155%
2	2	873	1463	67%
2	3	1600	2119	32%
3	1	148	236	37%
3	2	362	464	28%
3	3	831	1005	20%
4	1	157	238	52%
4	2	163	212	30%
4	3	398	569	43%

As shown in Table 1, for popular μTorrents, our improved algorithm can find more source nodes than BT client software after the same iteration number, especially in the first iteration 30%~155% source nodes can be increased. With the increasing of iteration times, we find that the number of source nodes is increasing but the increased proportion is lower. This shows that: for popular resource, this improved algorithm can find the same number of source nodes with current BT client by sending fewer get_peers messages. Hence it can save query time and bandwidth effectively.

Table 2 shows that for unpopular resource, this method cannot improve search efficiency markedly. With the increasing of iteration times, the number of source nodes does not change significantly. Thus, this algorithm can increase the efficiency of lookup source nodes for the popular resource significantly without modifying the existing protocols.

Table 2. Comparison of the Number of Source got from Traditional Method and Improved Method for Unpopular Seed

Seed	Iteration number N	Traditional source number	Improved source number	Increased percentage
1	1	1	1	0%
1	2	77	96	25%
1	3	23	23	0%
2	1	7	7	0%
2	2	8	8	0%
2	3	80	80	0%
3	1	1	1	0%
3	2	1	1	0%
3	3	19	19	0%
4	1	3	3	0%
4	2	3	3	0%
4	3	31	32	3%

4. Source Nodes Exchange Algorithm(SNEA)

For popular μTorrents, maybe due to a) source nodes are too small or part of them are offline, b) values nodes are relatively distant, values-node-based lookup algorithm can't markedly improve the efficiency of looking up source nodes. To solve this problem, we propose source nodes exchange algorithm.

4.1. Algorithm description

In Mainline DHT network, when a node have pieces of a resource file, it will send announce_peer message to the nodes whose ID are enough close to the file's info-hash. Thus, requesting node may get information about this source node through sending get_peers and download this file through establishing BT session with it.

To get more source nodes every time sending get_peers, we put forward source nodes exchange algorithm. The main idea is: when a resource file is published in DHT network, part of nodes around the info-hash will receive announce_peer of multiple source nodes. These nodes inform the source nodes of other source nodes' information they know, and these source nodes can form a connected graph by these information, called source set. When requesting node searches source nodes, if only it find one source node in the set, it can get a source node set. Thus, getting more source nodes with less iteration and improving the search efficiency.

To implement this modified algorithm, the additional storage space in the source nodes to store other source nodes' information and extra interactive messages are needed. The message fields are in Bencode encoded.

(1) announce_source: values nodes send it to source nodes to tell it other source nodes' information. It's information format is:

Fileds	Length(Byte)	Meaning
y:q	1	Represent sent packet
q:announce_source	15	Protocol type
id:abcdefghij0123456789	20	Its own ID
info_hash	20	Resource hash
sources	6n	Source node list

For example:

d1:ad2:id20:abcdefghij01234567899:info_hash20:mnopqrstuvwxyz1234567:sourcesl123456ee1:q15:announce_source1:t2:aa1:y1:qe.

(2) announce_source_reply: source nodes return this message when receive announce_source message. It's information format is:

Fileds	Length(Byte)	Meaning
y:r	1	Represent replied packet
id:mnopqrstuvwxyz123456	20	Its own ID

For example:

d1:rd2:id20:mnopqrstuvwxyz123456e1:t2:aa1:y1:re.

(3)get_source: requesting node send this message to source nodes to request information of more source nodes. It’s information format is:

Fileds	Length(Byte)	Meaning
y:q	1	Represent sent packet
q:get_source	10	Protocol type
id:abcdefghij0123456789	20	Its own ID
info_hash	20	Seed’s hash

For example：

d1:ad2:id20:abcdefghij01234567899:info_hash20:mnopqrstuvwxyz123456e1:q10:get_source1:t2:aa1:y1:qe.

(4)get_source_reply: source nodes replies get_source_reply message to requesting node with all source nodes information it knows. It’s information format is:

Fileds	Length(Byte)	Meaning
y:r	1	Represent replied packet
id:abcdefghij0123456789	20	Its own ID
info_hash	20	Seed’s hash
sources	6n	Source node list

For example：

d1:ad2:id20:abcdefghij01234567899:info_hash20:mnopqrstuvwxyz1234567:sourcesl123456ee1:q15:announce_source1:t2:aa1:y1:re.

```
Algorithm2: source nodes exchange algorithm
Input: info-hash of file F
Output: source node list sourcelist
Begin:
    search for node close to info-hash of F in routing table of P and put this information into peerlist.
    for each node in peerlist Q
    do
        send get_peers message to Q
        if (return nodes-list) then
            update peerlist: add some nodes in node-list into peerlist
        else
            tmp_sourcelist = values-list
            for each source node S in tmp_sourcelist
            do
                send get_source message to S
                update tmp_sourcelist with returned source list
            end for;
            update sourcelist with tmp_sourcelist
        end if
    end for
End
```

The following example (as shown in Figure 2) describes this algorithm: node P proposes search request, S1, S2 and S3 are three source nodes of resource F, P1 and P2 are values nodes.

The process of publishing resource:

(1) Assume that S1 sends announce_peer message to P1.

(2) S3 sends announce_peer messages to P1 and P2 respectively. Now P1 has known that S1 and S3 are source nodes and sends announce_source messages to them respectively.

(3) S1 and S3 receive messages and store information of each other, then reply announce_source_reply message to P1;

(4) Similarly, after P2 receives announce_peer message from S2, it sends announce_source to S2 and S3 respectively.

(5) After S2 and S3 receive messages, store each other information and reply announce_source_reply message to P2. Now S1,S2 and S3 form a connected graph. It has prepared for future source node search.

The process of looking up source node:

(6) Assume that after several iterations, P sends get_peers message to P1. Then P1 returns values-list containing information of S1 and S3;

(7) P sends get_source messages to S1 and S3, then S1 and S3 replied P with other source node information that they know.

(8) P receives the reply and knows S2 is a new source node, then sends a get_source message to S2.

(9) S2 replies get_source_reply message.

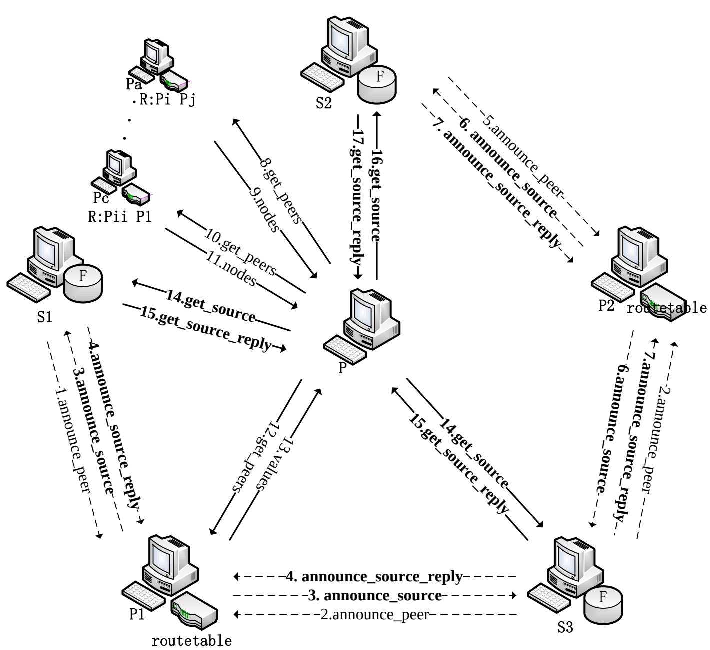

Figure 2. Process of Releasing Resource and Searching of Source Nodes Exchange Algorithm

4.2. Experimental Verification

The improved algorithm adds new message types and source nodes behavior has changed in release and searching process. Thus, experiment can't be carried out in the current Mainline DHT network. In order to test this improved algorithm, we imitate a simple Mainline DHT network in several servers, and then in this mimetic network search for source nodes of the same resource using current and improved algorithm separately. The steps are as follows.

Current algorithm simulation:

⑴ Program to simulate Mainline DHT network of N nodes (N takes 50000, 200000 and 500000). These nodes simulate behavior of normal nodes such as normal offline (offline rate reaches 80% in an hour).

⑵ Simulate 1500 source nodes of a popular μTorrents and 300 source nodes of an unpopular μTorrents;

⑶ Source nodes publish resource information to in mimetic network with unimproved algorithm.

⑷ Simulate a normal requesting node. Search for source nodes of popular μTorrents and unpopular μTorrents in analog network according to the current algorithm.

Improved algorithm simulation:

⑸ simulate DHT network of N nodes and source nodes of popular μTorrents as well as unpopular μTorrents like above.

⑹ According to the improved algorithm, simulate the release of a resource: when a node receives announce_peer messages from multiple source nodes and sends announce_source messages to them. The source node receiving announce_source message stores the information about other source nodes in local relevant structure and replies announce_source_reply.

⑺ Simulate a normal Mainline DHT node. Search for source nodes using improved algorithm, sending get_source message to source nodes that has been found.

⑻ Source nodes receiving get_source looks up relevant structure and returns stored information of other source nodes. If there is no other source node information, sources list is empty.

We did a number of experiments in accordance with the process above and the results are shown in Table 3 and Table 4.

Table 3. Comparison of the Number of Source got from Traditional Method and Improved Method for Popular Seed

Seed	Iteration number	Unimproved source number	Improved source number	Increased percentage
1	50000	278	557	100%
1	200000	105	160	60%
1	500000	54	117	61%
2	50000	243	510	60%
2	200000	152	256	66%
2	500000	46	78	60%
3	30000	59	101	90%
3	200000	43	95	110%
3	500000	18	45	120%

Table 4. Comparison of the number of source got from traditional method and improved method for unpopular seed

Seed	Iteration number	Unimproved source number	Improved source number	Increased percentage
1	50000	20	20	0%
1	200000	7	31	342%
1	500000	3	25	733%
2	50000	18	65	261%
2	200000	5	19	280%
2	500000	2	13	550%
3	50000	17	32	88%
3	200000	8	30	275%
3	500000	3	25	733%

Experimental results show that for popular μTorrents, improved algorithm can improve efficiency of searching for source node by 60%~120%. But values-node-based lookup algorithm has a very good effect without change of Mainline DHT. Therefore, this improved algorithm is not the best for popular resource. For unpopular μTorrents, the number of source nodes is pretty small when using current lookup method, while it increases significantly to three times to seven times when using improved method.

5. Conclusion

In this paper, by analyzing the resource announce and search process in current Mainline DHT network, we find some shortcomings of the existing methods in searching for the source nodes. To improve the searching results, we propose "Fast search algorithm based on values nodes" and "Improved algorithm based on the source nodes exchange agreement". We prove the effectiveness of the two algorithms with experiments in real-world environment and simulation environment. Fast search algorithm based on values nodes does little change to Mainline DHT network and interaction between nodes, reduces bandwidth required to search for source node.

Results show it is suitable for popular torrents. We add four interactive messages to existing protocol in improved algorithm based on the source nodes exchange agreement. It can greatly improve the search efficiency of the source node of unpopular torrents compared with existing methods

Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grant No. 61173144; the National Science and Technology Support program under Grant No. 2012BAH37B01.

References

[1] Distributed Hash table, http://zh.wikipedia.org/wiki/.
[2] S. Ratnasamy, P. Francis, M. Handley, R. Karp and S. Shenker, "A Scalable Content Addressable Network", Proceedings of ACM SIGCOMM 2001, **(2001)**, pp. 161-172.
[3] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek and H. Bal-akrishnan, "Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications", Proceedings of the ACM SIG-COMM 2001 Conference, **(2001)**.
[4] A. Rowstron and P. Druschel, "Pastry: Scalable, Decentral-ized Object Location, and Routing for Large-Scale Peer-to-Peer Systems", IFIP/ACM International Conference on Distributed Systems Platforms, **(2001)**.
[5] K. Aberer, "P-Grid: A Self-organizing Access Structure for P2P Information Systems", Proceeding of CoopIS 2001, Trento,Italy , **(2001)**, pp. 179-194.
[6] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph and J. D. Kubiatowicz, "Tapestry: A Resilient Global-scale Overlay for Service Deployment", IEEE Journal on selected areas in communications, vol. 22, no. 1, **(2004)** January.
[7] P. Maymounkov and D. M. Kademlia, "A Peer-to-peer Information System Based on the XOR Metric", http://kademlia.scs.cs.nyu.edu.
[8] S. Rhea, D. Geels, T. Roscoe and J. Kubiatowicz, "Handling Churn in a DHT", Report No. UCB/CSD-03-1299 Computer Science Division (EECS) University of California Berkeley, California 94720, **(2003)** December.
[9] D. Stutzbach and R. Rejaie, "Improving Lookup Performance over a Widely-Deployed DHT", Proceedings of IEEE 2006 Infocom.
[10] R. Jimenez, F. Osmani and B. Knutsson, "Sub-Second Lookups on a Large-Scale Kademlia-Based Overlay", proceedings of IEEE P2P **(2011)**.
[11] P. Rosch, K. -U. Sattler, C. von der Weth and E. Buchmann, "Best Effort Query Processing in DHT-based P2P Systems", Proceedings of the 21st International Conference on Data Engineering (ICDE '05), **(2005)**.
[12] BT DHT Protocol specification http://www.kuqin.com/p2p/20070812/166.html.

Authors

Xiangzhan Yu currently works as associate professor in the School of Computer Science & Technology of Harbin Institute of Technology. His research interests include P2P network, network security, cloud security and privacy preservation.

Xingqi Shang is a graduate student in the School of Computer Science & Technology of Harbin Institute of Technology. His research interests include P2P network, network security, cloud security and privacy preservation.