Journal of Zhejiang University SCIENCE A
ISSN 1009-3095 (Print); ISSN 1862-1775 (Online)
www.zju.edu.cn/jzus; www.springerlink.com
E-mail: jzus@zju.edu.cn

Study on semantic-oriented hybrid indexing strategy of resource metadata in peer-to-peer network

LIU Zhen†, ZHOU Mao-wei, HUANG Hong-bin, LUO Xue-shan
(*College of Information System and Management, National University of Defense Technology, Changsha 410073, China*)
†E-mail: hero12251976@163.com

Received Apr. 10, 2006; revision accepted Aug. 31, 2006

Abstract: Applying ontology to describe resource metadata richly in the peer-to-peer environment has become current research trend. In this semantic peer-to-peer environment, indexing semantic element of resource description to support efficient resource location is a difficult and challenging problem. This paper provided a hybrid indexing architecture, which combines local indexing and global indexing. It uses community strategy and semantic routing strategy to organize key layer metadata element and uses DHT (distributed hash table) to index extensional layer metadata element. Compared with related system, this approach is more efficient in resource location and more scalable.

Key words: Peer-to-peer (P2P), Ontology, Metadata, Resource indexing
doi:10.1631/jzus.2007.A0088 **Document code:** A **CLC number:** TP391

INTRODUCTION

In some large scale corporations and virtual organization environment (education system, military organizations, etc.), much electronics information resources are distributed physically and diverse in types. Each unit of an organization manages some resources respectively, according to the domain knowledge or the particular mission. In technical aspect, each unit is a node working independently in the information network. These nodes can join or leave the network at any time, without paying attention to states of other nodes. It is the same as that of peer-to-peer (P2P) network in system management without centralized control. Managing indexing and distributing the information resources for global sharing is a critical problem to be solved.

The existing P2P systems, such as Gnutella (Matei *et al*., 2002), Napster (http://www.napster.com), etc. of the first generation and PAST (Rowstron and Druschel, 2001a), Pastry (Rowstron and Druschel, 2001b), CFS (Dabek *et al*., 2001), etc. of the second generation based on DHT (distributed hash table), deal mainly with indexing the document key words and global identifiers (such as filenames). The capability of query is limited. They cannot query or locate semantic-based or diversified information resources. As a result a series of semantic-based P2P network emerged such as EDUTELLA (Nejdl *et al*., 2002), SWAP (Ehrig *et al*., 2003), etc. They adopt RDF to describe the shared metadata and related semantic contents and increase the ability of sharing resources. EDUTELLA adopts super-peer based management mechanism, clustering the resources based on ontology, formula and query and adopts a Hypercube topology to ensure the uniqueness and efficiency of traversing and adopts the broadcast strategy among the super-peers. SWAP adopts a non-structure topology and costs less in topology maintaining. It provides a REMINDIN-based query strategy which is a mechanism of routing and transmitting based on learning. In the previous existing P2P system, indexing of distributed resources is focused on semantic interactive manipulation. These systems applied local-indexing strategy and their query routing strategy is blindfold in some degree. When the nodes scale increases, the system efficiency decreases rapidly and the scalability is poor.

According to existing P2P systems, the resources indexes can be classified as concentrated index, local index and global index. Concentrated indexing strategy commonly uses a centralized server to manage resource indexes, and is apt to become the system bottleneck. Local indexing strategy aims at non-structure network topology. Metadata are partitioned based on resources object space, and the term lists of each resource are stored on a node. According to this strategy, query is broadcast to all nodes. The disadvantage is that all the nodes are involved during the query, which leads to poor scalability. Global indexing strategy aims at the structured topology, and metadata are distributed based on terms. Each node stores the complete inverted list of some terms. Query containing multiple terms is sent to nodes responsible for those terms. In this way resource locating is efficient and the system is scalable, but the communication cost for a join of multi-terms grows proportionally with the length of inverted lists.

For semantic-oriented sharing of resources in the decentralized environment, a semantic-oriented resource-sharing platform based on peer-to-peer resource-agents (SORSPPPRA) is proposed. The resource-agent network manages heterogeneous resources of different organizations. Fig.1 shows the organization architecture of resources. Each agent manages one or several ontologies established at the comprehension of the related domain. Resource metadata is described based on some ontology in any resource agent N and registered to the resource agent N. Engine Node is used to index the location of resources in the Agent Node.

This paper focuses on the research of indexing strategy of distributed resources in order to bring forward a hybrid indexing mechanism to support the efficient location in peer-to-peer networks. It combines the characteristics of the local indexing and global indexing and uses community strategy, DHT and semantic routing to index the semantic elements of the core-layer and the extensional-layer metadata, which improves the capability of resource discovery and system scalability. Section 2 describes the resource metadata model and querying model. Section 3 explains the frame of the system indexing mechanism and core technology. Section 4 analyzes the system performance with simulation. Section 5 forms a conclusion and discusses prospects.

METADATA MODEL

In this paper, resource metadata are described by an ontology-based approach. Metadata are described in core layer and extensional layer. Core layer

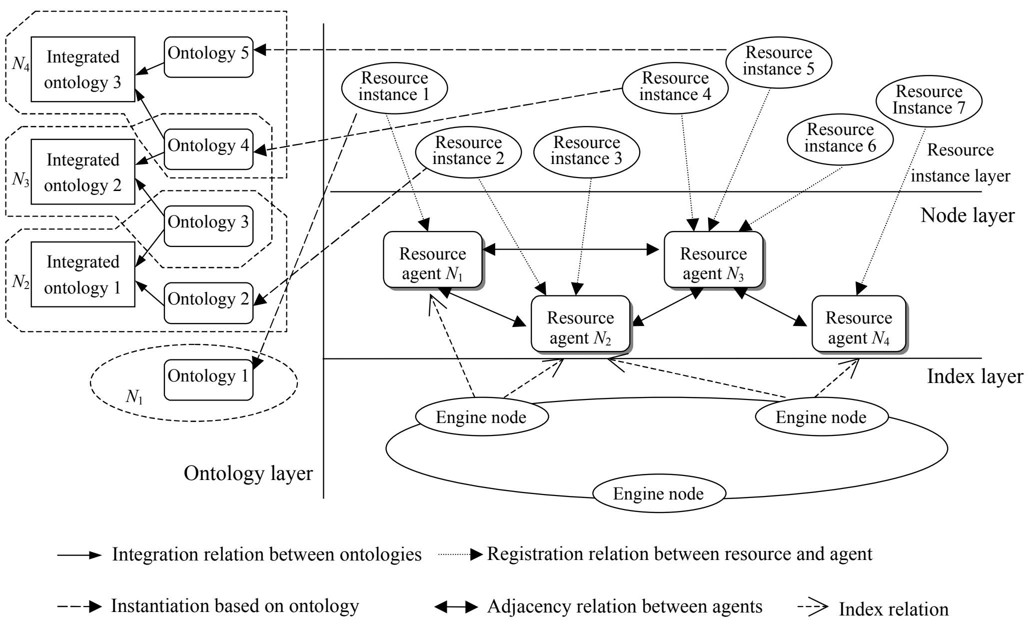

Fig.1 Architecture of multi-ontology based resource organization

metadata are included in all the resource descriptions describing the global feature of resources and including resource management element, summary content element, format element and security element. Extensional layer metadata comprise the extent of core layer metadata, provide task-oriented or domain-oriented semantic content description of resources and will improve efficiency and precision of information resource locating. Following this, the simple description of ontology structure and metadata model of resource objects is provided.

Definition 1 Metadata model of resource object $M_{\text{object}}=(C_{\text{key}},C_{\text{ext}},A,R,P,I)$. Assume that T is a set of terms. D_{basic} is a set of pre-defined basic data types. C_{key} is a set of core-layer concepts. C_{ext} is a set of extensional-layer concepts. A concept $c\in C_{\text{key}}\cup C_{\text{ext}}$ is defined as $c=(t_c,A_c)$, where $t_c\in T$, $A_c\subseteq A$. A_c is a set of attributes of c. A is a set of all attributes. Attribute $a\in A$ is defined as $a=(t_a,dt_a)$, where $t_a\in T$, $dt_a\in D_{\text{basic}}\cup C_{\text{ext}}$. R is a set of semantic relations. $\forall r\in R$ is defined as $r<c,c'>$, c, $c'\in C_{\text{key}}\cup C_{\text{ext}}$. Primary types of semantic relations include: equivalence relation R_{same}, inherit relation H_{Inh}, aggregation relation H_{Aggr}, association relation RP. I is a set of resource instances of all concepts of node ontology.

Definition 2 Resource request model is defined as $Q=(C_Q,R_Q,q_c)$, C_Q, R_Q indicate the set of concepts and the set of relations that resource request refers to, respectively. q_c comprises the constraint conditions that requested resources must satisfy.

In Fig.2, an example of ontology, metadata and resource request about an education domain is presented.

INDEXING ARCHITECTURE AND WORKING MECHANISM

Fig.3 shows the general indexing architecture. The general idea is as follows. According to the metadata structure described in Section 2, each source describes the resource objects metadata and registers the metadata of the resources to the subject resource-agent node N_i. N_i manages the related resource objects and indexes the related semantic elements. In this system the indexing strategy is different between the resource core-layer metadata and extensional-layer metadata. As the core-layer metadata is a part of all of the resources, a community strategy is adopted to affiliate N_i to satisfy the special constraints

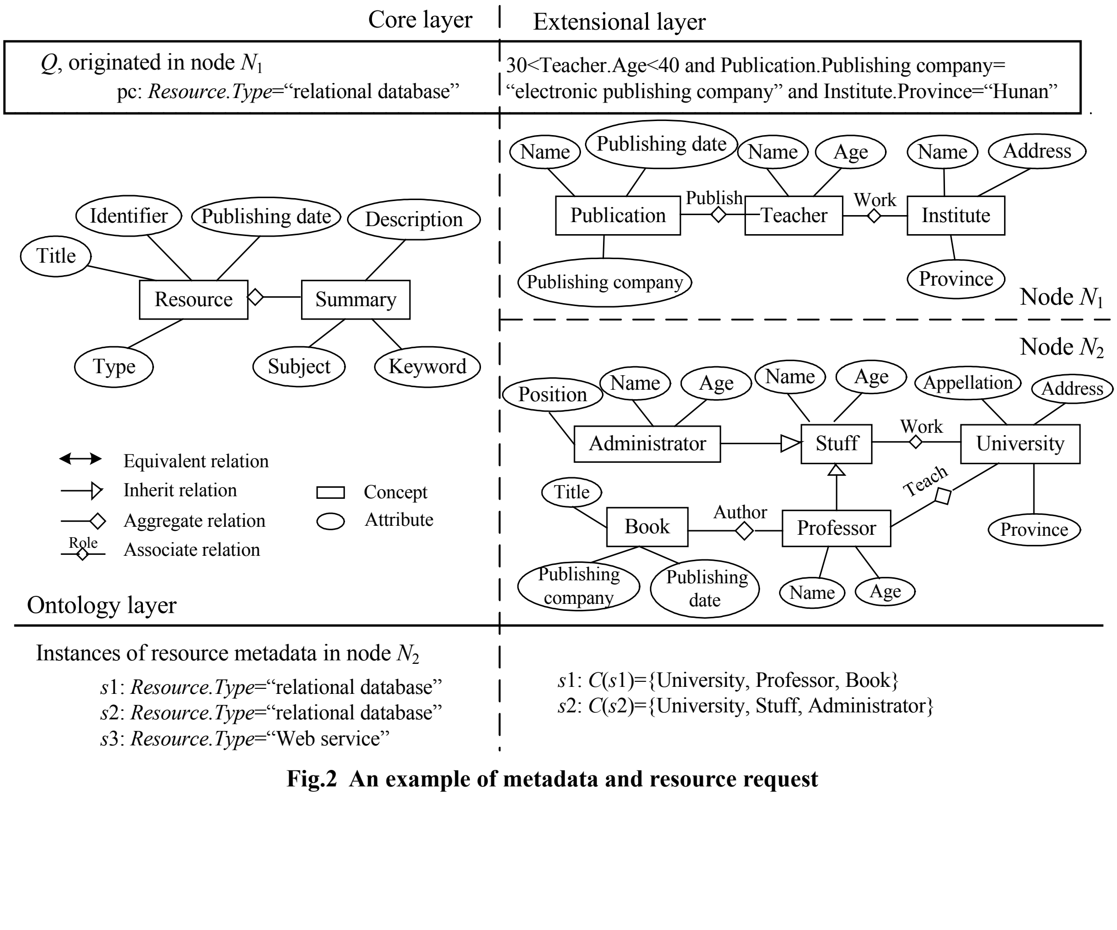

Fig.2 An example of metadata and resource request

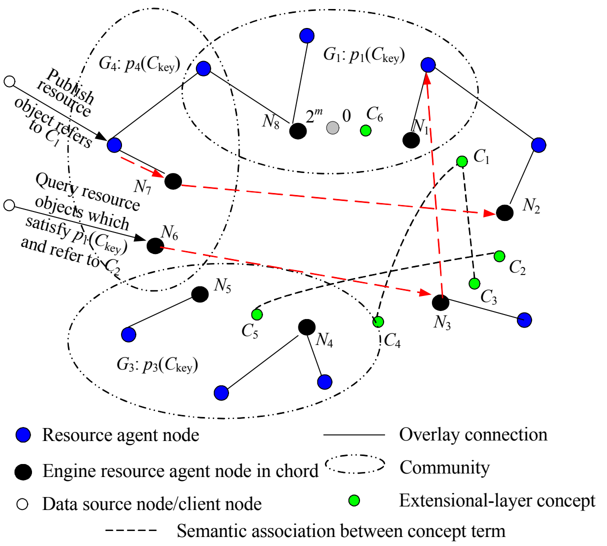

Fig.3 The general indexing architecture

to the community G_i. Hence, the efficiency of locating the community resources is improved. As extensional-layer metadata are not a part of all of the resource objects, the local indexing will limit the efficiency of location when the system scale increases. To improve the efficiency of extensional-layer metadata element location in the distributed environment, a DHT way combining the global indexing and the local indexing is used. This system uses Chord (Stoica *et al.*, 2001) to provide the related extensional-layer concepts with indexing services, and associate the nodes dynamically by the semantic relevancy between the concepts.

Community based core-layer metadata indexing

The resource-agent nodes are grouped into a community series based on some constraints $p_i(C_{\text{key}})$ referring to some core-layer metadata. For example, the concept element "Resource" of the core-layer metadata is used to partition the node space. Referring to constraint $p_1(\{Resource\})$="*Resource.Type*= database", all the nodes containing database resources are grouped into one community. Each community has a community identifier GID and community entrance constrains $p(C_{\text{key}})$. Different communities may contain some same nodes. When a new resource object r_i joins in the subjected resource-agent node N_i, N_i first analyzes r_i's metadata M_i and extracts the core-layer concepts and the extensional-layer concepts to deal with respectively. Dealing with the extensional-layer concepts is introduced in the following subsection. For the core-layer metadata $M_{r_i}^{\text{key}}=(C_{\text{key}},I_{r_i})$, $M_{r_i}^{\text{key}}$ is sent to all the communities through the connections among different communities and matches $p_i(C_{\text{key}})$ of the community G_i. When I_{r_i} satisfies p_i, N_i joins in G_i and the overlay connection to G_i is established.

Chord based extensional metadata hybrid indexing

DHT works by organizing nodes as a topology and distributing data to designated nodes. It uses a DHT to map a term to a node where the inverted list for the term is stored. Structured overlay network and DHT aim to provide efficient and scalable lookup performance in a distributed system. Chord (Stoica *et al.*, 2001) and CAN (Ratnasamy *et al.*, 2001) are typical DHT-based P2P systems. It provides a stronger result guarantee than unstructured overlay networks.

In SORSPPPRA, the extensional-layer metadata is not a part of all the resource objects, while it extends the resource metadata according to a special domain. As a result, it can distinguish resources more efficiently. It exactly and efficiently locates resources in the DHT way. This system uses Chord protocol for indexing extensional-layer metadata element. A series of engine nodes are chosen from resource-agent nodes to form a ring topology. The system takes SHA1 (National Institute of Standards and Technology, 1995) as a mapping function from extensional-layer concepts to engine nodes. The engine nodes should be stable and have enough capacity to offer the desired level of quality of service. The other nodes must connect to one engine node at least. Each engine node is mapped to a 0~2^m ring including a Successor List and a Finger Table. The Successor List lists the successor node of the current node n. In the Finger Table, the node ID has intervals with exponent of 2 and m items at most. The ith item stores a node entry s=*successor*(n+2^{i-1}). When extensional-layer metadata of the resource object r_i are issued, the extensional-layer concept c_i is extracted first and then mapped to the k_i ($0 \le k_i \le 2^m$) by SHA1. Concept c_i is managed by the successor node ID nearest k_i, and the concept list of extensional-layer metadata is broadcast to the nodes that are responsible for these concepts that r_i contains. Fig.4 shows the indexing structure of information resource in the engine node which combines the global and local indexing. By this way, the efficiency of query referring to multiple concepts is improved. To locate the approximate concepts, the engine nodes would trigger the corresponding semantic checking periodically. If the new concepts have joined in during the period, the engine node would compute the associate degree between the increment terms and the terms in other nodes and discover the semantic relations (synonymy, broad sense, and narrow sense) between different nodes. So the topology connections are generated according to these related semantic relations.

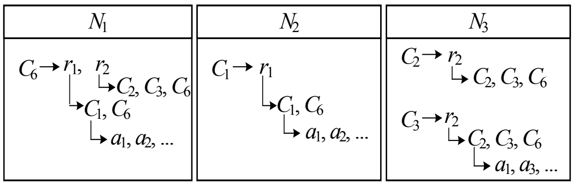

Fig.4 Hybrid indexing of extensional-layer concepts

Definition 3 $c_1, c_2, \ldots, c_k$ are indexed by $N_1, N_2, \ldots, N_k$, respectively. If $\exists SL_1(c_1,c_2)$, $SL_2(c_2,c_3)$, …, $SL_{k-1}(c_{k-1},c_k)$, where $SL_i(c_i,c_{i+1})$ is the semantic relation between c_i and c_{i+1}, $SL_i\in\{synonymy, broad\ sense, narrow\ sense\}$, so we call that c_1 has reachable semantic path with c_k, expressed as $RSP(c_1,c_k)$, and say that $c_1, c_2, \ldots, c_k$ are in the reachable semantic path. $|RSP(c_1,c_k)|$ is the length of reachable semantic path.

Working flow of resource discovery

The working flow of resource discovery is as follows:

Step 1: The client X can send a resource discovering quest to any resource-agent node N_i. The request structure is described as the query model structure described at the beginning of this section.

Step 2: N_i analyzes the query and extracts the core-layer concept set C_k and the core-layer concept constraints $p(C_k)$ and the extensional-layer concept set C_e.

Step 3: if $C_e=\varnothing$, locate the resource according to the community, the query Q is delivered among different communities. To judge whether the related community constraints $q_i(C_k)$ of G_i is compatible with $p(C_k)$, which means whether the instance satisfies the two constraints that exist, if so, then travel through the nodes in the community, return the resource objects ID which satisfies $p(C_k)$ to N_i and N_i returns the results to the client X. Otherwise the query is delivered to the next community until a special TTL is reached, or travels through all the communities.

Step 4: If $C_e\neq\varnothing$, N_i sends the query Q to an adjacent engine node N_j in the Chord network. According to Chord routing protocols, N_j delivers Q to all the engine nodes N_x responsible to the concept terms involved in C_e.

Step 5: N_x makes a local match, grades the matching scores, chooses the resource objects with score above a threshold and sends the query Q to the engine nodes responsible to the resource objects and judges whether it is compatible with the related community constraints of the nodes. If so, terminate the operation; else make a complete match including the core-layer, extensional-layer concepts and the instances. The matching algorithm is referred to in (Liu *et al.*, 2006). Then the resource objects with matching score meeting a special value are returned to N_i and sent to the client X at last.

Step 6: N_x is responsible to c_1 and has semantic relation between c_1 and c_2. If $\exists RSP(c,c_2)$, $c\in C_e$, c_1 is in the $RSP(c,c_2)$ and $|RSP(c,c_2)|<L$, where L is threshold of the length of reachable semantic path, N_x delivers the query Q to the related engine node N_y responsible to c_2.

Step 7: N_y executes the same operation as the N_x described in Step 5 and Step 6.

The queries involving extensional-layer concepts visit about $O(m\log_2 n)$ nodes in average. m is the number of extensional-layer concepts involved in query. N is the number of engine nodes. The queries that only involve core-layer concepts visit about $O(G+N_gG_m)$ nodes in average. G is the number of communities. G_m is the number of communities that satisfy the query in average. N_g is the number of nodes in each community in average.

To avoid dealing with the same query repeatedly and alleviate hot spot problem of the popular queries, the query result can be buffered on the initial resource-agent node and query routing nodes for a period. If a similar query arrives and the node has an existent result, the result is returned immediately without further routing.

Load balance

Now the computer storage capacity grows rapidly, but it could increase the cost of system storage and computation to maintain some specially popular extensional-layer concept terms. Thus a popularization threshold value (*PT*) can be designed at the engine nodes. When a certain concept term reverse list length (*TL*) of an engine node is bigger than *PT*, this engine node stops indexing this concept term (*t*) and stops dealing with the query referring to *t*. This operation will affect the system query performance little, because locating resources with too popular concepts cannot distinguish resources efficaciously and the query has little meaning.

Even if after carrying on corresponding process to some popular concepts, the indexed concept terms in each engine node are impossible to be completely balanced. The quantity of indexed resource in some engines is possibly very large. Simultaneously visiting times of the different terms are also not balanceable and computing cost at certain engine nodes is possibly very high. Thus system overall performance is influenced. Aiming at this problem, this paper takes a self-adaptive load balancing mechanism. Supposing the index key value scope of an engine node N_x is $[K_b, K_e]$. When the quantity of indexed terms and resources in N_x or the accessing frequency in unit time is higher than a certain threshold, an engine extension event is immediately triggered. First a key value K_t in $[K_b, K_e]$ is randomly produced. Then N_x uses routing strategy of random walk to search a node N_y that satisfies the condition to being an engine node (on-line time and load capacity satisfies specific condition) in the non-engine nodes, and K_t is used as N_y's node ID. Add N_y to the engine network and partition the N_x key value range. Thus N_x and N_y are separately responsible for $[K_b, K_t]$ and $[K_{t+1}, K_e]$. This self-adaptive load balance strategy will effectively balance engine nodes load.

SIMULATION

A simulation experiment platform RA_Chord is built by making some necessary changes to the Chord source code according to the system design. In this experiment, 100 resource-agent nodes are simulated, from which 32 nodes are chosen to be engine nodes. Simply design the core-layer metadata including a core concept "Resource" and construct 5 communities by "*Resource.Type*=database", "*Resource.Type*=picture", "*Resource.Type*=video", "*Resource.Location*=Changsha" and "*Resource.Location*=Other". Five hundred resources are formed. Resource ontology adopts common concepts in academic fields. Description of 500 extensional-layer metadata of resources is built based on the metadata model in this system. Each extensional-layer metadata has 20 concepts in average. The resources are distributed to 100 resource-agent nodes at random and the extensional-layer concepts are indexed to the related engine nodes. A semantic association dictionary is built up and the topology connections of the Chord engine nodes are built up according to the relationship of the terms. The existent P2P systems based on semantic schema adopt a non-structured topology generally, so in the experiment a non-structure P2P network, a kind similar to Gnutella is built to compare with RA_Chord. The routing strategy takes 5 Random_Walker scheme with TTL=8 to simulate. The Random_Walker chooses a node from the neighbor nodes of the current node to send the query until reaching the TTL value.

To show the efficiency of resource indexing in this system, this paper evaluates it through searching efficiency. The evaluation issues are as follows: (1) Hitting ratio is the proportion of successful searching. It is the ratio of the number of resources which are discovered successfully to the number of the real matching resources. In Fig.5a, 0, 40, 60, 80, 100 out of 100 nodes are chosen at random and 8, 16, 16, 32, 32 nodes are chosen as the engine nodes in RA_Chord. The related hitting ratios are gained respectively through experiment. From this, this system is affected little in searching efficiency with the increasing network scale. The performance is better than Random_Walker; (2) Searching cost. The searching delay time is used to measure the searching cost. In Fig.5b, 100 nodes and 32 Chord engine nodes are used. Ten queries are proposed for statistics. The hitting ratios in different time periods are compared. From Fig.5b, the efficiency in this system is obviously better than Random_Walker. That is to say this system can locate the resource in fewer hops.

From the simulation, the searching efficiency in this system is better than that in others. So does the scalability. But when the query does not involve the extensional-layer concepts, query routing is mainly based on the community's semantic relationship and as a result the efficiency is as much as other P2P systems based on communities (or interest domain).

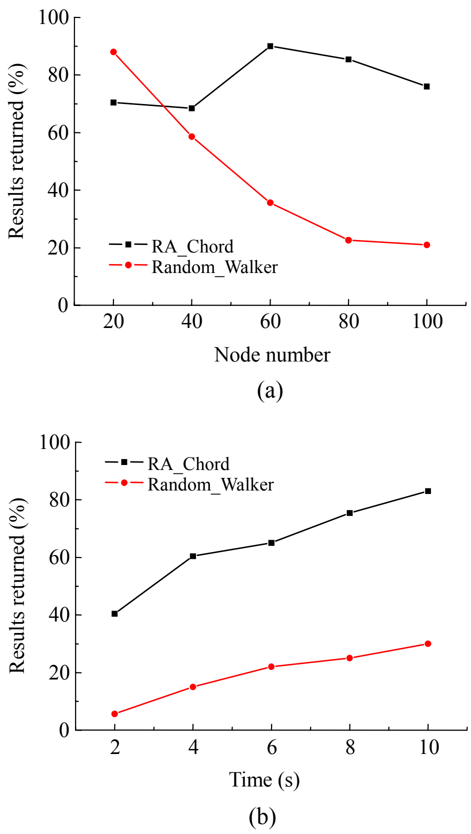

Fig.5 Comparison between RA_Chord and Random_ Walker. (a) Hit ratio; (b) Response time

CONCLUSION AND FUTURE WORK

This paper aims at semantic P2P system, a metadata description scheme based on ontology is proposed. From this foundation, a hybrid indexing architecture is provided. It is a resource semantic concepts-oriented indexing strategy which combines global and local indexing and adopts a community strategy, DHT and semantic routing to index semantic elements of the core-layer and extensional-layer metadata. The working mechanism of the system is introduced. This system improves the capability of semantic-oriented resource discovery efficiently. Simulation experiment showed that this system is better in hitting ratio and scalability than other existing P2P systems based on the semantics and schema.

This system combines structured peer network characteristic, as a result the cost of topology maintenance grows. How to concile the cost of maintenance with the efficiency of query needs further analysis. Meanwhile aiming at the indexing strategy which combines global and local indexing strategy, the index information dissemination is a critical step. So how to accelerate the distribution efficiency and reliability needs further study.

References

Dabek, F., Kaashoek, M., Karger, D., Morris, R., Stoica, I., 2001. Wide-area Cooperative Storage with CFS. SOSP'01. ACM Press, New York, p.202-215.

Ehrig, M., Haase, P., Siebes, R., Staab, S., Stuckenschmidt, H., Studer, R., Tempich, C., 2003. The SWAP Data and Metadata Model for Semantics-Based Peer-to-Peer Systems. Proc. MATES 2003. Springer, Erfurt, Germany, **2831**:144-155.

Liu, Z., Deng, S., Luo, X.S., Huang, H.B., 2006. Research on dynamic resource matching approach based on multi-ontology semantic similarity in peer-to-peer network. *Computer Science*, **31**(3):42-45.

Matei, R., Iamnitchi, A., Foster, P., 2002. Mapping the Gnutella network: properties of large-scale peer-to-peer systems and implications for system design. *IEEE Internet Computing*, **6**(1):50-57. [doi:10.1109/4236.978369]

National Institute of Standards and Technology, 1995. Publication 180-1: Secure Hash Standard.

Nejdl, W., Wolf, B., Qu, C., Decker, S., Sintek, M., Naeve, A., Nilsson, M., Palmér, M., Risch, T., 2002. EDUTELLA: A P2P Networking Infrastructure Based on RDF. Http://cid.nada.kth.se/pdf/CID-205.pdf

Ratnasamy, S., Francis, P., Handley, M., Karp, R., Shenker, S., 2001. A Scalable Content-addressable Network. Proc. ACM SIGCOMM. ACM Press, p.162-172.

Rowstron, A., Druschel, P., 2001a. Storage Management and Caching in PAST, a Large-Scale, Persistent Peer-to-Peer Storage Utility. SP'01. ACM Press, Banff, p.188-201.

Rowstron, A., Druschel, P., 2001b. Pastry: Scalable, Distributed Object Location and Routing for Large-Scale Peer-to-Peer Systems. IFIP/ACM International Conference on Distributed Systems Platforms (Middleware). Heidelberg, Germany, p.329-350.

Stoica, I., Morris, R., Karger, D., 2001. Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications. Proc. ACM SIGCOMM, p.149-160.