Concept Index for Document Retrieval with Peer-to-Peer Network

wenhui Ma, wenbin Fang, gang Wang, jing Liu
College of Information Science and Technology
University of Nankai
Tianjin 300071, China
wenhuima_nk@hotmail.com

Abstract

Traditionally, full text retrieval over structure peer-to-peer network has been implemented by inverted index by keywords. However, search based on this index scheme only support literally word match, not taking into account the meaning of words. In this paper, we present a new index scheme, inverted index by concepts, for document retrieval over structure P2P network. We introduce ontology to capture the semantic relations between words, which provide a solution for ambiguity and richness of natural language. The index scheme proposed in this paper transforms the keyword search to the match process of concepts, and implementing keyword search based on semantic. Our index scheme also avoids the intersection of inverted lists between nodes when executing multi-keyword search. Simulation experiment shows that search based on our index scheme has higher retrieval performance than that based on keyword index, and generates lower network traffic.

1. Introduction

Recently, the Peer-to-Peer (P2P) systems have gained tremendous interest for files sharing over Internet. Although P2P file sharing provides a scalable alternative to conventional central server based approaches, providing efficient file search in a large-scale P2P system remains challenging problem. The common way for keyword search over structure P2P networks, such as Chord [1] and CAN [2] is constructing distributed inverted index by keywords as they actually implement Distributed Hash Table (DHT) over them. This approach is adopted by some proposals [3] [4] to implement the full-text searching functionality to the P2P network. However, search based on inverted index by keywords over structure P2P networks only literally matches words in documents with those present in a user's query, which can lead to poor retrieval performance due to two facts in natural language. First, many words have multiple meanings, so many unrelated documents may be retrieved just because they matched some of the query keywords. Second, because the same concept can be described by multiple words, relevant documents that do not contain any of the query keywords will not be retrieved

In this paper, we present a new index scheme, distributed inverted index by concepts. We introduce ontology to capture the semantic relations between words and describe each document and query as a set of concepts of ontology respectively. Thereby, we construct concept indices and distribute them among nodes of DHT based P2P networks for documents retrieval. Based on concept index, keyword search is transformed to match process of concepts that occurred in documents and query, which implements the semantic search. Also concept index avoids the intersection of invert lists among nodes and lowers the network traffic.

The remainder of this paper is organized as follows. Related work is discussed in Section 2. Section 3 describes the background of our work, and distributed inverted index construction and document retrieval are discussed in section 4 and Section 5 respectively. Section 6 gives the simulation experimental results. Section 7 concludes the paper.

2. Related work

Most of the existing works of information retrieval over structure P2P networks use the approaches of inverted index by keywords.[3] [4] [5] However, keyword index does not take into account of the meaning of index terms, so it alone can only supports simple retrieval tasks and will be hard to support sophisticated retrieval algorithms. In [6], Tang proposed PeerSearch, which introduces the Latent Semantic Index (LSI) to capture the semantic relations

between terms. PeerSearch stores a document index in CAN using its vector representation as the coordinates, resulting in that indices stored close to each other are also close in semantics. This unifies the problem of semantic-based search with routing in an overlay network. [7] has proposed a two-phase distributed semantic index on top of DHT. It classified all document corpus in to clusters, and each of them correspond to a concept. There is a unique concept vector produced for each document. Based on the clusters, all documents are mapped on to the DHT based on P2P system, so a distributed semantic cluster is constructed that lead to good semantic locality on index placement so that the indices of similar documents are placed together or near each other. The query can be matched in the level of semantic.

3. Background

In this paper, the inverted index by concepts mechanism is constructed on top of DHT, of which we first give a brief introduction. We will also briefly describe the ontology used in our work.

3.1. Distributed Hash Table(DHT)

DHT is developed to improve the performance of data discovery in P2P systems, such as [1] [2]. It imposes constraints both on the node graph and on data placement to enable efficient discovery of data. Each data item is identified by a key, and nodes are organized into a structured graph, each of them has a unique identifier. The DHT systems provide two basic interfaces:

Insertion: For the interface of Insertion, the function of *put(key, object)* causes the DHT to route the given *object* to the node with a identifier closest to the *key*.

Retrieval: For the interface of Retrieval, the function of *get(key)* is provided. *object=get(key)*, causes the DHT to obtain the *object* from the node with a node identifier closest to the *key*.

The DHT systems provide efficient support for exact match queries. However, it is the property of exact match that constrains the full text search in DHT based P2P systems. Only the documents that are indexed by query keywords can be retrieved, which dose not take into account the semantic relations between words completely.

3.2. Ontology

In computer science, an ontology is a hierarchically structure set of terms to describe a domain that can be used as a skeletal foundation for a knowledge base [8]. Ontology provides terms that represents the knowledge and relations among these terms in target domain. In the following we do some definitions for the ontology used in this paper.

Definition: ontology O is a hierarchy knowledge description model as follows

$$O = \{C, R, I\}$$

Where C is a set of concepts, R is a set of relations and I is a set of instances.

Here interprets the elements of ontology in detail:

(1) concept: it is an abstract and conceptualising representation for knowledge of domain. Each concept has a unique label name in ontology, and is a set of synonyms.

(2) instance: it is specific representation for domain knowledge. Instance denotes itself and describes the domain knowledge with words. One concept can have several instances.

(3) relation: it denotes the interconnection between concepts, concepts and instances, such as “is-a”, “instance-of” and “part-of”.

With the concepts and relations, the ontology can effectively solve the problem faced in natural language where words have multiple meanings and one meaning is represented by multiple words.

4. Distributed inverted index by concepts

The approach of keyword index considers that each document’s content can be described by a bag of keywords. Thus the set of keywords is used to index document. This approach currently is used in DHT based P2P systems to construct distributed inverted index by keywords. But document indexing and retrieving through distributed keyword index is usually assumed that the index terms are mutually independent, not consider the semantic relations among them. So, it can usually result in the retrieval performance lowering because of the ambiguity and richness of natural language. In this paper, we introduce ontology to capture semantic relations between words for target domain, and generate a set of concepts to represent each document and a query respectively. Thereby we construct distributed inverted index by concepts for indexing and retrieving documents.

We generate a set of keywords D 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.

$$D = \{w_1, w_2 \ldots w_m\}$$

Here w_m is the keyword that occurs in the document. We map D on to the ontology and detect the concepts that contain the words of D.

Definition: the ontological concept c contains the word w, if $w \equiv c \bigcup w \equiv$ one of direct instance (c) $\bigcup w \equiv$ 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, a set of ontological concepts D_c is achieved that contain all words of D. When for a word no concept is detected in the ontology, the word is converted as a new concept and added into the D_c, which is used to match the query when query processing. Because the new concept does not belong to the ontology, it has no relation with any other concepts of ontology. On the contrary, a word w_m, as an instance of concept or one of synonyms of concept, maybe belong to several concepts. This is caused by the ambiguity and richness of natural language. If this is the case, we select the most appropriate concept that contains w_m through the context of this concept. Let $Context_c$ is the context of the concept c.

Definition: Context of concept c is a set of words

Context_c = { synonym(c) $\bigcup$ instance(c) $\bigcup$ synonym (superclass(c)) $\bigcup$ instance(superclass(c)) $\bigcup$ synonym (subclass (c)) $\bigcup$ instance(subclass(c)) }

For the word w_m that belongs to several ontological concepts, we calculate a mapping score $S(w_m,c)$, which indicates how good w_m is mapped on to the concept c.

$$S(w_k, c) = \left| Context_c \bigcap D \right|$$

We count the number of words that occur in both context of concept c and the set of keywords D of the document which the term w_m belong to, and select the concept c_i that has the maximum of $S(w_m,c_i)$. So, the set of concepts D_c is denoted as follows:

$$D_c = \{c_1, c_2, ..., c_k\}$$

Here c_i $(i=1,2,...k)$denotes the ontological concept that has the maximum of mapping score.

A weight associated with each concept of D_c is calculated. This weight quantifies the important of the index concept for describing the document semantic contents. For the calculation of concept weight, we extends the *tf*idf* scheme [9], which is usually used to compute the term weight in the Information Retrieval, and proposes the *cf*idf* scheme. *cf* represents the appearance frequency of concept in a document, and *idf*, which is invert document frequency of concept, represents how often the concept occurs in other documents in the system.

Let $freq_{ij}$ be the raw appearance frequency of keyword t_i that belongs to concept c in document d_j. Then we can calculate the $freq_{cj}$, the appearance frequency of c appeared in document d_j as follows (formula (1)):

$$freq_{cj} = \sum_{i=1}^{n} freq_{ij} \qquad (1)$$

Here, n is the number of keywords of document d_j that belong to concept c. So, the normalised frequency cf of c is (formula (2)):

$$cf = \frac{freq_{cj}}{\max(freq_{cj})} \qquad (2)$$

Where the maximum of frequency is calculated over all concepts which are appeared in document d_j. Let N is the total number of documents and n_i is the number of documents which the concept c appears in. The *idf* for c is given following (formula (3)):

$$idf = \log(\frac{N}{n_i}) \qquad (3)$$

So, we may calculate the $w(c)$, the weight of c, using *cf*idf*, and it is given by formula (4):

$$w(c) = cf * idf \qquad (4)$$

Using each concept of D_c as the key of DHT, a distributed inverted index by concepts is constructed, and each node is responsible for inverted lists of 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, which can make the query matched locally

5. Document retrieval

User issues a query that is composed of multiples keywords. 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 query:

$$Q_c = \{c_1, c_2, ..., c_l\}$$

Here c_i $(i=1,2...l)$ denotes the concept. Using the concepts of Q_c as the keys of DHT, the query is sent to the nodes that are responsible for the inverted lists of these concepts. As described in above section, the node also stores the concept set D_c of document. So, the query is transformed to the matching process of concepts between D_c and Q_c. The final matched documents for the query can be achieved locally in these nodes without consulting other nodes. The intersection of inverted lists for different concepts between nodes is avoided, so the network traffic is reduced immensely.

In this paper, we define a match function to retrieve documents for a query according to the sum of the maximum similar value of each concept in the set of concepts of query, Q_c, to all concepts in the set of concepts of document, D_c. Let $R(d, q)$ denote the match function between document d and query q, it is given by formula (5)

$$R(d,q)=\sum_{i=1}^{l}\max\{sim(c_i,c_j)\cdot w(c_j)\mid j=1,2,...,k\} \qquad (5)$$

Here $c_i \in Q_c$, $c_j \in D_c$; $l=\left|Q_c\right|$ and $k=\left|D_c\right|$; $w(c_j)$ represents the weight of concept c_j; *sim* represents the semantic similarity function of concepts. So, we set different threshold values of match for function *R (d, q)* to retrieve documents

For the semantic similarity function *sim*, [10] 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 (6)) yields the best results.

$$sim(c_1,c_2)=\begin{cases} e^{-\alpha l}\cdot\dfrac{e^{\beta h}-e^{-\beta h}}{e^{\beta h}+e^{-\beta h}} & \text{otherwise} \\ 1 & \text{if } c_1\equiv c_2 \end{cases} \qquad (6)$$

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 relation. h is the level in the tree of the direct common subsumer from c_1 and c_2. $\alpha>0$ and $\beta>0$ are parameters scaling the contribution of shortest path length l and depth h, respectively. In this paper, we use the formula (6) as the similarity calculation function, and set α=0.2 and β=0.6 that is used in [10].

6. Simulation experiments

In this section, we evaluate our concept index mechanism by simulation experiment. We use the RDF/OWL representation of WordNet[11] as our general ontology. WordNet[12] is a manually constructed lexical system, which is organized conceptually. WordNet's basic object is a set of synonyms, called a synset. A synset denotes a concept that describes the knowledge for specific domain. Nouns in WordNet are organized in a hierarchical tree structure based on hypernym/hyponymy. The hyponym of a noun is its subordinate, and the relation between a hyponym and its hypernym is an "is-a" relation. Hypernym and its inverse, hyponym, are transitive semantic relations between synsets. In this paper, we only use the nouns of RDF/OWL representation of WordNet and the relations of hyponym and hypernym between nouns, which are all indicated by RDF. We use Jena [13] to access the RDF representation of WordNet. Jena is a Java implementation for basic RDF handling. It aims at standard compliance and a friendly access from Java.

In order to analyse the retrieval performance of concept index proposed in this paper, we ran a web crawler that visited the web pages on the Yahoo news and download the HTML files recursively. Our crawler downloaded about 13,135 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 [14].

We constructed inverted index files for the texts. Each index file contained some inverted lists, one for a concept occurred in texts. In order to compare concept index with existing keyword index scheme, we also construct additional index files for these same texts. Each index file contained some inverted lists, one for a word occurred in texts. We simulated search among peers of structure P2P network by searching over the index files. So, we analyzed retrieval performance and network traffic for a query by doing a search with keyword match in keyword index files and with concept match in concept index files respectively.

The query performance is evaluated using the standard information retrieval measures, precision and recall. Figure 1 shows the change of precision and recall of query using keyword index and concept index respectively. As shown in Figure 1, compared with commonly keyword index, concept index can significantly improve the precision and recall of query.

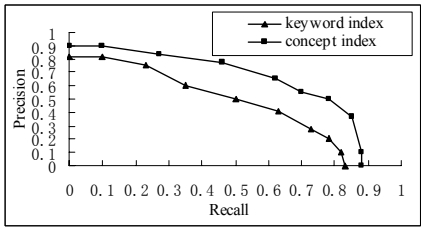

Figure.1. Precision-recall graph for query

Network traffic of a query is the number of bytes transmitted when a user issues a query. The traffic of transmitting the intermediate result lists from one node to another is the main part of query overhead.

Figure 2 gives the mean KB transmitted when a user executed a query using keyword match and concept match respectively. Figure 2 shows the network traffic for a query with concept match is much lower than that with keyword match.

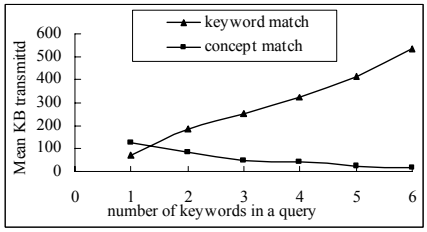

Figure 2. Network traffic for a query

7. Conclusions

In this paper, we describe a distributed inverted index by concepts. We exploit ontology to organize the target knowledge domain and capture the semantic relation between words. Our index scheme transforms the keyword search to the match process of concepts of documents and query, which effectively addresses ambiguity and richness of natural language and implements semantic search. Moreover, our concept index scheme reduces the network traffic when executing multi-keywords search. Simulation experiment shows that comparing to keyword index search based on inverted index by concepts can improve the retrieval performance immensely and generate lower network traffic.

8. Acknowledgements

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.

9. References

[1] 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.

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

[3] Reynolds, P., Vahdat, A. Efficient peer-to-peer keyword searching. Technical Report 2002, Duke University, CS Department, Feb. 2002.

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

[5] 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.

[6] C.Tang, Z. Xu and M. Mahalingam. PeerSearch:Efficient Information Retrieval in Peer-to-Peer Networks[J].HPL- 2002-198, 2002

[7] Yan Chen, Zhichen Xu, Chengxiang Zhai . A scalable Semantic Indexing Framework for Peer-to-Peer Information Retrieval. SIGIR 2005 workshop: Heterogeneous and Distributed Information Retrieval, August 19th, 2005.

[8] Swartout B., Patil R., Knight K., Russ T. Toward distributed use of large-scale ontologies. Proceedings of the Tenth Knowledge Acquisition for Knowledge-Based Systems Workshop. (KAW '96), Alberta, Canada, November 1996. (71).

[9] M. Berry, Z. Drmac, and E. Jessup. Matrices, Vector Spaces, and Information Retrieval. SIAM Review, 41(2): 335–362, 1999

[10] 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.

[11] Mark van Assem, Aldo Gangemi, Guus Schreiber. RDF/OWL Representation of WordNet. http://www.w3.org/TR/2006/WD-wordnet-rdf-20060619/.

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

[13] Brian McBride. Jena: Implementing the rdf model and syntax specification. 2001.

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