TCS-Chord: An Improved Routing Algorithm to Chord Based on the Topology-aware Clustering in Self-organizing Mode

Liu Ye, Yang Peng, Chu Zi, Wu Jiagao

the Key Laboratory of Computer Network and Information Integration, Ministry of Education, Southeast University, Nanjing, 210096, P.R. China
{yliu, pengyang, jerrychu, jgwu}@seu.edu.cn

Abstract. The study on semantic routing efficiency of DHT P2P networks is a key element to promote the development of P2P networks. It is critical for a P2P routing algorithm to be aware of the network physical topology, which leads to improvement of the routing efficiency. Structured P2P networks create a virtual topology on top of the physical topology. The only relation between the two layers exists in the Hashing algorithm, which makes a node's logical ID independent of its physical location. By analyzing the Hashing function's properties, some novel logical connections are presented among the destination node, the Chord semantic routing relay node sequence, and the ID of clustering neighbor nodes. This paper describes TCS-Chord (an improved routing algorithm to Chord based on the Topology-aware Clustering in Self-organizing mode) to improve the efficiency of Chord routing. Since the clustering nodes only have local views in the self-organizing mode, some rules are applied for a node to learn other nodes’ physical topology-aware locations. The TCS-Chord’s routing algorithm is described completely, and our experiments also indicate that TCS-Chord can improve the Chord semantic routing efficiently.

1 Introduction

The main purpose of the Knowledge Grid is to share and manage globally distributed knowledge resources in an efficient and effective way. P2P networks can be adopted as the semantic overlay layer of the Knowledge Grid. The scalability and autonomy make the P2P network a promising underlying substrate for a scalable Knowledge Grid[1]. The research on semantic routing efficiency of P2P network is a key element to promote the development of P2P network. Structured P2P networks create a virtual topology on the top of the physical topology. The only relation between the two layers exists in the Hashing algorithm, which makes a node's logical ID independent of its physical location. A message is taken from a routing hop to the successor node of a random location in the Internet, which results in high lookup delays and unnecessary wide-area network traffic[2]. Thus, it is critical for P2P routing to be aware of the network physical topology, which leads to improvement of the routing efficiency.

Nowadays, there are three main types of approaches to topology-aware semantic routing in structured P2P networks, such as proximity neighbor selection, topology-
based clustering and node$_d$ (node$_d$ is marked as a node’s identifier in structure P2P networks) assignment based on topology-aware.

Proximity neighbor selection is to choose routing table entries to refer to the topologically closest node among all nodes with node$_d$ in the desired portion of the node$_d$ value space. The idea is suitable for Pastry, which has several choice entries in each routing table, and leads to low delay stretch. But this idea does not work for overlay protocols like Chord$^3$, whose next hop desired node has a fixed value in the ID value space.

Topology-based clustering approaches include Brocade$^4$ and central controlling topology-based clustering$^5$. The Brocade effort adds a new overlay layer to the origin structured P2P networks. Supernodes are configured to manage nodes in AS (Autonomy System) and route messages among intra-AS to improve the performance of P2P routing. However, supernodes make self-organization and load balancing more difficult. A supernode is supposed to have strong computing capability. Additional protocols should be integrated into routing algorithms to process messages among supernodes and normal nodes. The topology-based clustering in centralized mode is easy to practice. With the rapid development of the scale, the more critical computing ability of central clustering server is required, which unavoidably causes the clustering server as a bottleneck in the system.

Node$_d$ assignment based on topology-aware attempts to insert the information of the physical location of a node into the value of its node$_d$. This method has already been implemented$^6$ in CAN with an achievement of a less delay stretch. However, the method destroys the uniform population of the node$_d$’s value space, which results from Hashing function property, and draws on load balancing problems in the overlay. Thus it does not work well in overlays with one-dimensional ID value spaces, such as Chord, Pastry$^7$, Viceroy$^8$ and etc. On the other hand, after a node$_d$ is created, the value of node$_d$ can not be changed during its live period. This manner can’t dynamically suit for the variety of physical topology.

Previous studies have paid little attention to maintaining self-organizing characteristics of P2P systems and applications. Thus, an improved routing algorithm to Chord based on the topology-aware clustering in self-organizing mode (TCS-Chord) is proposed in this paper. By analyzing the Hashing function, some novel logical connections are turned up among the destination node, the Chord semantic routing relay node sequence, and the identifiers of clustering neighboring nodes, which are the fundamentals to analyze TCS-Chord routing performance. TCS-Chord neither changes the uniform population of the node$_d$’s value space, nor configures some supernodes in P2P networks. An expected less stretch is attained in this way, while each node has equivalent properties.

The rest of this paper is organized as follows. Section 2 outlines a brief Chord routing algorithm. Section 3 presents the basic definitions, algorithms, and related analyses of TCS-Chord. Section 4 analyzes experiment results supporting our claims about TCS-Chord routing’s performance. Finally, we draw our conclusions and outline items for future work.
2 Chord Semantic Routing Algorithm

Chord forwards messages clockwise in the ID value space. Each node maintains a finger table. The $j$th entry in the finger table of the node with node $i$ refers to the live node with the smallest nodeid clockwise from node $i + 2^j$, $j \in \{1, 2, 3, ..., n\}$, $n$ equals 128 in Chord. The first entry points to node $i$’s successor, and the subsequent entries refer to nodes at repeatedly doubling distances from node $i$. The expected number of routing hops in Chord is $\frac{1}{2} \log_2 N$.

3 TCS-Chord Semantic Routing Algorithm

3.1 Basic Idea in TCS-Chord

In Chord routing scheme, a single semantic routing hop could decrease the distance to a query to 1 of the original logical distance with a $D$ cost of physical distance (such as latency), as well as a hop can forward a query at least half way along the remaining distance between the current relay node and the target identifier. TCS-Chord uses one selected neighbor to substitute the next relay node, which is chosen by using Chord scheme. TCS-Chord completes the same exponential approximation in a routing hop with a $d$ cost of physical distance. As the expected probability of substitution is not zero, an efficient routing process is achieved. $D$ is the average physical distance of all nodes in P2P networks, and $d$ is the average physical distance of all in a set of clustering neighbors on each node.

3.2 Topology-aware Clustering in Self-organizing Mode

To locate the nodes in P2P network, a set of coordinate axes should be built in terms of positions which may be mathematically specified. Each axis is a machine distributed in Internet. P2P nodes might ping these axis machines to get their own coordinates. Suppose the number of axes is $v$, the larger $v$ is, the more precise to describe the node position. Formula 1 is a judgement function to compare the relative distance between two nodes.

$$D_i(\text{node}_i, \text{node}_j) = \left( \sum_{i=1}^{v} \left| X_i - Y_i \right|^2 \right)^{\frac{1}{2}},$$

node$_i$’s coordinates is $<X_i>$, node$_j$’s coordinates is $<Y_j>$

In the self-organizing mode, nodes have no global views to cluster its neighbors in P2P networks. Each node has a different threshold parameter to cluster. Thus, $\forall$ node$_a$, if $\exists$ node$_b \in \Omega_j$, then node$_a$ may belong to $\Omega_b$ or not.
Algorithm 1. Clustering algorithm in self-organizing

$\theta_i \leftarrow \theta_{\text{default}} ; \Omega_i \leftarrow \{\text{node}_0\} ; \text{ClusterMax}_k \leftarrow \text{ClusterMax}_{\text{default}} ; /* \Omega_i$ is the neighbor set of node$_i$, ClusterMax$ _k$ is the number of maximum neighbors, $m_i$ is the current number of neighbors in the clustering set, $\theta_i$ is the current threshold to judge whether some node is its neighbor or not. */

(1) WHILE (node$_i$ is ACTIVE) DO
(2) According to the following 3 rules, get the physical locality coordinates of other nodes;
(3) if $D_i (\text{node}_j , \text{node}_k) \leq \theta_i$ 
  { 
    (4) $\Omega_i = \Omega_i + \{\text{node}_j\}$ ; 
    (5) According to Rule 3, publish the coordinates of itself and its neighbors to node$_j$ ;
    (6) if ($m_j > \text{ClusterMax}_k$) then 
      remove the node which is the farthest from node$_i$ ;
      $\theta_i = D_i (\text{node}_j , \text{node}_k)^{\max} , \text{node}_j \in \Omega_i$ ;
  }
(7) \forall \text{node}_i \in \Omega_i , if (\text{node}_i$ is DEAD) /* to check periodically whether the element is still alive. */
  { 
    $\Omega_i = \Omega_i - \{\text{node}_i\}$ ;
    $\theta_i = \theta_{\text{default}}$ ;
  }
(8) ENDDO

Rule 1(Flooding rule). In the self-organizing management mode, each node publishes the physical locality coordinates of its own to all other nodes by the means of flooding periodically.

Rule 2(In passing rule). A node embeds its locality coordinates into original messages it sends, so some nodes could extract the coordinates of the source node when relaying the data packets.

Rule 3(Reverse-influencing rule). If node$_a$ wants to add node$_b$ into its neighbor set, the possibility for nodes in $\Omega_a$ to be in the neighbor set of node$_b$ should be considerably high.

Discussions about the initial value of $\theta_{\text{default}}$ and ClusterMax$_{\text{default}}$ are given as followings.

Case 1. A new node which wants to join P2P networks knows nothing about the statistical information of the current topology, such as the number of nodes, and $D$ etc, $\theta_{\text{default}}$ is assigned with MaxInteger. The early stage in the self-organizing clustering algorithm is a slow convergence. An extreme case is that: $d > D$. As time goes by, every node gets more and more locality coordinates of other nodes, and
finally makes $d \ll D$. We can adopt the alive time of the node to solve this problem. Namely when a new node joins, it runs clustering module but still uses traditional Chord scheme. A fixed value of node’s alive time will trap the TCS-Chord routing module to run. We could also assign $\text{ClusterMax}_{\text{default}}$ with the expression $n^*(1 + \beta(R_c, R_b))$, where $n$ is the number of Chord routing table entries, $R_c$ is the computing ability of the node, $R_b$ represents the bandwidth, and $\beta(R_c, R_b) >= 0$ should also be satisfied here. Considering that there are many available resources on some nodes, the size of the set of clustering neighbors could be increased to contribute more to the expected global semantic routing efficiency.

Case 2. The P2P network has run for a period of time, on which a special application has been deployed. Because of periodical changes of the user behaviors, we can summarize statistical laws among $d$, $m$ and $D$. Suppose $g(m, N) = \frac{d}{D}$. By using the experiential formula to guide the new node to cluster in the self-organizing mode, we could improve the routing efficiency in good time.

The process for one node to acquire the locality coordinates of other nodes is shaped eclectically to fit both the performance and the function of P2P networks. According to rule 1, nodes could get enough information about others’ localities by consuming a large amount of bandwidth. By rule 2, nodes have a slow clustering convergence. Rule 3 is an efficient feedback process for nodes to get others’ coordinates. In algorithm 1, Line(3–6) is a process to optimize $d$. Line 4 fits for all kinds of node’s leaving situations. So algorithm 1 is fit for the dynamic changes within P2P networks.

3.3 TCS-Chord Semantic Routing Algorithm

In structured P2P networks, nodeid is the output of Hashing function. Given that the input is IP address, the distribution of nodeid’s value satisfies the balanced property of consistent Hashing function. Namely, with high probability in a N-node P2P networks, the expected identifier values of N nodes are uniformly distributed in the N continuous value subspaces. The validity and rationality of this assumption are proven in the papers that introduced consistent hashing [9][10]. And the related similar assumption and analysis can also be referred to theorem 4.1 in the paper of Chord [3]. Here we use SHA-1 to generate nodeid, where SHA-1 is a strongly collision-free Hashing function. The following theorems can be achieved with the statistic summings-up.

**Theorem 1.** In the statistic view, $m$ neighbors in node clustering set are uniformly distributed in $m$ continuous segments of value spaces.

TCS-Chord routing algorithm based on self-organizing clustering is given as algorithm 2. The meanings of presented parameters are described as followings: node$_{id}$ is the current relay node(or source node); $\Omega_i = \{\text{node}_i, i \in [1,m_i]\}$, where $\Omega_i$ is the neighbor set of node$_{id}$; the relay node set for the next hop in Chord routing table is $\{\text{nodeChordRoute}_i, \text{nodeChordRoute} = \text{finger} (\text{node}_k+2^j), j \in [1,n]\}$, node$_{destiny}$ is the target
node.

**Definition 1** (the approximation subspace $\alpha$).

$$
\alpha = \begin{cases}
\frac{(node_{\text{Destiny}} - node_k)}{2}, & \text{if } (node_{\text{Destiny}} > node_k); \\
\frac{(2^k + node_{\text{Destiny}} - node_k)}{2}, & \text{if } (node_{\text{Destiny}} < 2^k - node_k) \text{ and } (node_{\text{Destiny}} < node_k); \\
\frac{(2^k + node_{\text{Destiny}} - node_k)}{2^{2^k}} \cdot Y[0, node_{\text{Destiny}}], & \text{if } (node_{\text{Destiny}} > 2^k - node_k);
\end{cases}
$$

**Algorithm 2. TCS-Chord routing algorithm**

1. if $node_{\text{Destiny}} \in \Omega_k$, then {select $node_{\text{Destiny}}$ as the next hop node; break; }
2. if $node_{\text{Destiny}} \notin \Omega_k \&\& node_{\text{Destiny}} \in \{node_{\text{ChordRoute}}\}$, then {select $node_{\text{Destiny}}$ as the next hop node; break; }
3. if $\Omega_k - \{node_k\} \neq \emptyset$, then $check(\alpha)$; //check whether there is any nodeid in $\alpha$
4. if $check(\alpha).\text{number} > 0$
   then {select nodeid from $check(\alpha).\text{elements}$, where $\|node_{\text{Destiny}} - node_id\|_\infty$; }
   else {use Chord routing algorithm to select the next hop node;}

Let $N \in [0,2^n]$, $E(Entry) = n$, $E(Hops)(\text{Chord}) = \frac{1}{2} \log_2 N$, where $N$ is the number of global nodes in P2P networks, $n$ is the Entry number of Chord routing table, $n$ is fixed because of the fixed length of the output value of the Hashing function and is independent of $N$; $E(Hops)$ is expected routing hops.

Let $E(m_k) = m_k$, $m_k \in [1,N]$, where $m_k$ is the size of the set of clustering neighbor nodes of node $k$, $m$ is the average size of all sets of clustering neighbor nodes.

If $m \in (1,n]$, then $E(Hops)(m) = \frac{1}{2} \log_2 n^N$, where the clustering neighbor nodes do not affect the expected logical hops number.

If $m = N$, then $E(Hops)(m) = 1$, and in this scenario it equals the case that each node’s routing table contains a $N$ Entries.

Fig 1. approximately shows the function of $E(Hops)(m)$.

![Figure 1. E(Hops)(m) ~ m function diagram](image-url)
If \( m \in [n, N] \), \( E(Hops)(m) \) is a monotony decreasing function in range \( E(Hops)(m) \in [\frac{1}{2} \log_2 N, 1] \), let
\[
g(m) = \frac{1}{2} \log_2 N - \frac{1}{2} \log_2 \frac{N}{n} \cdot (m - n), \quad N \in [0, 2^n] \;
\]
else if \( N \to \infty \), then
\[
\lim_{N \to \infty} k = \lim_{N \to \infty} \frac{1}{N} \log_2 N = 0,
\]
where \( k \) is \( g(m) \)'s slope, then \( \forall m \in [n, N] \),
\[
\lim_{N \to \infty} E(Hops)(m) \equiv g(m) = \frac{1}{2} \log_2 N - \frac{1}{2} \log_2 \frac{N}{n} \cdot (m - n).
\]

Taken all the above steps into consideration, the following theorem can be deduced.

**Theorem 2.** Supposed that \( N \) is large enough. If \( m \in [n, N] \), the value of \( E(Hops) \) will be influenced so that we can use the values of function
\[
g(m) = \frac{1}{2} \log_2 N - \frac{1}{2} \log_2 \frac{N}{n} \cdot (m - n)
\]
to fit curve \( E(Hops)(m) \).

The logical connections among the destination node, the Chord semantic routing relay node sequence, and the ID of clustering neighboring nodes are illustrated as follows.

**Remark.** Let \( \rho = (m_i - 1) \cdot 2^{-i} \), according to theorem 1, in \( \Omega \), there are \( \rho \) nodes uniformly falling in the next hop range \( [0, 2^i] \). The node’s ID of the next hop relay node is \( node_i = node_i \cdot \text{finger}(node_i + 2^i) \), satisfying \( node_i \in [0, 2^i] \).

We define the function \( E(Chord) \) and \( E(TCS-Chord) \) to specify the expected physical routing path distance between two arbitrary nodes in the P2P network by using Chord(C) and TCS-Chord(S) scheme respectively.

\[
\begin{align*}
E(Hops)(m) &= \begin{cases} 
\frac{1}{2} \log_2 N & m \in [1, n] \\
\frac{1}{2} \log_2 N - \frac{1}{2} \log_2 \frac{N}{n} \cdot (m - n) & m \in [n, N]
\end{cases} \\
E(MaxH)(m) &= 2 \cdot E(Hops)(m) \\
E(C_i) &= E(D_i) = \begin{cases} 
0 & ;i = 0 \\
\frac{1}{D} & ;0 < i < \log_2 N
\end{cases} \\
E(S_i) &= \begin{cases} 
E(d_i) & ;\rho \geq 1 \\
E(\rho \cdot d_i + (1 - \rho) \cdot D) & ;0 \leq \rho < 1
\end{cases} = \begin{cases} 
0 & ;i = 0 \\
\frac{1}{D} & ;0 < i < \log_2 N^{-1} \\
\rho \cdot d_i + (1 - \rho) \cdot D & ;\log_2 N^{-1} < i \leq E(MaxH)(m)
\end{cases} \\
E(m_i) &= m
\end{align*}
\]
\[
E(\text{Chord}) = E\left( \sum_{i=0}^{\log_2^{N}} C_i \right) = E\left( \sum_{i=0}^{\log_2^{N}} E(D_i) \right) = \left( \sum_{i=0}^{\log_2^{N}} t \cdot \frac{1}{\log_2^{N}} dt \right) \cdot E(D) = \frac{1}{2} \cdot \log_2^{N} \cdot D
\] (2)

If \( m \) satisfies \( \log_2^{m-1} < \log_2^{N} - \frac{\log_2^{N} - 2}{N-n} \cdot (m-n) \), then

\[
E(\text{TCS - Chord}) = E\left( \sum_{i=0}^{E(MacFlip(m))} S_i \right) = E\left( \sum_{i=0}^{E(MacFlip(m))} E(d_i) \right) + E\left( \sum_{i=\log_2^{m-1}}^{E(MacFlip(m))} E(\rho \cdot d_i + (1-\rho) \cdot D_i) \right)
\]

\[
= \frac{1}{2} \cdot \left( \log_2^{m-1} \right)^2 \cdot \frac{d}{2 \cdot E(Hops)(m)} + \frac{(m-1)}{2 \cdot E(Hops)(m)} \cdot \frac{1}{\ln(2)} \cdot \left\{ \left( \frac{1}{2} \cdot (t + \frac{1}{\ln(2)}) \right) \right\}
\]

\[
\log_2^{m-1} \cdot (d-D) + \frac{1}{2} \cdot \left( \frac{1}{2 \cdot E(Hops)(m)} \right)^2 \cdot \left( \log_2^{m-1} \right)^2 \cdot D
\] (3)

If \( m \) satisfies \( \log_2^{m-1} > \log_2^{N} - \frac{\log_2^{N} - 2}{N-n} \cdot (m-n) \), then

\[
E(\text{TCS - Chord}) = E\left( \sum_{i=0}^{E(MacFlip(m))} S_i \right) = E\left( \sum_{i=0}^{E(MacFlip(m))} E(d_i) \right) - E\left( \sum_{i=\log_2^{m-1}}^{E(MacFlip(m))} E(S_i) \right)
\]

\[
= E\left( \sum_{i=0}^{E(MacFlip(m))} E(d_i) \right) - E\left( \sum_{i=\log_2^{m-1}}^{E(MacFlip(m))} E(d_i) \right) = E(Hops)(m) \cdot d
\] (4)

The diagram of curve associated with formula 2, formula 3 and formula 4 is shown in Fig. 4, which also contains experimental data results. When \( i \) is small, as in the early stage of semantic exponential routing close to the target node, the probability of using neighbor nodes to relay messages is high. When \( i \rightarrow \log_2^{N} \), the probability becomes lower, and here the chosen relay node is the same as that by using Chord scheme. When \( \rho > 1 \), there are \( \rho \) nodes randomly distributed in the next hop routing range. The method to select the neighbor with a closer identifier to the target node from these \( \rho \) nodes has a constant close approach, although this contribution to P2P routing scheme is too small to take into account.

4 Experimental Results And Discussion

To test the efficiency and effectiveness of the TCS-CHORD routing algorithm, we run sets of experiments on hierarchical Transit-Stub topologies which are generated by GT-ITM\[11\] software toolkit. TS topologies model networks using a 2-level hierarchy of routing domains with transit domains that interconnect lower level stub domains. In these topologies (shown in Table. 1), the average physical distance (for instance, latencies) of intra-transit links is \( 2^{N-1} \) times longer than that of intra-stub.
Table 1. Sets of networks modeled by Transit_Stub topologies

<table>
<thead>
<tr>
<th>N</th>
<th>Wan_Dmax</th>
<th>Wan_Davg</th>
<th>Lan_dmax</th>
<th>Lan_davg</th>
<th>k</th>
</tr>
</thead>
<tbody>
<tr>
<td>90</td>
<td>2980</td>
<td>1242.31</td>
<td>46</td>
<td>22.20</td>
<td>5</td>
</tr>
<tr>
<td>156</td>
<td>6732</td>
<td>2537.85</td>
<td>48</td>
<td>23.86</td>
<td>6</td>
</tr>
<tr>
<td>240</td>
<td>2871</td>
<td>1389.03</td>
<td>62</td>
<td>25.55</td>
<td>5</td>
</tr>
<tr>
<td>342</td>
<td>3547</td>
<td>1833.83</td>
<td>63</td>
<td>26.19</td>
<td>5</td>
</tr>
<tr>
<td>462</td>
<td>3596</td>
<td>1767.25</td>
<td>82</td>
<td>27.81</td>
<td>5</td>
</tr>
<tr>
<td>600</td>
<td>3759</td>
<td>1897.74</td>
<td>96</td>
<td>27.14</td>
<td>5</td>
</tr>
<tr>
<td>756</td>
<td>4268</td>
<td>2112.98</td>
<td>160</td>
<td>28.91</td>
<td>5</td>
</tr>
<tr>
<td>930</td>
<td>2487</td>
<td>1164.24</td>
<td>111</td>
<td>28.88</td>
<td>4</td>
</tr>
<tr>
<td>1122</td>
<td>4107</td>
<td>2120.90</td>
<td>172</td>
<td>30.48</td>
<td>5</td>
</tr>
<tr>
<td>1332</td>
<td>2407</td>
<td>1181.39</td>
<td>188</td>
<td>31.06</td>
<td>4</td>
</tr>
<tr>
<td>1560</td>
<td>4534</td>
<td>2072.00</td>
<td>185</td>
<td>31.49</td>
<td>5</td>
</tr>
<tr>
<td>1806</td>
<td>4369</td>
<td>2298.79</td>
<td>173</td>
<td>32.30</td>
<td>5</td>
</tr>
<tr>
<td>2070</td>
<td>4830</td>
<td>2545.63</td>
<td>201</td>
<td>33.01</td>
<td>5</td>
</tr>
<tr>
<td>2352</td>
<td>2807</td>
<td>1324.51</td>
<td>223</td>
<td>33.05</td>
<td>4</td>
</tr>
<tr>
<td>2652</td>
<td>3020</td>
<td>1528.52</td>
<td>187</td>
<td>33.57</td>
<td>4</td>
</tr>
<tr>
<td>2970</td>
<td>3497</td>
<td>1681.81</td>
<td>251</td>
<td>34.55</td>
<td>4</td>
</tr>
<tr>
<td>3306</td>
<td>4986</td>
<td>2554.99</td>
<td>270</td>
<td>35.45</td>
<td>5</td>
</tr>
<tr>
<td>3660</td>
<td>6274</td>
<td>2832.37</td>
<td>288</td>
<td>36.05</td>
<td>5</td>
</tr>
<tr>
<td>4020</td>
<td>3536</td>
<td>1575.91</td>
<td>286</td>
<td>35.88</td>
<td>4</td>
</tr>
</tbody>
</table>

| N : number of nodes in sets of networks; |
| Wan_Dmax : the maximum value of physical instance in the total network; |
| Wan_Davg : the average value of physical instance in the network; |
| Lan_dmax : the maximum physical instance of intra-stub in the network; |
| Lan_davg : the average physical instance of intra-stub in the network; |
| k = log2(Wan_Davg/Lan_davg); |

We choose RD (Relative Distance) as the parameter to evaluate the experimental result, because RD evaluates the locality properties of structured P2P routes. When a lookup message travels from a source node to its target node randomly, RD is treated as the ratio of Dist_p2p to Dist_ip, where Dist_p2p is the sum of the hops in IP layer using P2P routing algorithm, and Dist_ip is the sum of the hops using simple fictitious routing scheme in which the source knows all other target nodes’ ip addresses. The P2P routing performance can become more efficient with a lower RD.
Figure 2 shows the average of lookup RD in Chord and TCS-Chord respectively, where TCS-Chord software runs on per node with the threshold parameter for $\theta_i = 100 = a_0 \times \text{Lan}_{d_{avg}}$. With the increase of the number of nodes in P2P networks, the average diameter of a Lan is also on the increase. Thus, with the fixed $\theta_i$, the curve of TCS-Chord is increasing slowly on the whole. Under the circumstances of the worst routing performance in Chord, we could achieve a remarkable improvement by running TCS-Chord routing algorithm on all the nodes or some of them.

Figure 3. $g(m, N) \sim m$, $N=2970$ nodes

Figure 4. Effect of average number of clustering elements per node in TCS-Chord, $N=2970$ nodes
In Figure 3, we fix the total number of nodes at 2970 in Table 1 and draw the graph of \( g(m, N) \), where \( g(m, N) = \frac{d}{D} \) is a function of the network size and the number of clustering neighbors in a P2P network based on the TS topology. In Figure 4, we use the network topology in figure 3 and describe the effect of decreasing RD determined by both experiments data and theoretic formulae. Results tallies with the theory perfectly. \( m \in [16, 24] \) is the best size of neighbors for a peer to run with the clustering algorithm. Look at figure 3, which we generate the network topology with the parameter of 18 nodes in a stub, and we find that TCS-Chord could properly suit the topology of P2P networks. When \( m \) equals 1, TCS-Chord is transformed back to Chord.

Experiments verify that TCS-Chord can improve the efficiency of semantic routing algorithm to some extent. Furthermore, while a special network topology is generated, any link latency of two arbitrary nodes is fixed. Thus, experiments cannot reflect the dynamic changes of all latencies of links in real P2P applications. TCS-Chord clustering algorithm could be fit for dynamic changes of P2P networks, but experiments based on the fixed topology cannot show this characteristic.

5 Conclusion

By analyzing the Hashing function, some novel logical connections are presented among the destination node, the Chord semantic routing relay node sequence, and the ID of clustering neighboring nodes in this paper. We apply these connections to the design of the routing algorithm of TCS-Chord model. Compared with other related researches, we have made two contributions: (1) TCS-Chord maintains equivalent properties without introducing supernodes, (2) and sustains the uniform population of the node’s value space without inserting the topology-aware information into the node’s value, which otherwise will make some nodes overload in special circumstances. Our theoretical analysis, along with the results of experiment running on hierarchical TS topologies illustrates an expected less stretch has achieved in TCS-Chord. While certain decisions in TCS-Chord are Chord specific, we believe that similar design decisions can be made for other structured P2P networks, such as Pastry, SkipNet etc. Good performance of these advanced routing algorithms could be predictable as well. We believe TCS-Chord is an interesting enhancement to leverage P2P routing performance.

In the future work, larger-scale simulations for P2P topology-aware routing algorithms in the live Internet environment are necessary to confirm the validity of TCS-Chord and refine the algorithms. In the next term, we are going to utilize the Planet-Lab circumstance to provide a large-scale testbed for this purpose.
Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (No. 60573133), the National High-Tech Research and Development Plan of China under Grant No. 2003AA413320, and the National Grand Fundamental Research 973 Program of China under Grant No. 2003CB314801.

References