Scalable Keyword Search Based on Semantic in DHT Based Peer-to-Peer System

Wenhui Ma
College of Information Science and Technology, University of Nankai
Tianjin 300071, China
wenhuima_nk@hotmail.com

Wenfang Wang
College of Information Science and Technology, University of Nankai
Tianjin 300071, China
wwwfonline@eyou.com

Jing Liu
College of Information Science and Technology, University of Nankai
Tianjin 300071, China
jingliu@nankai.edu.cn

ABSTRACT

The common way for keyword search in Distributed Hash Tables (DHTs) based Peer-to-Peer (P2P) system is to construct distributed inverted index by keywords. But it suffers from the problem of unscalable resources (e.g. bandwidth, storage) consumption. In this paper, we present SKS, a scalable keyword search approach in DHTs based P2P system. SKS introduces the ontology to organize the specific domain, which captures the semantic relations between words. SKS constructs distributed inverted index by concepts, which decreases the number of index entries publishing for documents and avoids the intersection of inverted lists between nodes when executing multi-keyword search. With the concept index SKS transforms the keyword search to the match process of concepts, implementing semantic search. Simulation experiment shows that SKS is more efficient than the approach of distributed inverted index by keywords in indices publishing overhead and query overhead.

Categories and Subject Descriptors

H.3.3 **[Information Storage and Retrieval]**: Information Search and Retrieval – *Search Process*;

C.2.4 [**Computer Communication Networks**]: Distributed Systemsnn – *Distributed applications*;

General Terms

Algorithms, Design

Keywords

Peer-to-Peer, Distributed Hash Tables, ontology, search

1. INTRODUCTION

In Recent years, the P2P systems have emerged as a popular way to share huge volumes of data. The one of most challenging design aspects is efficient techniques for data retrieval. Unstrctured P2P systems such as Napster [1] and Gnutella [2] support keyword search through flooding query to some or all of nodes. Structured P2P systems, such as Chord [3] and CAN [4], has addressed some of the scalability and reliability problems that exist unstructured P2P systems through using DHTs, but they can not support full text search directly. While, as they actually implement DHTs over them, keyword search can easily be implemented by distributing inverted indices among nodes by keywords. Then query with multi-keyword can be executed through the intersection of inverted lists among nodes. However, [5] has demonstrated that the two techniques above suffer from the unscaleable resources consumption, such as storage and bandwidth.

In this paper, we propose a scalable keyword search approach in DHTs based P2P systems, called SKS. SKS introduces the ontology to organize the specific domain, and constructs inverted index by concepts. And each node of P2P system is responsible for inverted list of some concepts. SKS captures the semantic relations between words through ontology, which reduces the total overhead of system when index entries of documents publishing. For each document in the inverted lists of concepts, a set of ontological concepts that describe the content of the document is also stored in the node, which avoids the intersection of inverted lists among nodes when executing multi-keyword search. Moreover, SKS transforms the multi-keywords query to the match process of concepts, implementing keyword search based on semantic.

The remainder of this paper is organized as follows. Related work is discussed in Section 2. Section 3 describes the ontology, and document index constructing and query processing are discussed in section 4. Section 5 gives the simulation experimental results. Section 6 concludes the paper.

2. RELATED WORK

Some solutions have been proposed to overcome the unscalable resource consumption for keyword search in DHTs based P2P system. In [6] and [7], they partitioned the index by a set of keywords. This index scheme reduced the bandwidth consumption because no intersection of inverted lists is done among nodes, but it need considerably larger storage space than standard inverted index. Liu et al. in [8] proposed a set of mechanisms to provide a scalable keyword search. The focus of their work is to address common keywords problem. They used Fusion Dictionary and Keyword Fusion to balance unfair storage consumptions at peers and transform the user's query to contain more specific search keywords. So the query will associate with short list of files and generate lower network traffic. Tang in [9] proposes a eSearch, which combines the local index partitioned by document with the global index partitioned by term to construct a hybrid index structure in DHTs based P2P system. eSearch avoids the union operation from one node to another and reduces the query overhead. However, eSearch's search efficiency comes at the expense of publishing more terms. Although it uses several

* **Conference name:** Infoscale 2007, June 6-8, 2007, Suzhou, China
* **Copyright number (LaTeX \crdata{}):** 978-1-59593-757-5

optimizations, such as top term selection, to reduce the communication traffic and the storage space, the quality of query results may be degraded more or less. [10] has reduced bandwidth consumption by pursuing a hybrid between partitioning by keyword and partitioning by document, and implements keyword search using multi-level partitioning (MLP) in P2P system. However, MLP is designed and implemented on top of SkipNet [11], relying on a node group hierarchy, and it can not apply to other DHTs based P2P system.

3. ONTOLOGY

In computer science, the famous definition for ontology is Gruber's definition [12] "an ontology is an explicit specification of a conceptualisation". Therefore, an ontology defines a set of representational terms called concepts. Interrelationships among these concepts organize concepts in a hierarchically structure to describe a target domain. Ontology aims at defining meaning of the terms and relations among these terms for a specific domain, and providing a commonly understanding between users or applications for that domain.

Figure 1 shows a portion of ontology about computer science. This ontology is described by a Directed Acyclic Graph (DAG). Here each node in the DAG represents a concept. Each concept contains a label name(denoted by word) that is unique in this ontology and a set of synonyms. Each concept also has instances that are specific representation for domain knowledge. The nodes are linked by directed edges, which denote relations between concepts and specify in which way concepts are related to each other, such as "is-a", "part-of", or "instance-of" and so on.

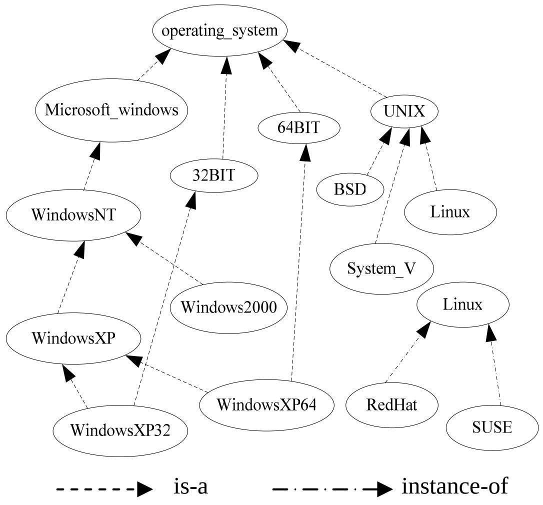
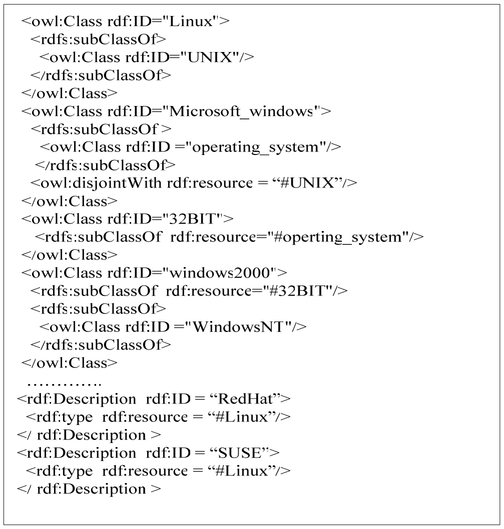


Figure 1. A small ontology about operating system

SKS adopts OWL [13] to represent domain ontology. It is extension of RDF Schema and adds language primitives to support richer expressiveness required. For the Figure 1, a snippet of its corresponding OWL code is shown in Figure 2.

4. INDEX CONSTRUCTING AND QUERY PROCESSING

Traditionally information retrieval system considers that each document's content can be described by a set of representative keywords. Thus the set of keywords is used to index and retrieve the document. This approach is also used in DHTs based P2P systems. Commonly keyword search in DHTs based P2P systems is implemented through constructing keyword index and distributes it among all nodes of system. But document indexing and retrieving through inverted index by keywords is usually assumed that the index terms are mutually independent, not consider the semantic relations between index terms. As a result the documents that do not contain any of the query keywords but semantic related the query request will not be retrieved, or unrelated documents that contain some of the query keywords will be retrieved.



```
 <owl:Class rdf:ID="Linux">
  <rdfs:subClassOf>
   <owl:Class rdf:ID="UNIX"/>
  </rdfs:subClassOf>
 </owl:Class>
 <owl:Class rdf:ID="Microsoft_windows">
  <rdfs:subClassOf >
   <owl:Class rdf:ID ="operating_system"/>
   </rdfs:subClassOf>
  <owl:disjointWith rdf:resource = “#UNIX”/>
 </owl:Class>
 <owl:Class rdf:ID="32BIT">
   <rdfs:subClassOf  rdf:resource="#operting_system"/>
 </owl:Class>
 <owl:Class rdf:ID="windows2000">
  <rdfs:subClassOf  rdf:resource="#32BIT"/>
  <rdfs:subClassOf>
   <owl:Class rdf:ID ="WindowsNT"/>
  </rdfs:subClassOf>
 </owl:Class>
  …………..
<rdf:Description  rdf:ID = “RedHat”>
 <rdf:type  rdf:resource = “#Linux”/>
</ rdf:Description >
<rdf:Description  rdf:ID = “SUSE”>
 <rdf:type  rdf:resource = “#Linux”/>
</ rdf:Description >
```

Figure 2. A snippet of OWL code for ontology

In this paper, SKS introduces domain ontology to capture semantic relations between keywords, and generates a set of concepts respectively to represent the content of each document and query. Thereby SKS constructs distributed inverted index by concepts instead of keywords. So, the number of index entries publishing for documents is decreased, and semantic search is implemented through the concepts match between the document and query.

4.1 Concept index Constructing

In SKS, construction of the distributed inverted index by concepts involves several steps.

1) Generating a set of keywords for each document, while "stop" words are eliminated, and the remaining words are stemmed so that there is only one grammatical form (or the stem common to all the forms) for a given word.

2) Mapping the set of keywords into the domain ontology and detecting the concepts that contain these keywords in the set.

Definition: the ontological concept c contains the word w, if $w \in \{c$ Y one of direct instance (c) Y one of synonym (c)}. Then an ontological concept c occurs in the document as long as a word contained by c occurs in this document.

So, the set of keywords is transformed to a set of concepts D_c from ontology that occurs in the document. When for a keyword no concept is detected in the ontology, the keyword converted as a new concept is added into the D_c. Because the new concept does not belong to the ontology, it has no relations with any other concepts of ontology.

3) Based on D_c of each document, inverted index by concepts is constructed. Using the concepts as the keys of DHTs, the inverted index is distributed among all nodes of system, and each node *p* is responsible for inverted lists for some concepts. In addition, for each document *d* in inverted list for concept *c*, the set of concepts D_c of *d* is also stored in the node *p*.

4.2 Query Processing

User issues a query request that is composed of keywords. Then the query is parsed with eliminating stop words and stemming using the same operation done to document. A set of concepts Q_c containing the query keywords is generated for a query. Using the concepts of Q_c as the keys of DHTs, the query is sent to the nodes that are responsible for the inverted list of concepts of Q_c. As described in above section, node *p* also stores the set of concepts D_c of document. So, the query is transformed to the match process of concepts between D_c and Q_c. The matched documents for the query can be retrieved locally in these responsible nodes without consulting other nodes. The intersection of inverted lists for different concepts between nodes is avoided, so the communication overhead is reduced immensely.

Definition: The document *d* matches the query *q*, if $\forall c_i, c_i \in Q_c$, $\exists c_j, c_j \in D_C$ of document satisfies

$$sim(c_i, c_j) \geq T$$

Where D_c and Q_c is the set of concepts for document *d* and query *q* respectively; *sim* represents the semantic similarity function; *T* is a constant that represents the similarity threshold.

For the semantic similarity function *sim*, [14] has compared different similarity measures and have proposed that for measuring the similarity between concepts in hierarchical structured semantic network, where semantic similarity considers to be determined by the shortest path length as well as the depth of the subsumer, so the following similarity measure (formula (1)) yields the best results.

$$sim(c_1, c_2) = \begin{cases} e^{-\alpha l} \cdot \dfrac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}} & \text{otherwise} \\ 1 & \text{if } c_1 \equiv c_2 \end{cases} \qquad (1)$$

Here c_1 and c_2 are concepts of ontology; *l* is the length of the shortest path between c_1 and c_2 in the graph spanned by the concept's relations. *h* is the level in the tree of the direct common subsumer from c_1 and c_2. α>0 and β>0 are parameters scaling the contribution of shortest path length *l* and depth *h*, respectively. In this paper, we use the formula (1) as the similarity calculation function, and set α=0.2 and β=0.6 as used in [14].

For the case of concepts c_1 and c_2 that are in the D_c and Q_c but not belonging to ontology, the semantic similarity function *sim* is:

$$sim(c_1, c_2) = \begin{cases} 1 & \text{if } c_1 \equiv c_2 \\ 0 & \text{otherwise} \end{cases} \qquad (2)$$

5. EXPERIMENTS

In this section, we evaluate SKS by simulation experiment. We constructed an ontology about computer science refer to the ACM topic hierarchy [15], which contains a set of 1287 topics in the computer science domain, and WordNet [16], where the terms are grouped into semantic equivalence sets. In order to analyze the SKS retrieval cost , we ran a web crawler that visited the web pages on the Yahoo news and download the HTML files recursively. Our crawler downloaded about 11,135 HTML pages. We develop a HTML parser using Java to clean HTML tags and extract plain text. We also develop a text parser using Java to eliminate the stop words and replace words by stems, adopting the algorithms introduced in [17].

We constructed inverted index files for the texts. Each index file contained some inverted lists, one for a concept occurred in texts. We measured number of entries for a text to analyze the overhead of index entry publishing for a document. We analyzed the communication overhead of query by doing a search in the inverted index files, extracting appropriated entries, and measuring the size of all entries. In order to compare SKS with existing keyword search approaches based on standard inverted index scheme--inverted index by keywords, we also construct inverted index files based on keywords for these same texts. We evaluate them by overhead of index entry publishing and query overhead.

Index entry publishing means storing the index entries for a document to appropriate nodes in P2P system. The overhead of network is the number of bytes transmitted when index entries storing in the nodes. The number of bytes transmitted can be obtained through multiplying number of entries generated for a document by the size of each entry.

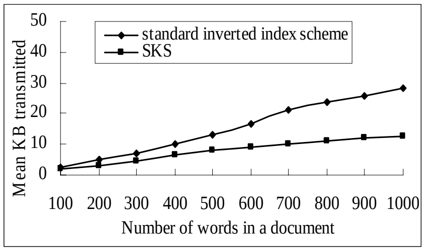


Figure 3. Overhead of index entries publishing

Figure 3 gives the mean KB transmitted when a document is shared using standard inverted index scheme, and SKS with the similarity threshold of 1. Figure 3 shows overhead of SKS is much lower than that of standard inverted index scheme.

Query overhead is the number of bytes transmitted when a user issues a keyword search. We multiplied the size of each entry that matched the query by the number of entries to obtain the number of bytes that would be transmitted across the network during the search.

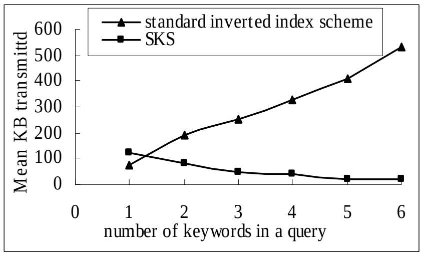


Figure 4. Query overhead

Figure 4 gives the mean KB transmitted when a user issued a keyword search with standard inverted index scheme, and SKS with the similarity threshold of 1. Figure 4 shows the query overhead of SKS is much lower than that of standard inverted index scheme.

In this paper, we use the similarity function of concepts to determine the match of document and query. So, the similarity threshold *T* will impact the query overhead. We evaluate the query overhead when the similarity threshold *T* changes.

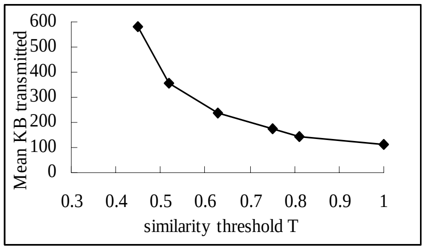


Figure 5. Query overhead when T changes

Figure 5 shows mean KB transmitted when similarity threshold *T* changes. With the value of *T* increasing, the scope of concepts match in the ontology becomes small. So, the query overhead is decreased.

6. CONCLUSIONS

In this paper, we have presented a scalable keyword search mechanism based on semantic, SKS, in DHTs based Peer-to-Peer system. SKS exploits ontology to capture the semantic relations among words and generate a set of concepts respectively for document and query. SKS constructs the distributed inverted index by concepts instead of keywords. So, the number of index entries publishing for documents is decreased. The concept set representing of document is added in the concept index, which avoids the inverted list transmitting among nodes. As a result, the total overhead of network is lowered. With the concept index SKS also implements semantic search based on keywords.

Our future work includes studying the index update generating from the change of global statistics for concept, and improving the retrieval performance through using ontology to do nature language disambiguation. Some optimizations, such as caching, compression and so on, can be combined with SKS to improve search efficiency.

7. ACKHOWLEDGEMENTS

This paper is sponsored by NSF of China (No.90612001), Science and Technology Development Plan of Tianjin , (No. 043800311, 043185111-14) and Nankai University Innovation Fund and ISC.

8. REFERENCES

[1] The Napster Homepage.: http://www.napster.com.

[2] The Gnutella Homepage.: http://gnutella.wego.com.

[3] I.Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for Internet applications. In Proc. of the ACM SIGCOMM ’01, 2001.

[4] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In ACM SIGCOMM, 2001.

[5] Jinyang Li, Boon Thau Loo, Joseph M.Hellerstein, M.Frans Kaashoek, David R.Karger, and Robert Morris. On the Feasibility of Peer-to-Peer Web Indexing and Search. In IPTPS, 2003.

[6] Omprakash, D. Gnawali. A Keyword-set Search System for Peer-to-Peer Networks. MIT’s thesis Lib, 2002.

[7] Toan Luu, Fabius Klemm, Ivana Podnar, Martin Rajman, Karl Aberer. ALVIS Peers: A Scalable Full text Peer-to-Peer Retrieval Engine. In P2PIR’06, Arlington, Virginia, USA 2006.

[8] L. Liu and K. D. Ryu. Supporting Efficient Keyword Based File Search in Peer-to-Peer File Sharing Systems. IBM Research IBM Research Report. RC23145 (W0403-068), 2004.

[9] C. Tang and S. Dwarkadas. Hybrid global-local indexing for efficient peer-to-peer information retrieval. In NSDI, 2004.

[10] Shuming Shi, Guangwen Yang, Dingxing Wang, JinYu, Shaogang Qu, and Ming Chen. Making Peer-to-Peer Keyword Searching Feasible Using Multi-Level Partitioning. In IPTPS, 2004.

[11] Nicholas J. A. Harvey, Michael B. Jones, Stefan Saroiu, Marvin Theimer and Alec Wolman. SkipNet: A Scalable Overlay Network with Practical Locality Properties. USITS'03, 2003.

[12] Gruber T R. A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition, 1993,5: 199～220.

[13] Sean Bechhofer, Frank van Harmelen, Jim Hendler, et al, OWL Web Ontology Language Reference, http://www.w3.org/TR/2004/REC-owl-ref-20040210/, February 2004.

[14] Yuhua L, Bandar ZA, McLean D. An approach for measuring semantic similarity between words using multiple information sources. IEEE Trans. on Knowledge and Data Engineering, 2003,15(4): 871-882.

[15] The ACM Topic Hierarchy. http://www.acm.org/class/1998/.

[16] G. Miller, “WordNet: A Lexical Database for English”, in Proc. of Communications of CACM, Nov 1995.

[17] W.B. Frankes and R. Baeza-Yates. Information Retrieval: Data Structure and Algorithm. Prentice Hall, Englewood Cliffs, NJ, USA, 1992.