Volume 3, Issue 12, December 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Parallel Search in Structured Chord Protocol for Quick Resource Discovery in Grid Computing

Jeyabharathi[*]
Ph.D Scholar
Manonmaniam Sundaranar University
Tirunelveli, India

Pethalakshmi
Associate Professor and Head of Computer Science
M.V.M Government Arts College (W)
Dindigul, India

Abstract— A grid is large-scale virtual organization which solve complex scientific and compute-intensive problems. As grid technique is growing rapidly in recent years, large-scale grid systems appear to provide the capability of flexible, secure, coordinated resource sharing, and problem solving among dynamic virtual organizations. These systems consequently are required to manage a large amount of related resources. In particular, the shared grid resources can vary from plain desktop systems to clusters and from storage devices to large datasets, even grid service could be seen as an extension of grid resource. An efficient resource discovery mechanism is one of the fundamental requirements for grid computing systems, as it aids in resource management and scheduling of applications. Among various discovery mechanisms, Peer-to-Peer (P2P) technology witnessed rapid development and the key component for this success is efficient lookup applications of P2P. Chord is a P2P structural model widely used as a routing protocol to find resources in grid environment. In this paper, we discuss multi-ring chord with parallel search for quick resource discovery in Grid Computing. We have applied fuzzy classification and Intruder Detection techniques for splitting the chord ring into multiple rings. We conclude that our work reduces the searching time efficiently.

Keywords—Grid Computing, Resource Discovery, Parallel Search, Chord, Fuzzy Classification

I. Introduction

Grid computing is the collection of computer resources from multiple domain systems to achieve a common goal. In a Grid computing, resource discovery is a very important aspect. When a client requests for resource, along with the request it presents a set of attributes that should be satisfied by the resource provider. The resource discovery process may be responsible for generating a set of best possible services for the given set of requests. In grid resource discovery there are two distinct concepts [1] are used. One is to identify the resource and other is to allocate the resource for the requesting system. To discover the resources from a grid system there are lot of resource discovery mechanisms are used. These mechanisms identify the resources from the grid system and then allocate the resources to the requesting system in the grid environment.

The shared grid resources can vary from plain desktop systems to clusters and from storage devices to large datasets, even grid service could be seen as an extension of grid resource. Therefore, Grid Resource Discovery (GRD) plays a crucial role in the whole system [2], and its discovery Models and strategies have a vital influence on the grid system performance.

GRD can be classified into attribution matching and semantic matching pattern. In attribution matching pattern, GRD can also be classified into Centralized, Hierarchical, Peer-to-Peer based, and group-clustering models from the system architecture aspect. Among the models given above, centralized and hierarchical models are easier to design and manage, whereas they prove to be inefficient as the scale of grid system rapidly increase due to the bottleneck of servers and single point of failure. Therefore, the P2P-based model which combines the P2P technology into grid is applied widely to overcome those problems.

Grid and P2P [3] are the two most common types of resource sharing systems currently in wide use. These two systems are evolved from different communities and serve different needs. Grid systems interconnect computer clusters, storage systems, instruments, and in general available infrastructure of large scientific computing centres in order to make possible the sharing of existing resources, such as CPU time, storage, equipment, data and software applications. Most Grid systems are of moderate-size, are centrally or hierarchically administered and there are strict rules governing the availability of the participating resources. In P2P system most resource location queries are not attribute-dependent as in Grids but they either specify a file name, i.e., keyword searches or range queries.

P2P network is a kind of distributed network, the network's participants to share their own part of the hardware resources (processing power, storage capacity, network connectivity, printer, etc.), which shares the resources required by the network services and content, can be directly accessed to by other peer nodes without intermediate entities. P2p network can be divided into centralized P2P networks, unstructured P2P networks and structured P2P networks. The structured P2P networks based on a DHT, which has the advantage of distributed and avoided the drawback of unstructured P2P system. It uses a structured hash algorithm, stored keywords to different nodes in the network by certain rules, through the corresponding routing algorithm to get the keyword. In a Distributed Hash Table, the more well-known agreement is Chord [10], Pastry [8], Tapestry [6], CAN [9] and so on. Chord algorithm is relatively simple and easy to achieve. Among structured P2P lookup protocols, Chord has been researched and applied widely as a result of its simplicity, high efficiency and credibility. The rest of this paper covers distributed hash tables in section 2 and Chord protocol in section 3. In section 4, we discuss about parallel search in Chord and we conclude this section in 5.

II. DISTRIBUTED HASH TABLE

A distributed hash table (DHT) is an infrastructure to support resource discovery in large distributed systems. In a DHT, data items such as resources, indexes of resources or resource metadata are distributed across an overlay network based on a hash function. Distributed hash tables (DHT) [4, 5] are a decentralized lookup scheme that aims to provide scalable lookups with high result guarantee for large distributed systems consisting of multiple peer nodes. A DHT, as with a hash-table data structure, provides an interface to retrieve a key value pair. A *key* is an identifier assigned to a resource. Traditionally this key is a hash value associated with the resource. A *value* is an object to be stored into the DHT. This could be the shared resource itself (file), an index (pointer) to a resource, or a resource metadata. A DHT achieves its design objectives including lookup scalability and result guarantee through Key-to-Node mapping, Data-Item Distribution and Structured Overlay Network. Chord, Pastry, Tapestry and CAN are well known DHT based protocols.

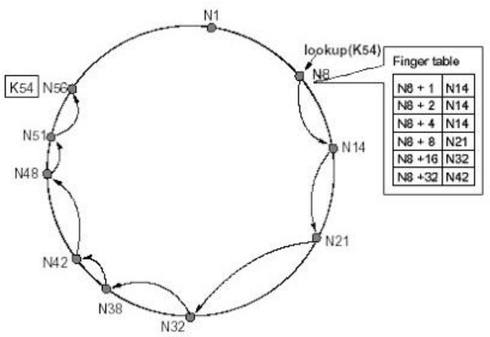
Tapestry [6] is a self-organizing routing and object location system, is based on Plaxton [7]. To detect link and server failures during normal operations, it can rely on TCP timeouts. Tapestry assigns multiple roots to each object. When locating an object, Tapestry performs the same hashing process with the target object ID, generating a set of roots to search. It also supports some dynamic operations. *Pastry* [8] is a location protocol sharing many similarities with Tapestry. Key similarities include the use of prefix/suffix address routing, and similar insertion/detection algorithms, and similar storage overhead costs. Objects in pastry are replicated without controlling by the owner. Upon “publication” of the object, it is replicated and replicas are placed on several nodes whose nodeIDs are closest in the namespace to that of the object’s object ID. It assumes that clients use the object ID to attempt to route directly to the vicinity where replicas of the object are kept. *CAN* [9] is another distributed and structured P2P lookup services. Each key will be evenly hashed into a point of d-dimensional space as its identifier. When a node joins, it will randomly select a point of d-dimensional space. Then it will be responsible for half of regions this point belongs to, and hold all keys whose IDs belong to this region. *Chord* [10] is a decentralized P2P lookup service that stores key/value pairs for distributed data items which is going to be discussed throughout this paper.

III. CHORD PROTOCOL

Chord is proposed by Ion Stoica of the University of California and Robert Morris of lab of MIT et al., as a resource routing protocol based on DHT. The Chord protocol [10] supports just one operation: given a key, it maps the key onto a node. Depending on the application using Chord, that node might be responsible for storing a value associated with the key. Chord uses a variant of consistent hashing to assign keys to Chord nodes. In Chord, each node needs “routing” information about only a few other nodes. In the steady state, in an N-node system, each node maintains information only about O(log N) other nodes, and resolves all lookups via O (log N) messages to other nodes. Chord maintains its routing information as nodes join and leave the system; with high probability, each such event results in no more than O($\log^2$ N) messages. Three features that distinguish Chord from many other peer-to-peer lookup protocols are its simplicity, provable correctness, and provable performance. A Chord node requires information about O(log N) other nodes for efficient routing, but performance degrades gracefully when that information is out of date. This is important while practising because nodes will join and leave arbitrarily, and consistency of even O (log N) state may be hard to maintain. Only one piece of information per node need be correct in order to guarantee correct routing of queries.

A. Consistent Hashing

The consistent hash function assigns each node and key an m bit identifier using a base hash function such as SHA-1. A node’s identifier is chosen by hashing the node’s IP address, while a key identifier is produced by hashing the key. The identifier length m must be large enough to make the probability of two nodes or keys hashing to the same identifier negligible. Consistent hashing assigns keys to nodes as follows. Identifiers are ordered in an identifier circle modulo 2^m. Key *k* is assigned to the first node whose identifier is equal to or follows in the identifier space. This node is called the successor node of key *k*, denoted by successor (*k*). If identifiers are represented as a circle of numbers from 0 to 2^m -1, then successor (*k*) is the first node clockwise from *k*.



N8 + 1	N14
N8 + 2	N14
N8 + 4	N14
N8 + 8	N21
N8 +16	N32
N8 +32	N42

Fig 1. Chord Ring with 64 nodes

Fig 1 shows an identifier circle with *m*=6. The circle has ten nodes: 1,8,14,21,32,38,42,48,51 and 56. The successor of identifier 1 is node 1, so key 1 would be located at node 1. Similarly, key 5 would be located at node 8, and key 53 at node 56.Consistent hashing is designed to let nodes enter and leave the network with minimal disruption. To maintain the consistent hashing mapping when a node n joins the network, certain keys previously assigned to n's successor now are assigned to n. When node n leaves the network, all of its assigned keys are reassigned to n's successor. In the example given above, if a node were to join with identifier 54, it would capture the key with identifier 53 from the node with identifier 56.

B. Existing Variations of Chord

Till recently, researchers have proposed many improved schemes to enhance routing efficiency of Chord Protocol. To deal with the routing delay and ignorance of physical topology information in P2P, [11] presents a topology-aware structured P2P system, named TB-Chord, which applies a suit of mechanisms to extend Chord to optimize the utilization of physical network topology and overlay network. The TB-Chord makes effective use of network topology structure during network routing. The result of experiments show that the TB-Chord, compared with traditional Chord, has obvious improvements in the routing delay and the hops of overlay networks. At the same time, TB-Chord does not need super node or plus layer which will pay more maintenance costs. Fan chao et al., [12] proposed BNN-Chord algorithm based on neighbors' neighbor (NN) chord. NN-Chord algorithm extends finger table using neighbors' neighbour link which is based as learn table. This table maintains information of the successor's successor node, which can reasonably increase finger density of routing table to find the neighbor node, which is more close to the object. The BNN-Chord algorithm effectively reduces the search path length and can improve the system performance.

Chunling Cheng et al [13] proposed advanced chord routing algorithm based on redundant information replaced objective resource table. The redundant information in the finger table of Chord ring not only takes valuable space, but also increases search delay. There is a direct idea that using inspired information to replace the redundant information can be used, so that each node can use the space effectively and establish links with much more nodes, and the idea can improve the search performance greatly. Towards an Efficient P2P system capable of processing multi-attribute Multi-keyword fuzzy-matching queries with High recall ratio and load balancing, *ZHAO Xiu-Mei et al.,* [14] proposed a new resource indexing model which is expanded from Chord and called MF-Chord. *Yufeng Wang et al.*, [15] analyzed the of Chord algorithm to reconstruct the finger table in Chord, in which counter clockwise finger table is added to achieve resource queries in both directions, and the density of neighboring fingers is increased. Additionally, AB-Chord implements a new operation to remove the redundant fingers introduced by adding fingers in AB-Chord. In comparison with original Chord (and Bi-Chord), new fingers are inserted into the middle of two neighboring fingers in clockwise and counter clockwise direction. *Huayun Yan et al.,* [16] modified the finger table of original Chord, and modified all the places which relate to the entry of the finger table, such as the node join, routing procedure etc. Huayan Yan et al., get a very perfect routing result through experiment and found that the routing hops is close to a constant number. The cost is more updating when joining a node and deleting a node in the system, and nearly double the size of finger table. This system will be more effective in a comparatively steady condition.

IV. PARALLEL SEARCH IN CHORD

Chord is a DHT implementation that supports *O (*log *N)*-hop lookup path length and *O(*log *N)* routing states per node, where *N* denotes the total number of nodes. Chord organizes nodes as a ring that represents an *m*-bit one-dimensional circular identifier space, and as a consequence, all arithmetic operations are modulo 2*m*. To form a ring overlay, each node *n* maintains two pointers to its immediate neighbors. The two neighbors are predecessor and node's successor. So Chord searches the resources in the ring by using its finger table.

In this method we apply Fuzzy classification which classifies elements into a fuzzy set and its membership function is defined by the truth value of a fuzzy propositional function. The Chord is constructed initially with 2^m nodes. The ring is divided into three rings according to the basic Fuzzy- rule. Nodes with more than 66% of resources are grouped in HOTTEST RING. Nodes with resources between 34 to 65% are grouped into HOTTER RING and nodes with less than 34% resources are allocated to the HOT RING. The goal of this method is to create a model which includes all the necessary conditions to clearly differentiate each ring. Whenever a new node enters, the algorithm correctly predicts the corresponding ring and the node will join into the desired place. In this place Divide-and-Conquer learning concept is applied to split the nodes into subsets and this process is recursively executed for the subset of nodes.

Fuzzy numbers are fuzzy subsets of the real line. They have a peak or plateau with membership grade 1, over which the members of the universe are completely in the set. The membership function is increasing towards the peak and decreasing away from it. The following figure depicts the fuzzy membership function for three groups of nodes.

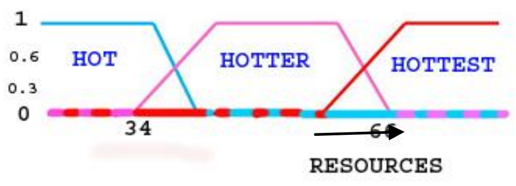


Fig.2 Fuzzy Membership Function for Three Rings

Now the 2^m nodes are grouped into three rings according to the classification procedure of decision tree algorithm with the help of basic fuzzy logic properties. The splitting is based on the number of resources each node has. In each ring, Ring-Head is selected based on the strength of the resources. Ring-Head is the entry point to each ring. If a query is started by the node of any ring, the Ring-Head checks its own ring and simultaneously sends the query to the Ring-Head of the remaining two Ring-Head. So the resources are simultaneously searched in all the three rings. The lookup process may successfully end in the ring which originates the query or any one of the two rings may find the resource. If the resource is located in the originated ring, the process is similar to the normal lookup process. Otherwise, the node which stores the key will return the result to its Ring-Head and in-turn the Ring-Head submits the result to the corresponding node via the head of the corresponding ring. As far as the lookup process is concerned, this method will find the key through parallel search and reduces the communication time dramatically. We named this method as FZ-Chord.

After splitting the rings, Resource Table is constructed by getting details of nodes from the finger table of each node. The entries of the table are node.id, number of resources and node status. Node status stores the status of resources of each node.id.

TABLE I
THE STRUCTURE OF RESOURCE TABLE

Node.id	No. of Resources	Node Status

Entropy is used at the time of creating Resource Table to avoid duplicate entries of resources. If more than one node maintains the same set of resources, it will be handled by entropy method to put single entry about those resources. For example, if nodes N2, N4, N9 have RAM with 1GHZ speed, Resource Table will have a single entry for these three nodes. Entropy finds this redundancy and makes the Resource Table meaningful.

Chord Ring with 2^7 (128) nodes is divided into three Rings as shown in fig 3.

Example 1:

Node 31 searches the key 75 which is in the HOT RING, the lookup process follows the following steps.

Step 1: Node 31 searches its own Resource Table and simultaneously it sends the request to its Ring-Head.

Step 2: Now Ring-Head of HOTTEST RING sends request to the Ring-Heads of Hotter Ring and Hot Ring.

Step 3: The search process is simultaneously going on with the help of Ring-Heads.

Step 4: Key 75 is located in the HOT RING and the corresponding Ring-Head sends the result to Node 31 through the Ring-Head of HOTTEST RING.

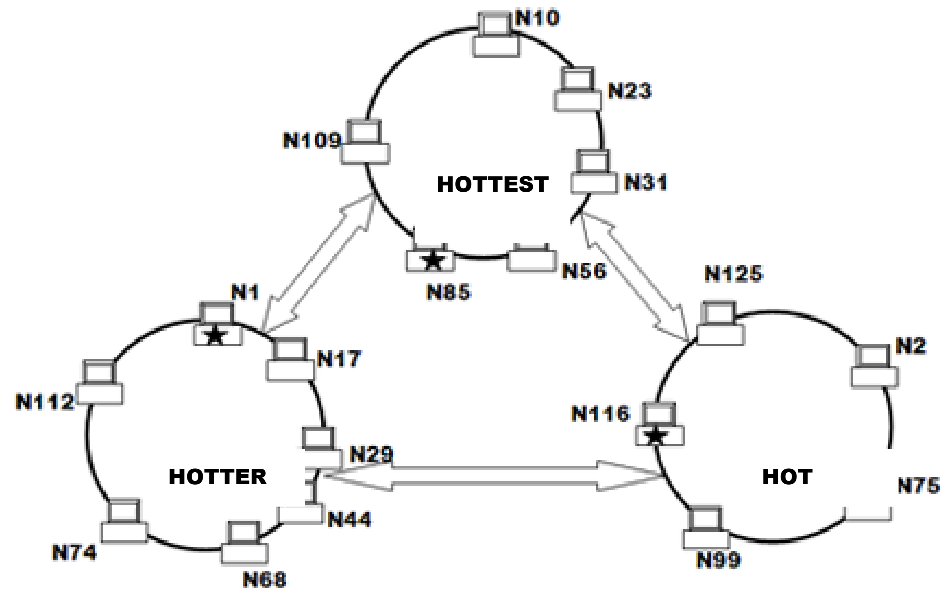


Fig 3. Three Chord Rings after Fuzzy classification

Example 2:

Node 44 searches key 109 which is actually located in HOTTEST RING. The search process is as follows

Step 1: Node 44 searches its own Resource Table and simultaneously it sends the request to its Ring-Head.

Step 2: Now Ring-Head of HOTTER RING sends a query request to the Ring-Heads of HOTTEST RING and HOT RING.

Step 3: The search process is simultaneously going on with the help of Ring-Heads.

Step 4: Key 109 is located in the HOTTEST RING and the corresponding Ring-Head sends the result to Node 44 through the Ring-Head of HOTTER RING.

A. Resource Based Intruder Detection (RID)

Node with unique resources is also included in the HOTTEST RING. For this purpose intrusion-detection method is used. Intrusion Detection is the act of detecting actions that attempt to compromise the confidentiality, integrity or availability of a resource. More specifically, the goal of intrusion detection is to identify entities attempting to subvert in-place security controls. The Common types of Intrusion Detection are Network Based (Network IDS), Host Based (HIDS) and Resource Based Intruder Detection (RID).

In a typical network, many of the nodes share the same system specifications. If there is some node with odd configuration, then that is probably an intruder. Any entry level intruder detection system will identify this and raise an alarm. If the node is authorized, then the system will be instructed to treat the intruder detection system alert as FALSE Alarm. But in many cases, the node will be discarded by the network to ensure security. So Any Elementary Intruder detection System can be used to filter nodes with odd resource.

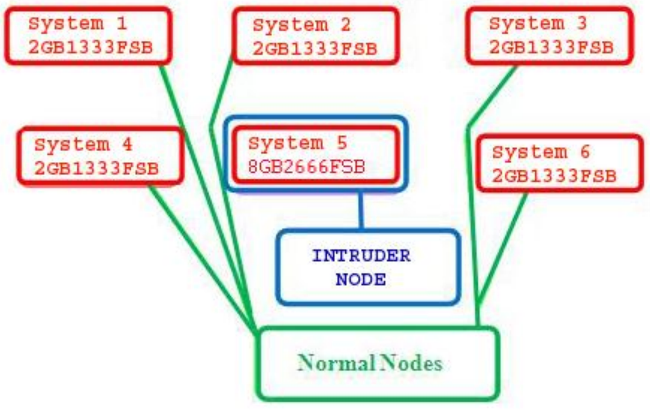


Fig 4. An Example of Intrusion Detection

Fig 4 explains the typical network with Node characteristics and the Application of RID. System 5 is identified as unique node (Intruder node) which stores the unique resource. The RID algorithms classify nodes with unique resources or significant differences in resource. This can be used for easy classification of resources. Unique nodes are nodes with special resources which are not available in any of the nodes in the ring. These nodes are also called as intruder nodes. Intruder nodes are identified and incorporated in the HOTTEST RING. The Ring-Head maintains the location of unique node in its Resource Table.

V. CONCLUSION

Grid computing enables systems to share geographically distributed resources as they pursue common goals. In large-scale Grid systems, discovery of heterogeneous resources is crucial to achieving scalable performance. The Grid resources need to be discovered, selected and invoked quickly and efficiently in order to satisfy the needs of a demanding environment. In recent years a lot of attention has been paid to the utilization of P2P technologies in Grid Resource Discovery. Since P2P approach will be suitable for dynamic distributed searching in a large-scale Grid environment, our research focuses on structured P2P protocols. As Chord is a good choice for single keyword search, we experiment Chord by applying fuzzy classification and intruder detection to split it into three rings and search simultaneously all the rings for quick discovery. This method reduces number of hops, number of message and communication time to search a resource.

REFERENCES

[1] Regnard Viven C and Shamila Ebenezer A, "A Survey on Resources Discovery Approaches in Distributed Computational Grids," International Journal of Computer Science and Management Research Vol 1 Issue 4 November 2012, ISSN 2278-733X.

[2] R.Buyya and S.Venugopal, "A gentle introduction to grid computing and technologies", CSI communications, vol.29, pp.9-19, July 2005.

[3] P. Trunfio, D.Talia, C. Papadakis , P. Fragopoulou, M. Mordacchini , M. Pennanen, K. Popov, V.Vlassov, S. Haridi , "Peer-to-Peer Resource Discovery in Grids: Models and Systems", This research work is carried out under the FP6 Network of Excellence CoreGRID funded by the European Commission (Contract IST-2002-004265). Preprint submitted to Elsevier Science ,3 August 2006.

[4] Gummadi, K., Gummadi, R., Gribble, S., Ratnasamy, S., Shenker, S., Stoica, I., " The impact of DHT routing geometry on resilience and proximity," In Proc. of ACM SIGCOMM, pp. 381–394.ACMPress, Germany (2003).

[5] Ratnasamy, S., Stoica, I., Shenker, S., "Routing algorithms for DHTs: some open questions. In Proc. of the 1st Intl. Workshop on Peer-to-Peer Systems", (IPTPS'02), pp. 45–52. Springer-Verlag, USA (2002).

[6] B.Y.Zhao, J.Kubiatowicz and A.Joseph, "Tapestry: An Infrastructure for fault-tolerant wide-area location and routing," Technical Report UCB/CSD-01-1141, University of California at Berkeley, Computer Science Department, 2001.

[7] C.Greg Plaxton,Rajmohan Rajaraman and Andrea W.Richa, "Accessing nearby copies of replicated objects in a distributed environment," in proceeding of ACM SPAA.ACM, June 1997.

[8] A.Rowstron, P.Druschel, (2001) "Pastry: Scalable, decentralized object location and routing for large scale peer-to-peer systems," in Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg, London, pp 329-50.

[9] S.Ratnasamy, P.Francis, M.Handley, R.Karp, S.Shenker, (2001) "A scalable content-addressable network," in Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, California, pp 161-72.

[10] Ion Stoica, Robert Morris, I.Stoica, D.L.Nowell, D.R.Kargar, M.F.Kaashoek, and H.Balakrishnan "Chord: A Scalable Peer-to-Peer lookup service for internet applications," IEEE/ACM Transactions on Networking, vol 11, issue 1, pp 17-32, Feb 2003. (First appeared in Proc. ACM SIGCOMM conference on Applications, Technologies, Architectures, and protocols for Computer Communication, pp. 149-160, 2001.)

[11] Wei Lv1, Qing Liao2, Jingling Zhao3, Yonggang Xiao "TB_Chord: An Improved Routing Algorithm to Chord Based on Topology-aware and Bi-Dimensional Lookup Method", 978-1-4244-3693-4/09/$25.00 ©2009 IEEE.

[12] Fan chao, Hongqi Zhang, Xuehui Du, Chuanfu Zhang, Zhengzhou, "Improvement of Structured P2P Routing Algorithm Based on NN-Chord", 7th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM), 2011.

[13] Chunling Cheng, Yu Xu, Xiaolong Xu, " Advanced Chord Routing Algorithm Based on Redundant Information Replaced and Objective Resource Table", in the proceedings of Computer Science and Information Technology (ICCSIT), IEEE,2010.

[14] ZHAO Xiu-Mei, LIU Fang-Ai, Jinan "MF-Chord: Supporting Multi-Attribute Multi-keyword Fuzzy-Matching Queries", ITIME '09. IEEE International Symposium on IT in Medicine & Education, (Volume:1).2009.

[15] Yufeng Wang, Xiangming Li " AB-Chord: an efficient approach for resource location in structured P2P networks", 9th International Conference on Ubiquitous Intelligence and Computing and 9th International Conference on Autonomic and Trusted Computing , 2012.

[16] Huayun Yan, Yunliang Jiang , Xinmin Zhou1, "A Bidirectional Chord System Based on Base-k Finger Table", International Symposium on Computer Science and Computational Technology,2008.