IMPROVEMENT OF CHORD OVERLAY FOR P2PSIP-BASED COMMUNICATION SYSTEMS

Xianghan Zheng and Vladimir Oleshchuk

Faculty of Engineering and Science, University of Agder, Norway
{xianghan.zheng,vladimir.oleshchuk}@uia.no

ABSTRACT

Chord has been suggested by IETF P2PSIP working group as mandatory overlay technology in the future P2PSIP-based communication systems. Chord allows for the available peer/resource lookup in no more than $\log N$ *hops, where N is the total number of the peers in the overlay network. However, as a protocol originally designed for background downloading applications, Chord has a few drawbacks when supporting P2PSIP real-time communication systems. These drawbacks are related to ID assignment, the relation between ID and physical location, the routing styles and lack of cache, etc. In this paper, we investigate several approaches that can improve the efficiency of the peer/resource lookup algorithm. After that, we first use the formal verification to check the correctness of the proposed Bi-Chord and Recursive routing approaches, and then implement two systems with 512 P2PSIP peers (Chord-based and improved Chord based P2PSIP communication systems) for evaluation. The evaluation includes number of hops, message flows, and practicality from theoretical point of view, and the comparison of the delay in two systems according to the measurement. We get the conclusion that the combination of Bi-Chord, Cache entry record, and Semi-Recursive routing is most suitable for P2PSIP-based communication systems. Finally, we include the conclusions and future work.*

Keywords: Chord, Peer-to-Peer (P2P), Session Initiation Protocol (SIP), P2PSIP

1. INTRODUCTION

P2PSIP WG has suggested Chord protocol as a mandatory underlying overlay technology [1, 2]. In this overlay, each peer maintains a finger table that stores a few successors' connections. Chord routes the message by sending/forwarding messages to the next successor, step by step, until the destination.

However, as a protocol originally designed for background downloading applications, Chord owns several disadvantages when supporting the P2PSIP communication systems. Firstly, Chord lookup protocol is based on clockwise lookup. It causes high delay when communicating with peers that are in the anti-clockwise direction. Secondly, Chord uses consistent hashing (e.g. SHA-1, etc) to partition a keyspace so that each peer is responsible for roughly the same load of resources. However, physically close peers might be assigned with different IDs that are far away from each other in the overlay, and therefore causes longer latency when connected. Thirdly, Chord is implemented in either iterative or recursive style. However, recursive routing might increase the hop number; and iterative routing might be not efficient in traversing NAT (specified in Section 3). Fourth, Chord lacks of cache mechanism to preserve the useful information for future session establishment.

A few attempts have been made to solve the weakness of Chord in P2PSIP communication systems. [3, 4] propose a system model that physically close peers in the overlay are assigned with close peerIDs because most frequently communicated peers are those who are geometry related each other. For instance, peer *M* in Figure 1 is assumed to only (or most frequently)

communicate with the peers in district A and D. Note that our evaluation work in Section 4 is based on this model.

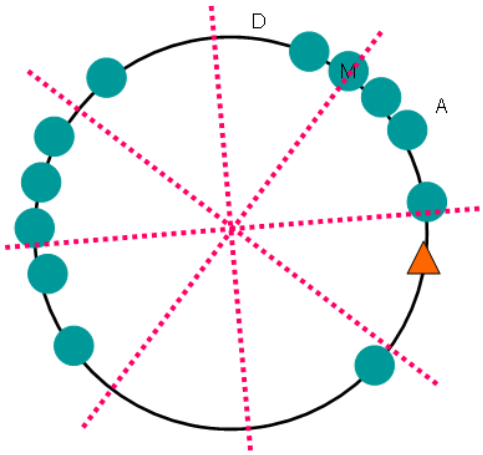

Figure 1. A geometry-related overlay model

In this paper, we study several approaches that could further reduce the communication delay in P2PSIP communication systems. After that, we evaluate these approaches based on the comparison in several aspects: number of hops, message flow, practicality, and tested delay; we get the conclusion that the combination of Bi-Chord approach, Cache Entry Record and Semi-Recursive routing might be a better way for P2PSIP communication systems. Finally, we make the conclusion and describe the future work.

2. BACKGROUND

2.1. Peer-to-Peer SIP

P2PSIP is a future trend that combines internet protocol with the telecommunication protocol. Previous research covers two ways of combining P2P and SIP: P2P over SIP and SIP over P2P. The first approach uses SIP messages to maintain the P2P overlay. The P2P information is encapsulated in the SIP extension header and Session Description Protocol (SDP) body. However, this will require a lot of work in defining the SIP extension protocols. Also, the session layer control message might cause long delays in transport and poor scalability. In the second approach, SIP is implemented upon the functionality of maintaining the P2P network. A separated P2P protocol is proposed to mange the network overlay activities. For example, P2P protocol could define the overlay network algorithms, the TCP and UDP transport protocols, variety of caching, striping, congestion control algorithms and error handling [5]. This optimizes the transport schemes and cause less problem when inter-working with the traditional SIP network. A third solution that combines the above two approaches has been discussed in a few publications and P2PSIP WG [6-10]. The concept is to use the specific link layer protocol to maintain the P2P overlay while define SIP extension protocol to enhance the session layer services and applications, e.g. NAT Traversal, etc.

2.2. Chord Overlay

In Chord overlay, peers and resources construct a ring, as shown in Figure 2. In the ring, peers and resources are represented by an integer Node ID/Resource ID. Each peer stores a certain amount of <*id*, *value*> pairs, in which *id* is the peer/resource ID, *value* is the peer address information or the data storage. Peer/resource ID is assigned by consistent hashing [11], e.g. SHA-1 algorithm. For instance, the peer ID can be produced by hashing the IP address of the

particular peer; and the resource ID can be generated by hashing the data value. The Resource ID is stored in the first peer, whose ID>= Resource ID (see Figure 3).

Each peer contains a routing table, called Finger table, for storing the routing information records. The Finger table records logN successors where N is the number of peers in the overlay (see Figure 4). Suppose the space size of overlay is 2^m, for some integer m and the i-th successor ID of a peer with ID P is:

$$Succid(i) = (P + 2^{i-1}) \bmod 2^m \ (0 < i \leq m)$$

Each peer contacts periodically its successors for updating the Finger table. It also contacts the predecessor that is the previous peer in the identifier circle. This is useful when a peer leaves the ring and asks the previous peer to update its Finger table.

Chord routes the message by sending messages to the next successor that is nearest to the destination identifier. Consider an example, when peer 3 is searching peer 28 (Figure 5.). The peer 3 would first check its finger table records; choose a successor (peer 22) nearest to the destination, and then send a request to this successor. The peer 22 would also check its own finger table and forward the message to its successor (destination peer 28). The total cost is no more than logN hops and $\frac{1}{2}\log N$ in average where N is the number of peers in the overlay [12].

Chord also defines the advertisement function about joining/leaving procedure for peers. The advertisement function would tell the corresponding successor and predecessor to update their finger table.

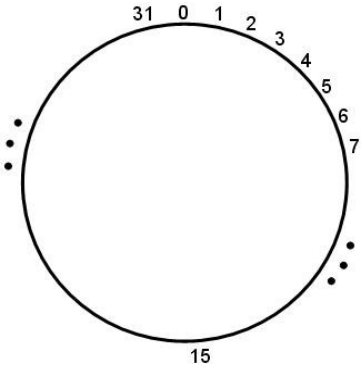

Figure 2. Chord Ring

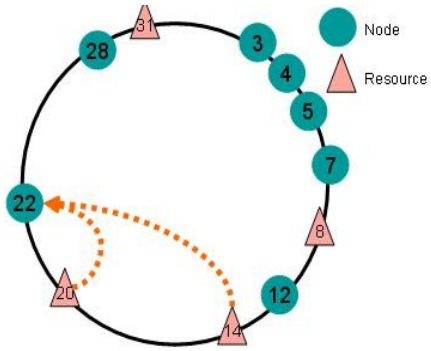

Figure 3. Chord Storage

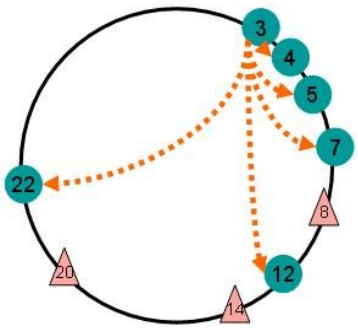

Figure 4. Direct Connection

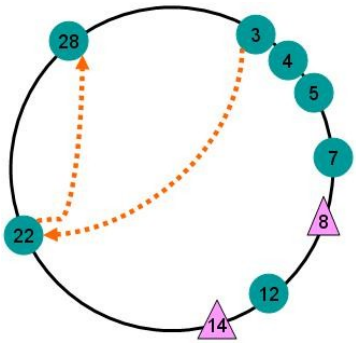

Figure 5. Message Flow

3. IMPROVEMENT OF LOOKUP APPROACHES

In this section, we propose four approaches that could improve the resource/peer lookup efficiency in P2PSIP-based communication systems. Then, we evaluate these approaches and get the conclusion that the combination of Bi-Chord, Cache entry record and Semi-Recursive routing is the most suitable for P2PSIP communication systems.

3.1. Bi-direction Chord (Bi-Chord)

One simple solution is to use two directional lookup mechanism in the search of peer/resource, called Bi-Chord [13]. In this solution, suppose the overlay space is 2^m, each peer stores m successor and (m–1) predecessor records in its Finger table. The P2PSIP request is sent/forwarded to one of the successors/predecessors that is clockwisely closest to the target and then forwarded step-by-step until the destination is reached.

Figure 6 shows the connections of a peer with identifier 3. It holds five connections with its successors (peer 4, peer 5, peer 7, peer 12, and peer 21) and two connections with its predecessors (peer 1 and peer 28). For searching a peer, for instance peer 30, peer 3 firstly send the P2PSIP request to the peer 28; peer 28, after using its own finger table, forwards the request to the destination peer 30, as represented in Figure 7.

This approach reuses most of the Chord lookup algorithm and routing style. According to the algorithm in section 2, it takes $(\log(N)-1)/2$ in average before the message reaches the destination, where N is the number of peers in the overlay.

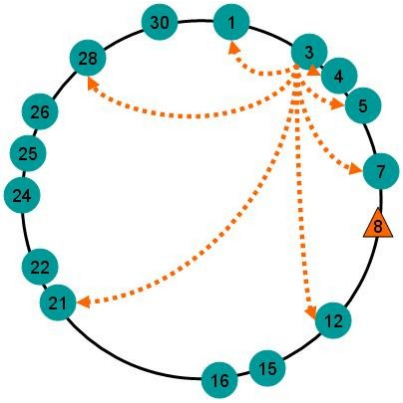

Figure 6. Bi-Chord overlay

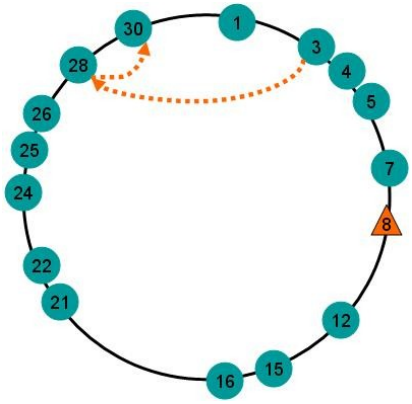

Figure 7. Message forwarding

3.2. Bi-Greedy Lookup

The greedy algorithm may be implemented to further enhance Bi-Chord efficiency. In this approach, each peer maintains the same finger table as Bi-Chord (Figure 8). The major difference is that each peer transmits the P2PSIP request to one of its successors/predecessors that is as close as possible to the destination, independent of the clockwise or anti-clockwise direction. Each peer chooses either its successor or predecessor for routing messages, on the distance basis.

Suppose peer A wishes to initiate/forward a P2PSIP message to the destination B, it chooses the shorter distance of

(a) One of its predecessors closest to peer B.
(b) One of its successors closest to peer B.

If these two choices have equal path lengths, the message will follow the rule b.

Figure 7 shows an example of the Bi-Greedy lookup initiated from peer 3 and ended in peer 24. The message is firstly routed to peer 21 (peer 3's successor), then to peer 25 (successor of peer

21), and finally to peer 24 (predecessor of 25). According to [14], the average path length of Bi-Greedy algorithm is: $\log(N)/2 - \sqrt{\log(N/2\pi)} + 1$

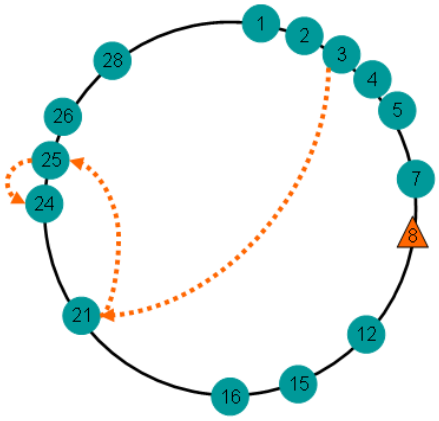

Figure 8. Bi-Greedy lookup routing

3.3. Cache Entry Record

We can also use a cache entry record approach to improve performance indirectly. The concept is as following: P2PSIP peer in the overlay maintains a cache that records the communication history details (see Table 1), including the previous communicated peer identifier, the corresponding public IP address, port, etc. For searching the destination peer, source peer first check its cache entry record. If the destination peer (peer identifier, public IP address, port, etc) is in the table, the session might be established directly. Otherwise, the source peer will execute the lookup algorithm described above.

In the stable overlay where peers do not change the identifier and the public endpoints frequently, the cost is only one hop. However, in the unstable overlay where peers change their identifier/IP frequently, this might even cost worse delay. It takes at most ($\log N + 1$) hops (e.g. Bi-Chord lookup) before reaching the destination.

Table 1
Cache Entry Record

(id==3)	**identifier**	**Public endpoint**
1	A	215.239.168.1:1980
2	B	159.250.16.2:8000
.	.	.
N	S	128.39.169.2:9000

3.4. Semi-Recursive Routing

There are three kind of routing options suggested by P2PSIP WG: Iterative, Recursive, and Semi-Recursive [6-8]. In this paper, we suggest using Semi-recursive routing (Figure 9) for P2PSIP Communication systems because it reduces the number of hops and lowers the latency. In addition, Semi-recursive routing is more resilient from failures of intermediate peers than recursive mode.

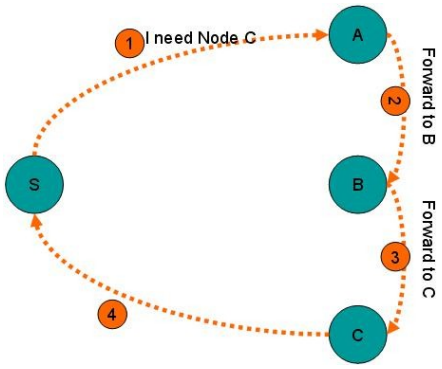

Figure 9. Semi-Recursive routing

4. Formal valification and Prototype Simulation

4.1. Formal verification

Before prototype implementation, we simulate a small overlay with 16 peers and then use the state based formal verification to check the correctness of the proposed Bi-Chord approach from logical point of view. The language used to model the algorithm is based on PROMELA (Protocol Meta Language). The tool used is Spin, which is a software for the state based formal verification of distributed system [10]. We regard the overlay is stable and without the maintenance functions (e.g. update the finger table, peer joining and leaving, etc).

Fig. 10 shows a use case scenario that two concurrent request processes send out the P2PSIP request and these request messages follows the Bi-Chord algorithm to the destination peer. The response returns in the Semi-Recursive style.

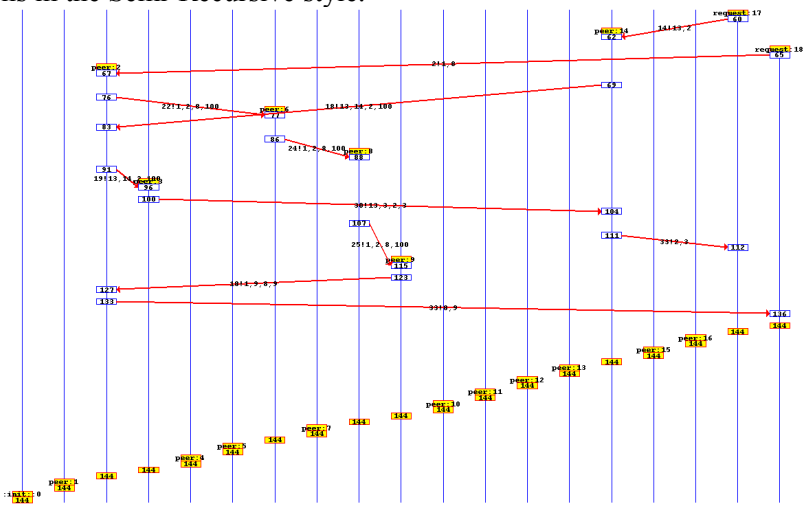

Fig.10. Message flow

We also use LTL (Linear Temporal Logic) to formulate correctness requirement. We define a LTL sentence for the property validation: [] (p -> <> q) where p is the request sending out from the request process and q is the response received. By using Spin we have shown that Bi-Chord protocol satisfies correctness requirement described above.

4.2. Prototype Implementation

In order for performance comparison and evaluation (specified in Section 5), we simulated two systems: Chord-based P2PSIP system and Improved Chord based P2PSIP system. Chord-based P2PSIP system follows the original Chord lookup protocol; as to the Improved Chord based

P2PSIP system, Bi-Chord (the reason why to implement this approach will be specified in Section 5), Cache entry record and Semi-Recursive routing are implemented.

Firstly, we simulate a Chord-based P2PSIP system. This system contains 512 P2PSIP peers in the overlay with the space size 2048. We use Java to create an application that owns 512 threads, each of which represents one P2PSIP peer. We configure the successors for each peer in a background database so that when initiation, peer could fetch it. A P2PSIP Peer management center (see Figure 11) is implemented to create the P2PSIP peer threads (Figure 12) and configure the peer attributes (e.g. peerID and the opening port). Each peer uses the loopback address (127.0.0.1) as its IP address (can be also 192.168.0.100 in NAT) and opens a specific port (e.g. 9001, etc) for receiving the connection. Two potential P2PSIP messages are defined for testing: "INVITE" (See Figure 13) as P2PSIP request and "180 Ringing" as P2PSIP response (Figure 14). For each P2PSIP Peer thread, it can send out the "INVITE" message when clicking "Search" button and receive the response in the background.

Then, based on Chord-based P2PSIP system, we build an improved system with the feature of Bi-Chord, Cache entry record and Semi-Recursive routing. The Chord lookup protocol is revised to realize Bi-Chord; the open source Apache Derby [14] is chosen as the embedded database for cache entry record implementation; finally the routing style is changed to Semi-Recursive routing.

Two systems are deployed separately on a platform with Windows XP professional system, 2*2.4G Intel Core CPU and 3G memory. We use the Wireshirk [15] to monitor the message transmission. The testing shows that two systems work well.

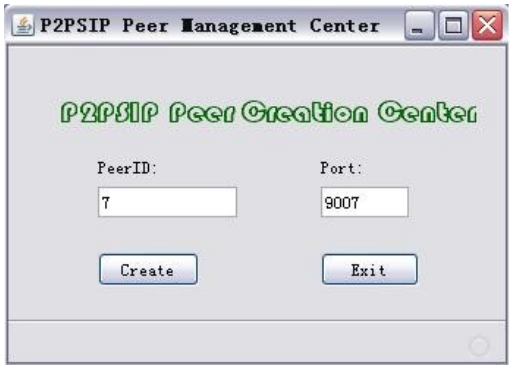

Figure 11. P2PSIP Peer Management center

Figure 12.P2PSIP Peer

```
INVITE 20 P2PSIP/2.0
Max-Forwards:10
From:3
To:20
Call-ID:472721
CSeq: 1 INVITE
Contact:3
Via:3 127.0.0.1:9003;
```

Figure 13. P2PSIP INVITE

```
P2PSIP/2.0 180 Ringing
To:20
From:3
Contact:20
CSeq: 1 Response
Content-Length: 0
Via:3 127.0.0.1:9003;
20 192.168.0.101:9020;
```

Figure 14.P2PSIP 180 Ringing

5. EVALUATION

Our evaluation is based on the model of Figure 1. We assume that the overlay is divided logically into equally S parts and peer *M* only communicate with peers in district A and D.

Note that our evaluation does not consider the case of churn (when P2PSIP peer join and leave the overlay) and NAT traversal problems.

5.1. Num of hops

Based on the Chord lookup protocol described in Section 2, the average complexity of Chord lookup algorithm is ½ $\log(N/S)$ in the area A. As to the area D, it takes $\log S$ hops before the message arrives to the left boundary of area D. Therefore, the average complexity is ½ $\log(N/S)/2+\log S$. Bi-Chord provides fairness in bi-directional peer/resource lookup. Thus, the complexity is ½ $\log(N/S)$ in both area A and D. Bi-Greedy further improves the lookup efficiency. According to the revision from previous research (in Section 3), the average complexity is: $\log(N/S)/2-\sqrt{\log(N/2\pi S)}+1$.

We compare three lookup algorithms by setting different S value (e.g. we set S = 8 and S = 16 for example), as represented in Figure 15. X axis represents the peer number in the overlay and Y axis represents the number of hops in average. We get the information that firstly, Bi-Chord and Bi-Greedy lookup algorithms are much more efficient than the original Chord lookup; secondly, the higher value of S, the smaller number of hops; thirdly, Bi-Greedy approach provides better result than Bi-Chord.

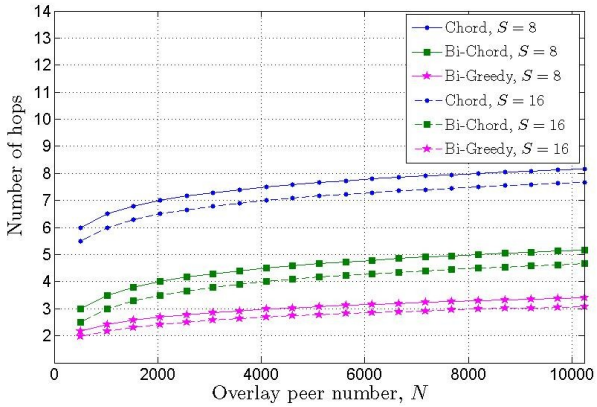

Figure 15. Three algorithms comparison

5.2. Number of message flows

We also compare the number of message flows in two types of message routing: Recursive and Semi-Recursive (As shown in the Table 2). Generally, the message number in recursive routing style is two times than Semi-recursive routing.

Table 2
Message flow comparison

	Recursive	Semi-Recursive
Num-of-Message (at most)	2 $\log(N/S)$	$\log(N/S)$ + 1
Num-of-Message (In average)	$\log(N/S)$	½ $\log(N/S)$ + 1

5.3. Cache Entry Record

Bi-Chord approach preserves most of the original Chord lookup and routing mechanisms. It should be easy to be implemented in reality. However, Bi-Greedy does not mean the best opinion. The enhancement of Bi-Greedy algorithm is limited comparing with Bi-Chord (about one hop difference). Additionally, the malicious peer in Bi-Greedy approach will be able to

send/forward the message in both directions and makes the debug and trace even more difficult. Furthermore, the extra function of accurate distance calculation and comparison might on the other way, add the burden for P2PSIP peer and offset the enhancement of Bi-Greedy. In summary, we advocate Bi-Chord approaches.

Cache record entry approach is a common solution in today's applications and systems for recording useful data, e.g. communication history, etc.

5.4. Delay Comparison

Finally, the delay of two systems is measured. We choose one peer (here we use peer 586 for example) as the source peer and select 8 destination peer groups, each of which contains 32 random selected peers in either district A or D. (We assume that S=8 and therefore both of district A and D have 64 P2PSIP peers). For each system, we initiate 32*8 P2PSIP request (from group 1 to group 8) manually from peer 586 and measure the delay of the response. After that, we calculate the average delay for each group, as represented in Figure 16.

In Chord-based P2PSIP system, the delay of each group is almost the same (about 15ms-16ms); however, in the improved Chord based P2PSIP system, the delay is greatly reduced, especially when the number of testing increases. We believe it is the contribution of the Cache entry record.

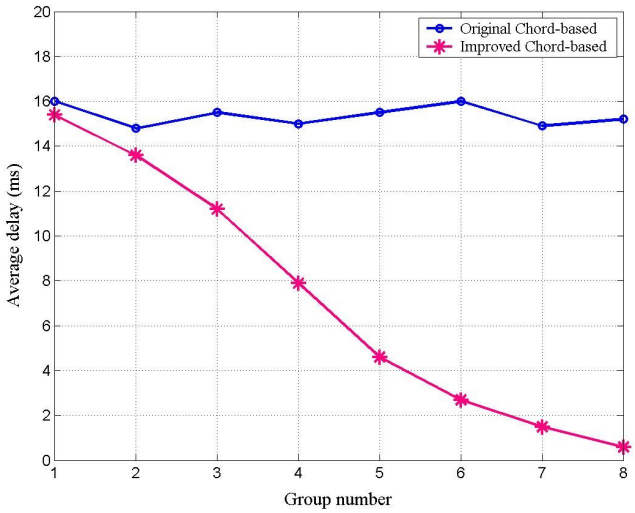

Figure 16. Delay comparison

After considering all the situations, we get the conclusion that the combination of Bi-Chord, Cache entry record and Semi-Recursive routing might be the best choice for the P2PSIP communication systems to reduce the number of hops and delay.

6. Conclusion and Future work

In this paper we propose several approaches to improve the peer/resource lookup efficiency of Chord protocol. After evaluation, we conclude that the best choice for P2PSIP communication systems is to combine Bi-Chord, Cache entry record, and Semi-recursive routing approaches together. We use Spin formal validation to check the correctness of the lookup algorithm. Then we have also implemented a prototype.

For further improving the peer/resource lookup efficiency in P2PSIP communication systems, a hierarchical overlay architecture proposed in [17] might be a possible solution.

However, the decentralization of P2PSIP might cause many privacy problems. A possible solution would be to use a proxy-based peer to relay the data traffic between the source peer and the destination peer [18].

Furthermore, we plan to study the browser-based approach [19] that might also provide secure approach and support many devices (e.g. many mobile phones, PDA, etc) that lack coherent support for P2PSIP services.

REFERENCES

[1] *P2PSIP*. p. http://www.p2p-sip.org.

[2] Ion, S., et al., *Chord: a scalable peer-to-peer lookup protocol for internet applications.* IEEE/ACM Trans. Netw., 2003. **11**(1): p. 17-32.

[3] Guangyu Shi, Y.L., Jian Chen, Hao Gong, Hongli Zhang, *T2MC: A Peer-to-Peer Mismatch Reduction Technique by Traceroute and 2-Means Classification Algorithm*, in *7th International IFIP-TC6 Networking Conference*. May 5-9, 2008: Singapore.

[4] Huang, L., *Location and Discovery of Subsets of Resources.* Internet - Draft (working in process), July, 2008: p. http://tools.ietf.org/html/draft-licanhuang-p2psip-subsetresourcelocation-00.

[5] Frank Dabek, J.L., Emil Sit, James Robertson, M. Frans Kaashoek, Robert Morris, *Designing a DHT for low latency and high throughput.* NSDI 2004.

[6] D. Bryan, P.M., E. Shim, D. Willis, S. Dawkins, *Concepts and Terminology for Peer to Peer SIP.* draft-ietf-p2psip-concepts-02, July, 2008.

[7] C. Jennings, B.L., E. Rescorla, S. Baset, H. Schulzrinne, *REsource LOcation And Discovery (RELOAD).* draft-bryan-p2psip-reload-04, June, 2008.

[8] G. Camarillo, P.N., J. Hautakorpi, *HIP BONE: Host Identity Protocol (HIP) Based Overlay Networking Environment.* draft-camarillo-hip-bone-01, Feb, 2008.

[9] David A. Bryan, B.B.L., Marcia Zangrilli, *The Design of a Versatile, Secure P2PSIP Communications Architecture for the Public Internet*, in *IEEE International Symposium on Parallel and Distributed Processing, IPDPS*. April, 2008.

[10] Marcin Matuszewski, E.K., *Mobile P2PSIP - Peer-to-Peer SIP Communication in Mobile Communities*, in *5th IEEE Consumer Communications and Networking Conference*. Jan. 2008.

[11] David, K., et al., *Consistent hashing and random trees: distributed caching protocols for relieving hot spots on the World Wide Web*, in *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*. 1997, ACM: El Paso, Texas, United States.

[12] Jani, H. and C. Gonzalo, *Evaluation of DHTs from the viewpoint of interpersonal communications*, in *Proceedings of the 6th international conference on Mobile and ubiquitous multimedia*. 2007, ACM: Oulu, Finland.

[13] Junjie Jiang, R., Changyong Liang and Weinong Wang, *BiChord: An Improved Approach for Lookup Routing in Chord.* Lecture Notes in Computer Science 2005: p. http://www.springerlink.com/content/egj6fpw0w8e136tm/.

[14] Prasanna, G. and M. Gurmeet Singh, *Optimal routing in Chord*, in *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*. 2004, Society for Industrial and Applied Mathematics: New Orleans, Louisiana.

[15] *Apache Derby.* p. http://db.apache.org/derby/.

[16] *Wireshark: Go deep.*: p. http://www.wireshark.org/.

[17] Lifeng Le, G.-S.K., *Hierarchical and Breathing Peer-to-Peer SIP system.* Communications, 2007. ICC '07. IEEE International Conference on, June, 2007.

[18] Xianghan Zheng, V.O., *Providing Privacy in P2PSIP-based Communication Systems.* Norsk informasjonssikkerhetkonferanse, Sep, 2008.

[19] Xianghan Zheng, V.O., Hongzhi Jiao, *A System Architecture for SIP/IMS-based Multimedia Services* in *International Conference on Telecommunications and Networking (TeNe 07)*. Dec, 2007.