SIPPER: Selecting Informative Peers in Structured P2P Environment for Content-Based Retrieval *

Shuigeng Zhou Zheng Zhang Weining Qian Aoying Zhou
Department of Computer Science and Engineering, Fudan University, Shanghai 200433, China
{sgzhou, zhzhang81, wnqian, ayzhou}@fudan.edu.cn

1. Introduction

In this demonstration, we present a prototype system called **SIPPER**, which is the abbreviation for **S**electing **I**nformative **P**eers in Structured **P**2P **E**nvironment for Content-based **R**etrieval. SIPPER distinguishes itself from the existing P2P-IR systems by the following two features: First, to improve retrieval efficiency, SIPPER employs a novel peer selection method to direct the query to a small fraction of relevant peers in the network for searching globally relevant documents. Second, to reduce the bandwidth cost of meta data publishing, SIPPER uses a new publishing mechanism, the *term-node* publishing mechanism, which is different from the traditional *term-document* model [2].

2 System Overview

Figure 1 illustrates the architecture of SIPPER, in which Figure 1(a) and Figure 1(b) show SIPPER network structure and the internals of a SIPPER node respectively. In SIPPER system, a large number of computers are organized by DHT into a Chord [1] network. Each joining computer is a peer containing its own documents collection. For each SIPPER node, it contains eight major components: User Interface (UI), Peer Communication Manager (PCM), Query Processor (QP), Peer Selector (PS), Statistic Information Publisher (SIP), Local Documents Base (LD), Statistic Information Base (SI), and Database Engine (DBE). Among them, QP, PS and SIP are the three core components. UI is the interaction interface between the users and the peer. PCM is responsible for data and messages exchange with other peers, it embodies the Chord protocol. QP is responsible for query processing: receiving queries from users, retrieving relevant documents from the selected peers, merging the partial results from different peers, and returning the final results to the users. PS selects a small number of informative peers for each query. In SIPPER, the number of selected peers is an automatically tunable parameter, which is determined by the query and the network. SIP preprocesses the local documents, and publishes local statistic information to other peers according to the proposed method. DBE is not different from its counterpart in other database applications. LD manages the local documents, and SI keeps the statistic information published by other peers.

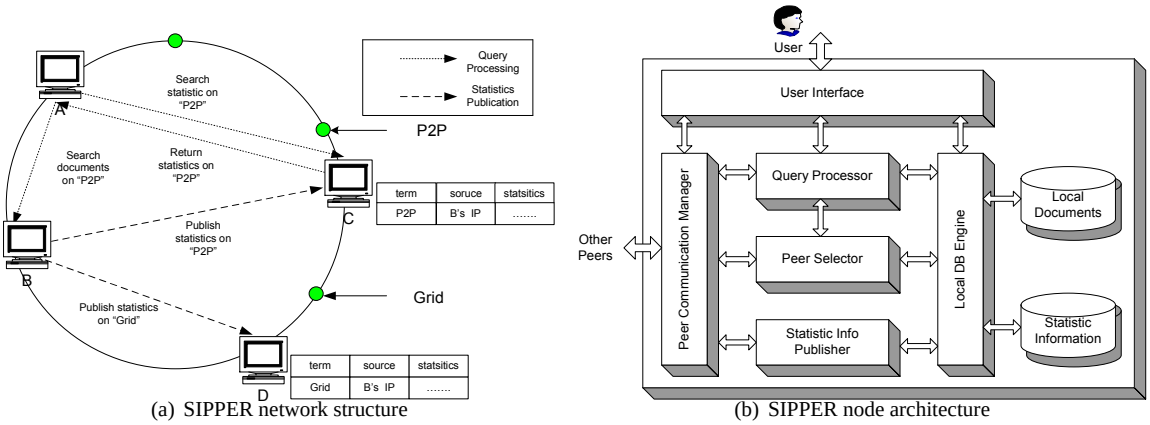
*This work was partially supported National Natural Science Foundation of China (NSFC) under grant no. 60373019 and no. 60496325, and the Shuguang Progam of Shanghai Municipal Education Committee.

2.1 Statistic Information Publisher

We use VSM to represent document and the TF*IDF method to calculate document vector. The weight of term t_i in document d_k is $w_{ik} = tf_{ik} \times idf_i$, and $tf_{ik} = \frac{freq_{ik}}{max\_freq_k}$, $idf_i = log\frac{N_d}{n_i}$. Here, $freq_{ik}$ is the occurrence count of t_i in d_k, max_freq_k is the maximum occurrence count of all terms in d_k, N_d is the total number of documents in the collection and n_i is the number of documents including t_i. We employ a *term-node* mechanism to publish statistic information of each peer, i.e., for *each term in a peer*, its statistic information is published one time. For term t_i in peer P_j, its published information constitutes a tuple $T_{ij} = (I_j, t_i, sum\_tf_{ij}, max\_tf_{ij}, n_{ij})$. Here, I_j is the ID of P_j, n_{ij} is the number of documents in P_j containing t_i, $sum\_tf_{ij} = \sum_{d_k \in P_j} tf_{ik}$, and $max\_tf_{ij} = max\{tf_{ik} | d_k \in P_j\}$. T_{ij} is mapped to the peer with the largest ID that is not greater than the hashing value of t_i. P_j also publishes the number of total documents it has. A fixed stop-word (e.g. "the") is selected to map the numbers of documents all peers have to a corresponding peer.

2.2 Peer Selector

A method to estimate the *goodness* of peers for a given query is developed. Let $G(q, P_j)$ be the estimated goodness of peer P_j for query $q = \{q_1, \ldots, q_l\}$, its value consists of two parts: $g_1(q, P_j)$ and $g_2(q, P_j)$. $g_1(q, P_j) = \sum_{d_k \in P_j} sim(q, d_k) = \sum_{i=1,\ldots,l} q_i \times sum\_tf_{ij} \times idf_i$. $g_2(q, P_j)$ is different from $g_1(q, P_j)$ by replacing sum_tf_{ij}



(a) SIPPER network structure (b) SIPPER node architecture

Figure 1. SIPPER architecture

with max_tf_{ij}, i.e., $g_2(q, P_j) = \sum_{i=1,\ldots,l} q_i \times max\_tf_{ij} \times idf_i$. $G(q, P_j)$ is a linear combination of $g_1(q, P_j)$ and $g_2(q, P_j)$, that is,

$$G(q, P_j) = \alpha \times g_1(q, P_j) + \beta \times g_2(q, P_j). \qquad (1)$$

We take $\alpha = 0.5 \times max\{g_1(q, P)\}$ and $\beta = 0.5 \times max\{g_2(q, P)\}$ such that the value of $G(q, P)$ is between 0 and 1. After the goodness values are calculated, the peers are ranked by the values, and the selected peers are those with the highest goodness values.

2.3 Query Processor

When peer P_j receives a query q, it will process the query as follows:

1. for each term $q_l (1 \leq l \leq m)$ in q, retrieving its statistic information by conducting *lookup*(q_l). *lookup* is an operator provided by Chord to locate an object specified by the input parameter. And getting the total number of documents in the system by conducting *lookup*("the").

2. with the collected statistic information of the query terms $\{q_1, \ldots, q_m\}$, a rank list of peers relevant to the query is calculated by the peers selection method proposed in last subsection. Suppose L (usually $L > k$) peers are selected, and denoted as $\{SP_1, \ldots, SP_L\}$.

3. forwarding the query q to each of the selected peers $\{SP_1, \ldots, SP_L\}$. At each selected peer, similarity-based searching is carried out over the local text collection, the *local* top-k documents can be obtained. And the similarity between the local top-1 document and the query q, denoted as $local\_max\_Sim$ is returned to the query peer P_j.

4. After collecting the $local\_max\_Sim$ values from the L selected peers, ranking the $local\_max\_Sim$ values, and the peers corresponding to the top-k $local\_max\_Sim$ values are selected for further searching. That is, the final top-k documents should locate in the k peers with highest $local\_max\_Sim$ values. This argument can be proved, but due to space limit, we omit the proof.

5. the top-k peers with highest $local\_max\_Sim$ values returning their local top-k documents to the query peer.

6. the query peer collects the local top-k documents from the k selected peers in step 4, and merges these local top-k documents to obtain the final top-k documents.

3 Demonstration Outline

An environment with no fewer than four nodes (laptops) will be established to illustrate SIPPER's features in the following ways: 1) We will first show how to do content-based retrieval on SIPPER, this includes setting directories and documents for sharing, issuing queries and browsing returned documents. 2) We will then illustrate the retrieval process of SIPPER, including statistic information publishing, peers selection and query processing, feel how fast SIPPER responds to user queries, and check the quality of the returned results. 3) We will show how SIPPER responds to a dynamic environment: peers joining and leaving, adding or reducing documents on peers.

References

[1] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A scalabel peer-to-peer lookup service for internet applicaitons. In *Proceedings of SIGCOMM'01*, 2001.

[2] C. Tang and S. Dwarkadas. Hybrid global-local indexing for effcient peer-to-peer information retrieval. In *Proceedings of NSDI'04*, 2004.