Journal of Theoretical and Applied Information Technology
15th November 2012. Vol. 45 No.1

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

# A RESTRICTED-SCOPE LOOKUP SERVICE IN PEER-TO-PEER NETWORK

**JIANCHUN LI, JIANYONG LI, DAOYING HUANG, LIHUA SHI**
School of Computer & Communication Engineering, Zhengzhou University of Light Industry,
Zhengzhou 450002, Henan, China

**ABSTRACT**

Consistent hashing-based P2P networks, which scalability is indicated by efficiency of query and maintenance cost, take the hashing algorithm as mapping relation between the data and the identifier of every node in the network. By analyzing some main characters in structured P2P network, we introduce the Restricted Scope Lookup Service(RSLS） to divide the network into some less size structured sub-systems, so as to limit the bound of routing message while keep the scalability of network. The logical relationship among nodes in sub-system is organized by the binary tree structure, in which some definitions and qualities are proved, and the self-organizing algorithm and the routing algorithm in sub-system are described too. Simulations have verified the cost of routing algorithm and stability scaling logarithmically with the number of nodes in sub-system.

**Keywords:** *P2P, Lookup Service, Structured Sub-Systems, Binary Tree*

## 1. INTRODUCTION

An important feature of Peer-to-Peer (P2P) system is that resources and services scatter to all nodes in the system other than the traditional mode which take them in the central, so that the operation to find key words become a core operation in the system. The unstructured P2P systems were confirmed out of place for the reasons of incomplete query results, slow lookup speed and poor scalable, therefore the different structures and special lookup services were developed to resolve it.

The current researches concentrate emphasis on the structured P2P system[1, 2, 3, 4], in particular P2P structured networks based on consistent hashing, were found many excellent characteristics. In these systems, operations provided by lookup service include two parts as follow: (1) an m-bit identifier is created for each key which is assigned to node whose identifier corresponding to the key (always by intercepting prefix or postfix of identifier ), and the node take charge of the indexes of the key;(2) operation of lookup(key) can return indexes from destination node, and this process include three steps: calculating the m-bit identifier of the key using a base hash function and corresponding node at first , routing the request message to the destination node by the system route algorithm, and responding the request by destination at last.

Using consisting hash, every node is assigned with a unified identifier which determines the node's logical location in the network. The neighbor relations among nodes determined by identifiers make the system as a structured topology, for instances of the ring Chord, Pastry and the butterfly Viceroy. Query is limited to the all active nodes in the network, and efficiency of query is related to the sum of nodes.

Scalability of structured P2P system is determined by efficiency of query (indicating by net diameter) and maintenance cost (indicating by the size of routing table). Research[5] has revealed the tradeoff relationship between them, and figure.1 shows the tradeoff curve for some systems. For example, N active nodes in Chord construct a ring ordered by the node's identifiers, and every node keeps a routing table(referred to as a "finger table" in Chord) which contain $O(\log N)$ nodes apart $2^i$ in clockwise, while the routing cost is $O(\log N)$ in Chord. Therefore, tradeoff relationship means to improve performance at one aspect must depress another.

This paper introduces Restricted Scope Lookup Service (RSLS), which is based on consistent hashing and can provide scalable and efficient lookup service. Dividing network into some structured sub-systems is precondition for RSLS, but the principle that how to divide is responsibility for application software above RSLS. For example, nodes which share same file can join a sub-system in P2P files share network, and users who choose the same video program can organize a same sub-system in a P2P stream media order and broadcast system. The number of nodes in sub-system must be fewer than that in whole network, so that performance can be improved across-the-board in sub-system. RSLS provides functions including how to organize the structured sub-system and maintains stability of sub-system as node joining or leaving, and the routing algorithm which transfers queries from source node to destination node is offered too. In an N-node sub-system, each node maintains information only about $O(\log N)$ other nodes, and the query which can be routed to destination only requires $O(\log N)$ messages. Compare to other lookup services[6, 7, 8 ,9, 10], RSLS has good qualities at following aspects: (1) nodes involved in the operation are limited in a certain sub-system and the number of them must be lower than that of whole network; (2) identifier of each node is created instantly, so that logical relationship between nodes needn't be established before nodes join the network, and assure anonymity of nodes, (3) the logical structure of sub-system is plain and some features can be proved in it.

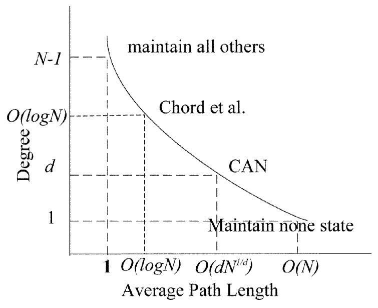


*Figure 1: Asymptotic Path Length*

The rest of the paper is organized as follows. Section 2 presents some definitions, description of central algorithm and performance analysis about RSLS. Section 3 presents simulations about RSLR and analysis for results. Section 4 concludes the paper.

## 2. THE RSLS PROTOCOL

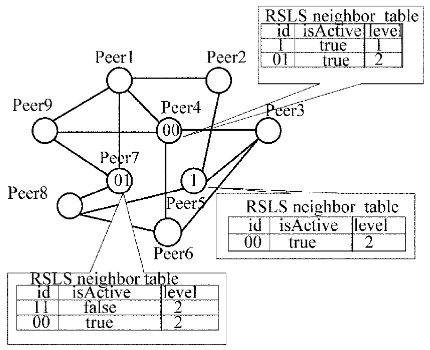
The RSLS protocol employs binary tree structure to represent logical structure of sub-system, and specifies the operations which regulate node to join sub-system and maintain sub-system stable. In figure.2, a sub-system includes three peers among the system.

### 2.1 Data Structure

The logical relationship among nodes form a binary tree structure, the relationship and growth process show in figure.3, and every leaf in the tree correspond to a node's current logical state in sub-system. The process for a new node joining sub-system can be described as selecting and connecting a active node in sub-system comply with the RSLS protocol , and the new node can be named as son node and the node accept it named as father node. Differ from other P2P system, identifier of node is temporary and unfixed, $id_x^h$ represents identifier of node $X$ when it's level is $h$ ,original node (when the first node join sub-system) is identified as $0$ , and other node at $h^{th}$ level can be identified with h-bit identifier. The action of joining or leaving can be described by the action of leaves in binary tree.

**Definition 1 (Saturation):** To suppose value of a binary tree's depth is $l$ and the number of its leaves is $N$ , we can present $S_l = N/2^l$ as this binary tree's saturation, and we call it as absolute saturated tree when this value is 1. Depth has quality of $l = O(\log N)$ because that RSLS control the actions when new node joins in sub-system.

**Definition 2 (Necessary interval):** For any two nodes $X$ and $Y$ at $h^{th}$ level, we describe the sum of different bits for same position in their h-bit identifiers as necessary interval between them, recorded as $D(X,Y)_h$ . The value can't surpass and the expectation may be $h/2$ .



RSLS neighbor table

| id | isActive | level |
|---|---|---|
| 1 | true | 1 |
| 01 | true | 2 |

RSLS neighbor table

| id | isActive | level |
|---|---|---|
| 00 | true | 2 |

RSLS neighbor table

| id | isActive | level |
|---|---|---|
| 11 | false | 2 |
| 00 | true | 2 |

*Figure.2: RSLS neighbor information table*

**Definition 3 (Neighbors):** Neighbors are nodes which have a logical relationship directly. Father node accepts son node as its' new neighbor, while son node finds it neighbors from information it inherits from father. Node lies at $h^{th}$ level have

$h-1$ neighbors and value of necessary interval between them is 1.

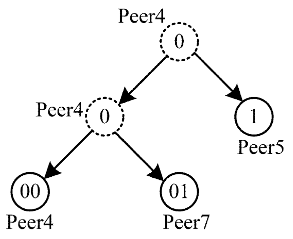


*Figure.3: Binary tree of RSLS growth*

RSLS use neighbor table (Figure.2) conserve all neighbors' information, which includes attributes about neighbors. $id^{h}_{node}$ represents the neighbor's identifier, *isActive* represents whether the neighbor had joined sub-system or not; and *level* represents neighbor's location in binary tree at current. When new node joins sub-system, father node adds it into local neighbor table while son node initializes its' own neighbor table. Node needs inform its' neighbors to refresh their own neighbor tables when some attributes had been altered.

## 2.2 Node Joins

The state of sub-system can be influenced as it affiliates a new node; hence affect the distribution of indexes. In order to assure the time complicacy of routing arithmetic as $O(\log N)$, RSLS defines three central operations: (1) how to initialize and maintain neighbor table; (2) how to relocate indexes of key among nodes; (3) how to route query in sub-system. We present first and second operation at follow.

**Policy that neighbor is prior as father node:** Any candidate node $X$ has ability to affiliate a new node $Y$ only has both characters of (1) all of its neighbors had joined already; (2) none of neighbors' current level is less than that of its' own. Otherwise, when node $X$ has a neighbor whose *isActive* attribute is false, node $Z$ should be selected to affiliate new node , and must satisfy the condition that $Y$ will be the referred neighbor of $X$ if $Z$ accept it into sub-system, on the other hand, to select neighbor $X_i$ which current level less than $X$ as candidate node once again.

**Maintain and initialize neighbor table:** some attributes of father node $X$ should be updated according to new logical position after it accepted son node $Y$. For example, its' identifier should be $id^{h+1}_{x} = 2*id^{h}_{x}$ when it come to $(h+1)^{th}$ level. Neighbor table should be refreshed at the same time, it should append node $Y$ as new neighbor and inform former neighbors to refresh their own neighbor tables. Node $Y$ initialize its' own attributes with information receiving from $X$. Its' identifier is $id^{h+1}_{x} = 2*id^{h}_{x} + 1$, and each neighbor of $Y$ is either son of father's corresponding neighbor or null. $Y$ should contact every active neighbor to get their information and inform them about the information of themselves.

**Index transfer:** After new node joined, it needs not only to publish its' own keys, but also to manage former keys in sub-system. RSLS adopt the same prefix matching policy to assign keys to nodes, and it means that the index of key always be managed by the node whose current identifier is section of prefix to m-bit identifier of key. For example, node $X(1101)$ lies at $4^{th}$ level of binary tree and preserves all indexes of key whose prefix of m-bit identifier is $1101$. The index can be divided into two parts whose prefix includes $11010$ and $11011$, and the latter will be transferred to son node $Y$ after it joined in sub-system by connecting $X$. Meanwhile, identifier of $X$ becomes $11010$ and that of $Y$ becomes $11011$.

## 2.3 Failure

When a node $X$ failure, some influence will make sub-system unstable transitorily, for the reasons as (1) the indexes of key $X$ maintaining should become untouchable; (2) routing path including $X$ is disable. This paper discusses the situation of node join sub-system one by one, and the case of node failure will be discussed in latter papers.

## 2.4 Routing Algorithm

Query for key is routed among nodes, and to choose next-hop node depend on the routing algorithm and information of neighbor tables. We can get some characteristics of routing algorithm by analyzing structure of binary tree (logical structure of sub-system).

Proposition: In an absolute saturated tree whose value of depth is $l$, query can reach destination node $Y$ from source node $X$ by the number of $D(X,Y)_h (0 < h \le l)$ routing messages.

Proof: As suppose $X$, $Y$ are any two nodes and $D(X,Y)_h = k$ at $h$ level, a neighbor node $X_1$ can be found which satisfy $D(X_1,Y)_h = k-1$ at neighbor table of $X_1$, in the same way, $X_2$ can be found at neighbor table of $X_1$ and

$D(X_2,Y)_h = k-2,...,Y$ must be found at neighbor table of $X_{k-1}$ finally. We name nodes sequence of $(X,X_1,X_2,...,X_{k-1},Y)$ as routing path from $X$ to $Y$, or least routing path if its' length is $D(X,Y)_h$ (including number of $D(X,Y)_h+1$ nodes).

Example of finding least routing path: As suppose $X$, $Y$ are any two nodes and $id_x^h=(k_1,k_2,...,k_j,...,k_h)$, $id_y^h=(k_1,k_2,...,\overline{k}_j,...,k_h)$ are their current identifiers, between which $k_j$ is highest different bit. We can find neighbor node $X_1$ whose identifier is $id_{x_1}^h=(k_1,k_2,...,\overline{k}_j,...,k_h)$ and $D(X_1,Y)=k-1$. It can obtain node $Y$ by $k$ operations as such. In fact, next-hop node of $X_i$ is its' neighbor $X_{i+1}$ whose identifier has a highest different bit with that of $X_i$.

RSLS uses consisting hash to get exclusive m-bit identifier for each key, and assigns key to node following the same prefix matching policy. The core operation at RSLS is routing query to destination node by uniform routing algorithm.

The routing algorithm can be described as process of decreasing necessary interval between routing node and destination node until reaching destination node. Suppose in an absolute saturated tree whose value of depth is $l$ and have $N=2^l$ leaves, so that the expecting routing interval between any two nodes is $(\log_2 N)/2$ which similar to that of Chord, but here $N$ is the number of nodes in a sub-system.

We analyzed a special binary tree above, which seldom occurred in practice; therefore, the operation might be failure at ordinary binary tree when it select a inactive neighbor as next-hop or the destination node is inexistent in sub-system. The rate of failure relies mainly on the saturation of binary tree obviously.

RSLS adopt father nodes to substitute failed node and inexistent destination node: At $h$ level, node $X_i$ will route the query to destination node $Y$, it route the query to its' father node $X_i^{'}$ when next-hop it calculate by routing algorithm is inexistent, and destination node be changed to node $Y^{'}$ which is father of node $Y$. Finally, node $Y^{'}$ route the message to $Y$ if $Y$ had joined sub-system already.

The improved routing algorithm increases the cost of routing process but ensures the reliability of it. Meanwhile, the algorithm has character of exact lookup. Whether or not destination node has joined sub-system is unknown for source node when it send query, but the improved algorithm allow the predicted destination node at first not to be node which conserve proper indexes. The expecting routing interval between any two nodes is still $O(\log N)$, and showed by the data of simulations is value of approximate $(\log_2 N)/2$.

## 3. SIMULATIONS AND CONCLUSION

We had simulated sizes of sub-system whose logical structures were presented as binary trees with different depth. Simulations test the data of saturation, maintenance cost and efficiency of routing algorithm in the binary trees. Saturation of tree indicates the capability of sub-system to accept user nodes; maintenance cost means the cost to keep sub-system stable again after a new node joined, and we employ the number of messages to find father node and refresh neighbor tables as maintenance cost in RSLS; The process of lookup (key) operation had been simulated, and cost of the operation and maintenance cost were verified scaling logarithmically with the number of nodes in sub-system.

We simulated the process of building sub-system at most 30,000 nodes and operations in it, our work based on premises as follow:

1). Every node has same opportunity to connect with a new node;

2). Output of key by consisting hash is symmetrical within extent of value region, and query for key is random and sum of them is approximate for every key in sub-system.

### 3.1 Data About Saturation

Saturation of binary is determined by the number of leaf and the value of its' depth, we use $S_l^{\min}$ represents the value of saturation when depth of binary tree reach $l$ just now, and use $S_l^{\max}$ represents of the value of saturation if any node accept a new node and reach higher depth $l+1$. The data of saturations at different size of binary trees show in figure.4. Data of $S_l^{\min}$ and $S_l^{\max}$ tend to stable while depth of tree become higher, so that it have good capability to accommodate nodes and quality of $l=O(\log N)$.

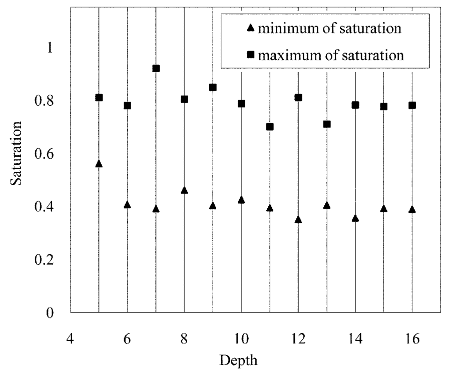


*Figure.4: The Saturation Of Different Binary Tree Depth*

### 3.2 Maintenance Cost

Neighbor tables of some nodes should be refreshed after one node had joined sub-system. We employ the number of messages to select father node and refresh neighbor tables as maintenance cost, and it involves father node, son node and their neighbor nodes. The cost relies on the position of father node in binary tree. Suppose candidate node lies at $h^{th}$ level and the real father lies at $m^{th}$ level, so the value of maintenance cost is $(h-m)+2m = h+m = O(\log N)$ . It need only additional sum of $O(\log N)$ messages to make sub-system stable again when a node join in. Figure.5 shows the data of maintenance cost in different depth of binary trees.

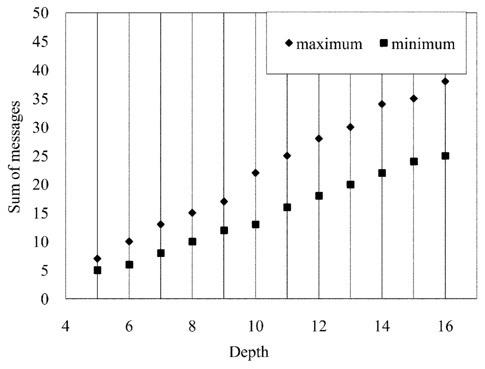


*Figure.5: The Cost For Nodes Joining At Different Depth Binary Tree*

### 3.3 Expecting Routing Interval

The efficiency of routing algorithm can be indicated by the number of routing operations in the process to route query from source node to destination node. Expecting routing interval is affected by the necessary interval among nodes and state of sub-system (determined by saturation mainly).

Accepting new node has influence to expecting routing interval on two aspects as follow: the value of expecting routing interval may be increased when the necessary interval between new node and others is more than it, and we call this influence as positive influence; on the other hand, new node may increase the saturation of tree which reduce the rate of routing failure and decrease value of expecting routing interval, and we call this influence as negative influence.

In the period of saturation increasing from $S^{\min}$ to $S^{\max}$, the expecting routing interval behave two different phases. At the beginning, positive influence will take more effect to expecting routing interval than negative influence if new node lies at $l^{th}$ level, for the reason that binary tree have more ability to accept nodes at $l^{th}$ level, expecting routing interval is increased as a whole. When expecting routing interval arrive at some extent and positive influence by new nodes is lessened but the negative influence is increased, so that expecting routing interval presents trend of decrease until the value of depth become $l+1$. We simulate 500 random operations of lookup for every node in different sizes of sub-system, and the influence of two directions was proved by results of simulation and obvious in large size sub-systems. Figure.6 shows that the value of expecting routing interval is bigger than $(\log_2 N)/2$ appreciably.

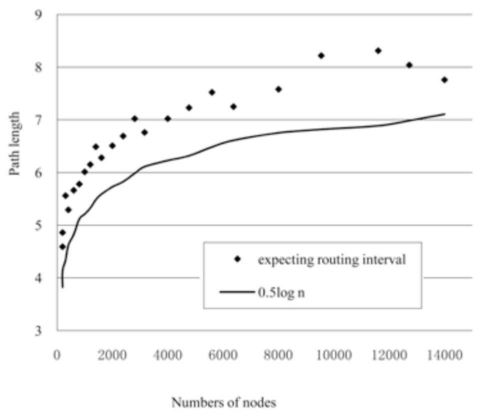


*Figure.6. The Path Length Of Different Sub-System Size*

## 4. CONCLUSION

We introduce RSLS protocol which provides lookup service in P2P environment. RSLS construct sub-systems in a P2P network, and improve the quality of lookup service through decreasing the sum of participant nodes. Sub-system has self-organization ability which organizes logical relationship among nodes by structure of binary tree. In a sub-system including $N$ nodes, every node conserve states of other $O(\log N)$ nodes, and route query to destination node by $O(\log N)$ intervals, so that sub-system constructed by RSLS has quality of scalability.

## ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China under Grant 60974005 and 60704004, the Specialized Research Fund for the Doctoral Program of Higher Education under Grant 20094101120008, the Nature Science Foundation of Henan Province under Grant 092300410201 and Zhengzhou Science and Technology Research Program under Grant 0910SGYN12301-6.

## REFRENCES:

[1] Jie Xu and Hai Jin, “A structured P2P network based on the small world phenomenon”, The Journal of Supercomputing, Vol. 48, No.3, 2009, pp. 264-285.

[2] Daniel R. Karrels, Gilbert L. Peterson and Barry E. Mullins, “Structured P2P technologies for distributed command and control”, Peer-to-Peer Networking and Applications, Vol. 2, No.4, 2009, pp. 311-333.

[3] Xiaobo Zhou, Kaiping Xue, Jian Zhou, Hancheng Lu and Peilin Hong, “HICUS: A quasi-structured P2P system based on hierarchical interest”, Journal of Electronics, Vol. 26, No. 2, 2009, pp. 198-204.

[4] Nadir Shah, Depei Qian and Rui Wang, “MANET adaptive structured P2P overlay”, Peer-to-Peer Networking and Applications, Vol.5, No. 2, 2012, pp. 143-160.

[5] Symp. on Principles of Distributed Computing. New York: ACM Press, 2002, pp. 183−192. http://www.podc.org/podc2002/.

[6] Alok Nandan, Michael G. Parker, Giovanni Pau and Paola Salomoni, “On index load balancing in scalable P2P media distribution”, Multimedia Tools and Applications, Vol. 29, No. 3, 2006, pp. 325-339.

[7] Egemen Tanin, Aaron Harwood and Hanan Samet, “Using a distributed quadtree index in peer-to-peer networks”, The VLDB Journal, Vol. 16, No. 2, 2007, pp. 165-178.

[8] Christos Doulkeridis, Akrivi Vlachou, Kjetil Nørvåg, Yannis Kotidis and Michalis Vazirgiannis, “Efficient search based on content similarity over self-organizing P2P networks”, Peer-to-Peer Networking and Applications, Vol.3, No. 1, 2010, pp. 67-79

[9] Xu Ke, Song Meina and Song Junde, “An improved P2P lookup protocol model”, Cluster Computing, Vol. 13, No. 2, 2010, pp. 199-211.

[10] Thomas Moscibroda, Stefan Schmid and Roger Wattenhofer, “Topological Implications of Selfish Neighbor Selection in Unstructured Peer-to-Peer Networks”, Algorithmica, Vol. 61, No. 2, 2011, pp. 419-446.