# Cone: A Topology-Aware Structured P2P System with Proximity Neighbor Selection

Wang Huijin , Lin Yongting
*Department of Computer Science, Jinan University, Guangzhou 510632, P.R.China*
*twang21cn@21cn.com*

## Abstract

*Traditional peer-to-peer (P2P) overlay networks such as Chord, CAN, Pastry and Tapestry provide a type of novel platform for a variety of scalable and decentralized distributed applications, whilst they bring tremendous delay to network routing due to less care of network topology structure. In this paper, we propose a structured P2P system with low network delay, named Cone, which is extended to enable physical topology aware and applicable to routing on the base of Chord. In the Cone, nodes are divided into groups according to prefixes of their identifiers, finger tables are set bidirectional, and associated pointers are indicated to the nearest nodes in relevant groups; By landmark+RTT method proximity information is generated and proximity neighbor selection is applied to exploit proximity information; And the routing algorithms, node arrival and departure mechanisms, and fault tolerant are designed and tested. The results of simulation experiments suggest that the Cone's performance is obviously improved in the delay of routing and the hops of overlay network by contrast with the one of Chord.*

**Keywords** peer-to-peer, topology-aware, proximity neighbor selection

## 1. Introduction

Recently, the emergence of DHTs (distributed hash table) promotes the development of peer-to-peer (P2P), and P2P has become a research hotspot in fields of computer network and distributed system. Several recent systems (Chord [12], CAN [7], Pastry [8] and Tapestry [16]) provide a self-organizing substrate for large-scale peer-to-peer applications. These systems can be viewed as providing a scalable, fault-tolerant distributed hash table.

In these DHTs systems, nodes and resources are allocated a unique identifier (id or key). Nodes are responsible for a bound of resource. Assuming a network consisting of N nodes, each node only maintain a small routing table, $O(\log N)$ entries is normal for routing table; they can achieve a routing length of $O(\log N)$ while publishing resource or querying resource. Because of a good ability of scalable and fault-tolerant, DHTs has been used for application platform, such as CFS [2] (Cooperative File System) using Chord and PAST [9] using Pastry.

Comparing with unstructured P2P system, DHTs are more efficient systems since they can locate targets within hops of a finite number. However, one hop in overlay network may involve several hops in physical network. The Number of hops in underlying network, related with a single hop of overlay network, depends on the "distance" between source node and target node. In fact, physical routing hops induced by intra-LAN and inter- continent are really different.

As can be seen, although the target node can be located in a logarithmic overlay hops, the physical path traveled during the overlay routing is often less than optimal. Nearly 70 percents overlay network mismatch with its physical network. Hence, Combining DHTs and network proximity is a new research field in structured P2P system, and many methods were proposed.

This paper proposes a topology-aware structured P2P system — Cone, which implements proximity neighbor selection over Chord system. Nodes are

divided into groups by identifier, and nodes in finger table entry point to the nearest node among its group. Therefore, routing process of The Cone takes both identifier space and physical network into account.

## 2. Related works

To solve mismatching between overlay topology and physical network topology in P2P system, several methods were proposed, including [14]: (1) generating physical network proximity information, (2) exploiting physical network proximity information to construct overlay network. The main reason for mismatching in structured P2P system is no care of the proximity between nodes in physical network. Three methods to generate proximity information were proposed, including expanding-ring search [6], heuristics and landmark clustering. In addition, how to exploit proximity information is a key problem for constructing structured P2P, and current related Methods [1, 10] mainly consist of geographic layout, proximity routing and proximity neighbor selection.

Landmark clustering was frequently used to build topology-aware overlay network. Ratnasamy et al. [11] use landmark clustering to build topology-aware CAN for exploiting proximity topology. Landmark+RTT approach was proposed in [14], because that proximity information generated by landmark clustering is not accuracy and that it was not very effective in differing nodes within an equal landmark order. Conventional landmark clustering, as introduced above, suffers from the premise limitation of a set of fixed, stationary landmark nodes, and this obviously introduces single point failure in overlay network. After this, an approach of Random Landmarking (RLM) was proposed to solve this pitfall [13]. In RLM, landmark nodes are responded for landmark keys using the overlay lookup procedure and serve as temporary landmarks for a joining node.

PChord [5] make another improvement on Chord to achieve better routing efficiency via proximity routing algorithm. Routing table in PChord mainly includes Finger table and Proximity List, finger table in PChord is the same as in Chord without considering physical network, and nodes included in proximity list have a small latency with current node. As the proximity routing, its routing strategy not only depends on overlay network, but also on physical network. The key modification of routing algorithm in PChord is the choice of a next hop. The next hop is decided by the entries both in finger table and proximity list: the node in finger table or proximity list, which is small to the key and the closest to the key among all nodes in identifier space, will be chose as the next hop.

Recently, geographic layout has been used to achieve effective routing mechanism in Chord. Chord6 [15] utilizes the hierarchical structure of IPv6 address to produce a node's id so that the nodes in the same portion have close ids. AChord [3] archives routing mechanism of topology-aware by employing anycast of IPv6. The main modifications of both Chord6 and AChord are that node's id is allocated in a new way instead of random distributing. Geographic layout provides network proximity in the routing, whilst it has several drawbacks. It destroys the uniform population of the id space, which causes load balancing problems; future more, neighboring nodes in the id space are more likely to suffer correlated failures, which can have implications for robustness and security.

## 3. Core Design of Cone

### 3.1. Nodes' identifier of Cone

Nodes' identifier space is a two-layer identifier in the Cone, i.e. Chord-layer identifier layer and Cone-layer identifier layer. Chord-layer identifier is the same as nodes' identifier in Chord, which is generated by mapping node's IP address into an identifier of m bits using a hash function such as SHA-1. The definition of Chord-layer identifier for the Cone is detailed as follows:

Maximum of nodes $N = 2^m$

Identifier for node id = SHA-1(IP) mod N

As showed above, the identifier of Chord-layer is called id, and is generated by SHA-1 function using IP address as variable. All nodes are agenized into a Chord-ring according to its id for Chord-layer identifier space. Node only keeps its successor and predecessor, and has the same maintenance mechanism as Chord.

In Cone-layer identifier, group concept is introduced. Nodes having the same prefix in Chord-layer identifier are divided into a group, and the prefix of identifier is the group identifier. Thereby, the identifier of Cone-layer includes gid and lid, and the first indicates the relevant group the node belonged to, which indicates the local identifier in group. Definition of Cone-layer identifier is showed as follows:

group identifier $gid = id / 2^g$

local identifier $lid = id \bmod 2^g$

(*g* denotes a configurable argument)

In Cone-layer identifier, nodes with the same gid are arranged to a small ring for its lid, and all of small rings are arranged to a big ring for small rings' gid. Cone overlay structured is illustrated in fig.1.

Any published resource with identifier as key, is restored in the node whose gid is $key/2^g$ and lid is $key \bmod 2^g$. If this node is not exist, the resource will publish to some certain node whose gid is the successor of $key/2^g$ in the big ring and whose lid is the successor of $key \bmod 2^g$ in the small ring.

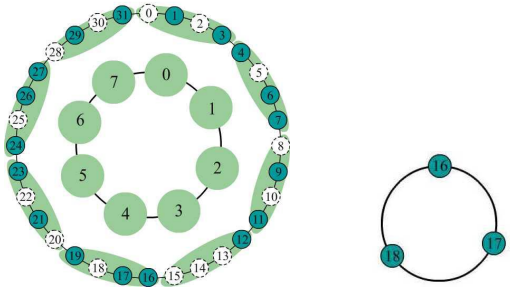

Figure 1: The Cone's logic architecture (the outside of left part is the chord ring of Chord layer and the inside is big ring of Cone layer; the right part is the small ring of group 4).

For the sake of making use of topology-aware information, landmark+RTT approach is used to help construction of the Cone. It is assumed that there are landmarks with k numbers, which distributed over the Internet. Prior to joining the overlay network, a joining node has to measure its RTT to all landmarks and then orders the landmarks according to its RTT.

### 3.2. Routing Table of Cone

In Cone overlay network, Landmark+RTT is used to generate topology-aware information and proximity neighbor selection is used to exploit topology-aware information. Routing table in the Cone includes front finger table, back finger table and group table. Nodes in the entry of front finger table or back finger table have a near distance with the current node in physical network (measured by RTT).

Front finger table is composed of *m-g* entries and each entry is defined as follow:

In the same way, back finger table is composed of *m-g* entries and each entry is defined as follow:

| symbol | definition |
|---|---|
| backFinger [i].start | $gid - 2^i \bmod 2^{m-g}$, points to a group with a distance of $2^i$ over the group of current node in anti-clockwise. |
| backFinger [i].node | If group .start exists in overlay network, then .node points to the nearest node in the group with current node in physical network. If not, then .node points to the node which is in the first group after group .start in anti-clockwise and which is the nearest node within the group in physical network. |

Group table maintains other online node information of current node's group. The size of group table is the number of online nodes within the group with a maximum of $2^g$. According to the character of DHT, nodes' ids are distributed equably in the ID ring space, and that there are $(2^g/2^m) * M$ nodes per group. In the other way, nodes maintained in the group table are close to a current node in ID space, but are far from a current node in physical network.

### 3.3. Routing algorithm

Identifiers in Cone are similarly two-layer identifiers, who with equal length prefix are divided into a group. A group can be aware of other groups by both front finger table and back finger table. A two-layer routing algorithms are proposed to accord with characters of the Cone, i.e. inter-group routing algorithm and intra-group routing algorithm, and in the first one a query is to locate a special group which associated resource is belonged to; and in the second one a query is to locate a node in the special group where associated resource is stored.

Inter-group routing algorithm: the next hop in inter-group routing algorithm only depends on front finger table and back finger table. A request is forwarded to nodes of difference groups, which are close to the current node. Description of routing algorithm is listed as follow:

Assume node A receives a query or requests a query with Key as target resource. If key.gid $\in$ (backFinger[0].node.gid, gid], then inter-group algorithm is finished and intra-group algorithm is to be started on node A; if key.gid $\in (gid, frontFinger[0].node.gid]$, then the query is forwarded to frontFinger[0].node and intra-group algorithm is to be started on frontFinger[0].node.

| symbol | definition |
|---|---|
| frontFinger [i].start | $(gid + 2^i) \bmod 2^{m-g}$, points to a group with a distance of $2^i$ over the group of current node in clockwise. |
| frontFinger [i].node | If group. start exists in overlay network, then .node points to the nearest node in the group with current node in physical network. If not, then .node points to the node which is in the first group after group .start in clockwise and which is the nearest node within the group in physical network. |

If not, follow step 2.

The query is forwarded to the node among nodes in both front finger table and back finger table with the closest distance in gid from the current node.

Pseudo code of inter-group algorithm describe as following:

```
Node A.findNearPeer(Key)
{
        If
Key.gid ∈ (backFinger[0].node.gid, gid]
                return A;
        if
Key.gid ∈ (gid, frontFinger[0].node.gid]
                return
frontFinger[0].node..findNearPeer(Key)
        Select T ∈ frontFinger[i].node ∪
backFinger[i].node
        |T.gid – Key.gid| = min(|node.gid – Key.gid|)
        return T.findNearPeer(Key)
}
```

Intra-group algorithm: the next hop that is also the last hop in each routing only depends on the group table. The query is forwarded to node where the resource is stored. Pseudo code of intra-group algorithm describe below:

```
Node A.findStoredPeer(Key)
{
  //assume that group table are stored in a decrease
order called group
  //gNum is the number of entry of group table
        for i =0 to gNum – 1
                if
Key.lid ∈ (group[i].lid, group[i+1 mod gNum].lid]
        Return group[i+1 mod gNum];
}
```

In the Cone, routing algorithm is composed of inter-group algorithm and intra-group algorithm. The routing algorithm can be described as follow:

```
  A.Lookup(Key)
{
  Return A.findNearPeer(Key).findStoredPeer(Key);
}
```

Here, we use a lookup procedure to represent the routing algorithm, findNearPeer procedure to represent the inter-group algorithm and findStoredPeer procedure to represent the intra-group algorithm.

In general, The Cone's routing scheme can result in some optimization below:

Finger tables (front finger table and back finger table) are bidirectional so that the number of hop in the Cone is shorted than in Chord.

Because nodes maintained in finger table are close to the current node that hops in inter-group algorithm can achieve a low network delay. Hence, network delay arose in routing of Cone is lower than of Chord.

### 3.4. Self-organization and adaptation mechanisms

In this section, we describe procedure for handling the arrival and departure of nodes in Cone overlay network. We begin with the arrival of a new node that joins the system.

*Node arrival*

When a new node arrives, it needs to initialize its identifier, sort of landmarks and routing table. The procedure of node arrival consists of four steps and listed in the following:

*Node initialization:* node gets to know its identifier for Chord-layer and Cone-layer identifier space using SHA-1 function with its IP address or related information. At the same time, it measure RTT to all landmarks and get the RTT sorting as topology information. Let us assume that the new node's id is X and that the new node knows initially about a nearby Cone node A (using IP method such as IP multicast). Subsequently, node X routs a findNearPeer message with the key equal to X , receives a list of nodes which are group members of X, and fill them into the group table and notifies them node X' arrival.

*frontFinger table initialization:* for initialing entry i of front finger table, Node X routs a findNearPeer message with the key equal to $frontFinger[i].start * 2^g$ with assuming the routing process is terminated at node Z. On receiving Node X's message, node Z selects a list of nodes from group table which have the same landmarks sorting with node X as a response for X, if no node exists, node Z itself is selected. Then node X measures its RTT to the list of node and the node with the smallest in RTT will be filled into entry i of front finger table.

*backFinger table initialization:* the initialization of backFinger table is similarly to frontFinger table. For initialing entry i of front finger table, Node X routs a findNearPeer message with the key equal to $\text{backFinger[i].start}*2^g$ with assuming the routing process is terminated at node Z, if Z.gid is not equal to $\text{backFinger[i].start}$, then this message is forwarded to Z.backFinger[0].node, Z is remarked as Z.backFinger[0].node. Node in group Z.gid is to be filled into entry i of backFinger table with the method described in preceding step.

*Notifying nodes:* nodes in front finger table and back finger table are notified of node X' arrival and update their routing table dynamically.

The Cone uses measuring RTT between current node and nodes in a special group, that have the same landmark sorting with current node, to find the nearest node for routing table. After joining the cone overlay network, the new node get a correct routing table which are topology-aware.

*Node departure*

Node in cone overlay network may depart or fail without warning. In this section, we discuss how the Cone overlay network handles node departures; node failure without warning is to be discussed in next section.

At first, the departing node (signed as A) sends the departure message to its group members for informing its departure. If there are other nodes in the same group, then the system selects a list of node which have the same landmark sorting with frontFinger[i].node, and sends this list to forntFinger[i].node and informs its departure. For entrys of backFringer[i], the similarity, is done as frontFinger[i]. If there is no other node in the group, node A will send backFinger[0].node to frontFinger[i].node, then send frontFinger[0].node to backFinger[i].node for informing group departure. The nodes received the node/group departure will update their routing table dynamically by the method mentioned above.

## 4. Experiments and result analysis

In order to examine the behavior and performance of the Cone, we perform large-scale experiments by simulation to the Cone and Chord, respectively. We constructed our simulator on Transit stub topology [17] generated by GT-ITM software package. There are five different network topologies in our experiment and all of them include 4 transit domains, 4 nodes within a transit domain, and 8 stub domains per transit domain node. The number of nodes in stub domain is different with each other, and is 8, 32, 128, 512, and 2048. It is assumed that Cone nodes are distributed in stub domain, and nodes in a transit domain are routers of IP network. During this simulation, 50 query requests are invoked per node and destination node are generated at random. The argument of the Cone is that m is 20 and *g* is 8.

### 4.1. Routing hop in Cone

In P2P overlay network, average routing hop and average routing distance are indicator of routing performance. The average routing hop is the average forward times invoked by a routing process and the average routing distance is the average distance tracing over the path from source node to target node for routing process. At some time they are called as routing hop and routing distance. Routing distance is to be discussed in the next section.

The different between the Cone and Chord qualify different routing performance over them. For the Cone, assume N is the number of online nodes. if $N<(2^m/2^g)$, there are average one per group and hence there are N groups over the overlay network; if $N\geq(2^m/2^g)$, the average number of online node within a group is larger than 1 and hence there are $2^m/2^g$ groups. Research [4] indicated that the routing hop is $1/3\log(N)$ in Chord system with bi-directed Finger tables. As discussed above, Cone execute inter-routing algorithm using bi-directed finger table among groups, so routing hop of inter-group routing algorithm, depending on number of groups, is $\frac{1}{3}\log(G)$, there G is the numbers of groups. During the intra-group routing algorithm, routing request can be forwarding to destination node within one hop by group table. Hence, there relationship between routing hops and numbers of online node can be formulated as follows:

$$R(N)=\begin{cases} 1/3\log_2^N+1 & N<2^m/2^g \quad (1) \\ 1/3\log_2^{2^m/2^g}+1=\dfrac{m-g}{3}+1 & N\geq 2^m/2^g \quad (2) \end{cases}$$

Comparison of routing hop between Cone and Chord is illustrated in figure 2.

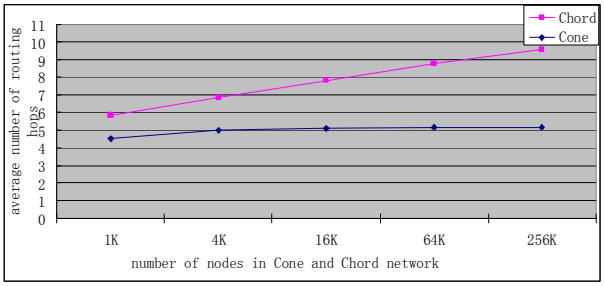


Figure 2: Average number of routing hops in Cone and Chord.

We can make a conclusion that routing hop in Cone is smaller than in Chord. When the number of nodes arrange from 1K to 4K with satisfying the first condition, routing hop become large with the number of nodes increasing, because there is one node per group in average; when the number of nodes is larger then 4K with satisfying the second condition, routing hop keeps balance with number is $(m-g)/3 + 1$, because that all of groups have become online.

### 4.2. Routing distance in Cone

The original idea of this paper elicits from a fact that Chord has a tremendous network routing delay, resulting from takes physical network topology into no account. To make use of topology-aware, structured P2P system-the Cone is proposed over Chord, and proximity neighbor selection method is used to exploited proximity information. Experiments show that: After taking topology into account, the Cone has an obvious advantage over Chord in routing distance. Comparison of routing distance between Cone and Chord is illustrated in figure 3.

Figure 3 shows that routing distance increases with the number of nodes increasing in Chord, because routing hop is increasing and average distance in each hop is almost keeping the same when the number of nodes is increasing. The Cone has a different character in routing distance that can be made from figure 3. As we can see from the figure 3, when the number of nodes in Cone is less than a special number (4K), routing distance increases with the number of nodes increasing, but incensement is smaller than Chord; when number of nodes is larger than 4K, routing hop keeps in the same with the number of nods increasing. At the same time, nodes in both front finger table and back finger table have a more and more short distance with the current node. Hence, routing distance decreases with number of nodes increasing when number of nodes is larger than 4K. When the number of nodes arrange from 4k to 256K, routing distance arrange from 112 to 40.

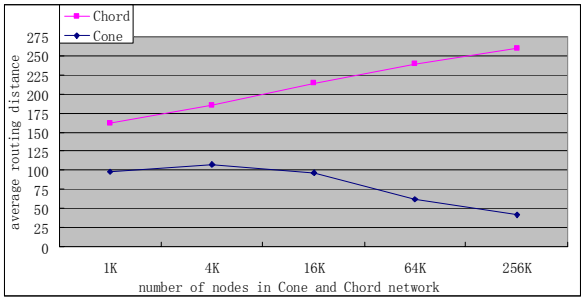


Figure 3: Average number of routing distance in Cone and Chord.

Please note that when the number of nodes increases 256K in the Cone with $m$=20 and $g$=8, the ratio of online node has achieved 25%, but the routing distance has become a low value. The Cone makes use of physical network topology to improve performance.

### 4.3. Decomposition of routing path in Cone

Routing in the Cone consist of inter-group routing and intra-group routing, the decomposition of routing path is illustrated in figure 4 with different number of nodes. As shown in the figure 4, there is no obvious change in routing distance in intra-group routing, but inter-group routing make a big progress in routing distance when the number of nodes is increasing. For the number of nodes in each group is increasing while the number of nodes increases in system, and that the nodes in both front finger table and back finger table has a shorter distance with the current node. It is this reason that makes the inter-group routing has a shorter distance.

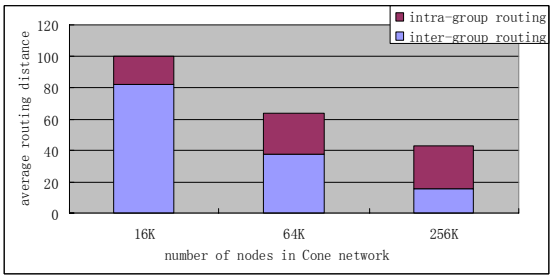


Figure 4: Decomposition of routing path in Cone.

### 4.4. Impact of argument *g* on routing distance

Overlay topology is decided by argument $g$, and there are two impacts on the Cone: (1) the maximum of online groups which decide the maximum of query hops; (2) for a Cone instance with a fixed number of nodes, argument $g$ decide its number of each group, for that it decide the proximity of finger table and query distance. Comparison of query distance in different number of nodes in condition of changing argument $g$ is illustrated in figure 5.

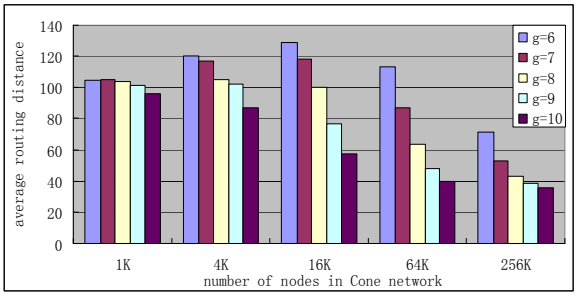


Figure 5: Impact on routing distance with argument $g$.

## 5. Conclusions

To deal effectively with the routing delay and query mismatch in P2P, this paper presents a topology-aware structured P2P system, named Cone, which applies a suit of mechanisms to extend Chord to optimize the utilization of both information- physical network topology and overlay network. In the Cone, nodes are divided into groups according to their identifiers; the related finger tables are set bidirectional and the links are pointed to the nearest nodes in the special group. The system facilitates the landmark+RTT method to

generate proximity information, and applies proximity neighbor selection to exploit proximity information. The Cone makes effectively use of network topology structure during network routing. The result of experiments show that the Cone, compared with Chord, has obvious improvements in the routing distance and the hops of overlay networks.

## References


[1] M. Castro, P. Druschel, Y. C. Hu, et al. Exploiting network proximity in distributed hash tables. In Proceedings of FuDiCo2002, Bertinoro, Italy, Jun. 2002.

[2] F. Dabek, M. F. Kasshoek, D. Karger, R. Morris, I. Stoica. Wide-area cooperative storage with CFS. In Proceedings of ACM SOSP, 2001.

[3] L. H. Dao, J. Kim. AChord: Topology-aware Chord in anycast-enabled networks. In Proceedings of the 2006 International Conference on Hybrid Information Technology (ICHIT 06), 2006.

[4] P. Ganesan, G. S. Manku. Optimal Routing in Chord. In Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms, 2004, pp. 169-178.

[5] F. Hong, M. L. Li, M.Y. Wu, J.D, Yu. PChord: Improvement on Chord to Achieve Better Routing Efficiency by Exploiting Proximity. In IEICE Transactions on Information and Systems, v E89-D, n 2, Feb. 2006, pp. 546-554.

[6] J. Hassan, S. Jha. Optimising Expanding Ring Search for Multi-Hop Wireless Networks. In Proceedings of IEEE GLBECOM, 2004.

[7] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker, A scalable content addressable network. In Proceedings of ACM SIGCOMM, 2001, pp. 329–350.

[8] A. Rowstron, P. Druschel. Pastry: scalable, decentralized object location and routing for large-scale peer-to-peer systems. In Proceedings of the 18th IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), 2001.

[9] A. Rowstron, P. Druschel. Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility. In Proceedings of ACM SOSP, 2001.

[10] S. Ratnasamy, S. Shenker, and I. Stoica. Routing Algorithms for DHTs: Some open question. In Proceedings of IPTPS'02, Mar. 2002.

[11] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker. Topologically-Aware Overlay Construction and Server Selection. In Proceedings of IEEE Infocom'2002, 2002.

[12] I. Stoica, R. Morris, D. Karger, et al. Chord: a scalable peer-to-peer lookup protocol for Internet applications. In Proceedings of the 2001 SIGCOMM conference, 2001, pp. 2-10.

[13] R. Winter, T. Zahn, J. Schiller. Topology-aware Overlay Construction in Dynamic Networks. In Proceedings of 3rd International Network, 2004.

[14] Z. Xu, C. Tang, and Z. Zhang. Building Topology-Aware Overlays using Global Soft-State. In Proceedings of ICDSC'2003, May 2003.

[15] J. Xiong, Y. Zhang, P. Hong, and J. Li. Chord6: IPv6 based topology-aware Chord. In Proceedings of the Joint International Conference on Autonomic and Autonomous Systems and International Conference on Networking and Services (ICAS/ICNS 2005), Aug. 2005.

[16] B. Zhao, J. Kubiatowicz, A. Joseph. Tapestry: an infrastructure for fault-tolerant wide-area location and routing. Computer Science Division University of California, UCB/ CSD20121141, 2001.

[17] E. Zegura, K. Calvert, and S. Bhattacharjee. How to model an internetwork. In INFOCOM96, 1996.