P2P Content-Based Query Routing Using Firework Query Model

Ka-Cheung Sia, Cheuk-Hang Ng, Chi-Hang Chan, and Irwin King
Department of Computer Science and Engineering
The Chinese University of Hong Kong
{kcsia, chng, chchan, king}@cse.cuhk.edu.hk

Abstract

Clustering technique is used in database and information retrieval system for organizing data and improving retrieval efficiency. We surmise such functionality is valuable to a Peer-to-Peer (P2P) distributed environment. In this paper, we introduce the concept of peer clustering at the level of overlaying network topology, and content-based query routing strategy to improve existing retrieval methods. We design and implement a DIStributed COntent-based Visual Information Retrieval (DISCOVIR) system with content-based query functionality and improved query efficiency. We demonstrate its scalability and efficiency through simulation.

1 Introduction

The appearance of Peer-to-Peer (P2P) applications such as Gnutella [5] and Napster [8] have demonstrated the significance of distributed information sharing systems. These models offer advantages of decentralization by distributing the storage, information and computation cost among the peers. Because of these desirable qualities, many research projects have been focused on designing different P2P systems and improving their performance.

Compared with previous works, our contribution to P2P network are 1) users can perform query based on content of information rather than simple filename or metadata, 2) efficient location of data under an environment with no index storage in centralized server or distributed among peers.

In this paper, we propose a strategy for clustering peers that share similar properties together, thus, data inside the P2P network will be organized in a fashion similar to a Yellow Pages. In order to make use of our clustered P2P network efficiently, we also propose a new content-based query routing strategy, the Firework Query Model (FQM) [9], which aims to route the query intelligently according to the content of query to reduce the network traffic of query passing in the network. In particular, we design and implement a DIStributed COntent-based Visual Information Retrieval system (DISCOVIR) [4], which is compatible to Gnutella network, for users to share and retrieve images.

2 Background

Both Napster and Gnutella have demonstrated the possibility of distributing storage over computers in the Internet. Such kind of P2P network offers the advantages of resource utilization [12], increased reliability [3] and comprehensiveness of information [7]. Besides, they also reveal the limited scalability for two reasons. One is the bottle-neck at the centralized server storing the index, like Napster. The other one is the flooding of query messages when data location process is decentralized, like Gnutella. To address the data location problem, Chord [13] and CAN [10] tackle it by distributing the index storage into different peers, thus sharing the workload of a centralized index server.

Distributed infrastructure of both CAN and Chord use Distributed Hash Table (DHT) to map the filename to a key, and each peer is responsible for storing certain range of (key, value) pairs. When a peer looks for a file, it hashes the filename to a key and ask the peers responsible for this key for the actual storage location of that file. Chord models the key as an m-bits identifier and arranges the peers into a logical ring topology to determine which peer is responsible for storing which (key, value) pair. CAN models the key as point on a d-dimension Cartesian coordinate space, while each peer is responsible for (key, value) pairs inside its specific region. Such systems take a balance between the centralized index and totally decentralized index approaches. They speed up and reduce message passing for the process of key lookup (data location); however, they incur a penalty for redistributing index storage when peers join and leave the network frequently, especially in a dynamic environment like the Internet. Moreover, such kind of schemes rely on the trustworthy of peers participating in the network. The problem is serious if malicious peers deny to respond to queries which it is assumed to be responsible for under the condition of no duplicate index storage in other peers.

Current researches and on-going developing systems [13, 10, 11, 15, 6, 2] focus on the requirement of efficient insertion and retrieval of content in a distributed storage infrastructure. Filenames or meta-data, such as ID3 tag of MP3, are both used as queries and indexing terms for data. Although DHT based methods are extensible from exact match to textual similarity matches of filenames by breaking into "n-grams"[14], penalty on system performance is not discussed in detail. Besides, manual operation of summarizing data to filename or meta-data is a tedious job, what if we want to perform more complex search, other than filename matching? Suppose I want to find an image similar to this one, I want to find a document with similar content to this one, etc. Such modern information retrieval tasks are addressed in the notion of client-server approach [1], what if we perform this in a P2P information system?

Based on the problems raised above, we ask: are we able to formulate a P2P model that optimizes for the data location process, while introducing comparatively less penalty when peers join and leave the network? Are we able to perform more complex content-based searching in this P2P network? Instead of distributing the storage of index into different peers, we allow a peer to index its own data collection while retaining the original Gnutella network topology but imposing a requirement on connections to serve as the global indexing structure. Moreover, we introduce the content-based image search functionality to illustrate the potential richness of queries in a P2P network.

3 Peer Clustering and Firework Query Model

Our design goal for DISCOVIR is to improve data lookup efficiency in a completely distributed P2P network, while keeping a simple network topology and number of message passing to a minimum. The DISCOVIR is built compatible to the Gnutella network. There are two types of connections in DISCOVIR, namely ***random*** and ***attractive***. Attractive connections are to link peers sharing similar data together. We perform peer clustering at the level of overlaying network topology instead of locally shared data, thus content-based query routing is realizable to improve query efficiency. As a result, it manages to be scalable when network grows. We have implemented a prototype version of DISCOVIR, built on top of LimeWire [7] with content-based image searching capability.

3.1 Peer Clustering

With the inherent nature of DISCOVIR network, we apply notation in graph theory to model it (see Table. 1). For the sake of generality, we try to keep this in high level of abstraction. In the actual realization, we choose the vector model in information retrieval literature as the underlying data structure for representing data. Here are some definitions:

Table 1: Definition of Terms

$G\{V, E\}$	The P2P network, with V denoting the set of peers and E denoting the set of connection
$E = \{E_r, E_a\}$	The set of connections, composed of random connections, E_r and attractive connections, E_a.
$e_a = (v, w, sig_v, sig_w)$, $v, w \in V, e_a \in E_a$	The attractive connection between peers v, w based on sig_v and sig_w
$Horizon(v, t) \subseteq V$	Set of peers reachable from v within t hops
$SIG_v, v \in V$	Set of signature values characterizing the data shared by peer v
$D(sig_v, sig_v)$, $v, w \in V$	Distance measure between specific signature values of two peers v and w.
$D_q(sig_v, q)$ $sig_v \in SIG_v$	Distance measure between a query q and peer v based on sig_v.
$C = \{C_v : v \in V\}$	The collection of data shared in the DISCOVIR network.
C_v	The collection of data shared by peer v, which is a subset of C.
$REL(c_v, q)$, $c_v \in C_v$	A function determining relevance of data c_v to a query q. 1-relevant, 0-non-relevant

Definition 1 *We consider information shared by a peer can be represented in a multi-dimension point based on its content, and the similarity among files is based on the distance measure between data points. Consider*

$$f \;:\; c_v \to \vec{c_v} \quad (1)$$
$$f \;:\; q \to \vec{q} \quad (2)$$

f *is the mapping function from file* c_v *to a vector* $\vec{c_v}$. *In the notion of image processing,* c_v *is the raw image data,* f *is a specific feature extraction method,* $\vec{c_v}$ *is the extracted feature vector characterizing the image. Likewise,* f *is also used to map a query* q *to a query vector* $\vec{q}$*, to be sent out when user makes a query.*

Definition 2 SIG_v *is the set of signature values representing data characteristic of peer* v*, with each* sig_v *representing each specific cluster of data. We define*

$$sig_v = (\vec{\mu}, \vec{\delta}), \quad (3)$$

where $\vec{\mu}$ and $\vec{\delta}$ are the statistical mean and standard deviation of the collection of data belonging to a subcluster, $C_v^{'}, C_v^{'} \subseteq C_v$. From now on, sig_v characterizes certain portion of data shared by peer p.

Definition 3 *$D(sig_v, sig_w)$ is defined as the distance measure between sig_v and sig_w, in other sense, the similarity between particular sub-cluster belonging to two different peers v and w. It is defined as,*

$$D(sig_v, sig_w) = ||\vec{\mu_v} - \vec{\mu_w}||. \tag{4}$$

$||\vec{\mu_v} - \vec{\mu_w}||$ is the Euclidean distance between centroid of two sub-cluster symbolized by sig_v, sig_w. With this formula, we define the data affinity of two peers, we will later use this to help organizing the network.

Based on the above definitions, we introduce a peer clustering algorithm, to be used in the network setup stage, in order to help building the DISCOVIR as a self-organized network oriented in content affinity. It consists of three steps:

1. **Signature Value Calculation**–Every peer preprocess its data collection and calculates a set of signature values SIG_v to characterize its data properties. Whenever the shared data collection, C_v, of a peer changes, the signature value should be updated accordingly. The whole data collection of the peer will be divided into sub-clusters automatically by a clustering algorithm, e.g. k-means, competitive learning, and expectation maximization. The number of signature values is variable and is a trade-off between data characteristic resolution and computational cost.

2. **Neighborhood Discovery**–After a peer joins the DISCOVIR network by connecting to a random peer in the network, it broadcasts a signature query message, similar to that of ping-pong messages in Gnutella, to reveal the location and data characteristic of its neighborhood, $Horizon(v, t)$. This task is not only done when a peer first joins the network, it repeats every certain interval in order to maintain the latest information of other peers.

3. **Attractive Connection Establishment**–By acquiring the signature values of other peers, one can reveal the peer with highest data affinity (similarity) to itself, and make an attractive connection to link them up. When an existing attractive connection breaks, a peer should check its host cache, which contains signature values of other peers found in the neighborhood discovery stage, and reestablish the attractive connection using peer clustering algorithm again.

Having all peers joining the DISCOVIR network perform the three tasks described above, you can envision a P2P network with self-organizing ability to be constructed. Peers sharing similar content will be grouped together like a Yellow Pages. Based on this content similarity based clustering, we will delineate a more complex query strategy in the next section. The detail steps of peer clustering is illustrated in Algorithm 1, and Fig. 1 depicts the peer clustering.

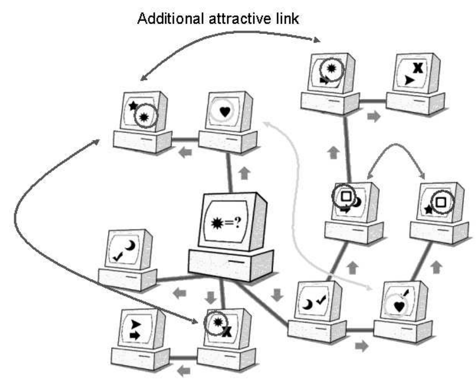

Figure 1: Illustration of peer clustering.

Algorithm 1 Algorithm for peer clustering

Peer-Clustering(peer v, integer ttl)
for all $sig_v \in SIG_v$ **do**
  for all $w \in Horizon(v, t)$ **do**
    for all $sig_w \in SIG_w$ **do**
      Compute $D(sig_v, sig_w)$
    end for
  end for
  $E_a = E_a \cup (v, w, sig_v, sig_w)$ having $min(D(sig_v, sig_w))$
end for

3.2 Firework Query Model Over Clustered Network

To make use of our clustered P2P network, we propose a content-based query routing strategy called Firework Query Model. In this model, a query message is routed selectively according to the content of the query. Once it reaches its designated cluster, the query message is broadcasted by peers through the attractive connections inside the cluster much like an exploding firework as shown in Fig. 2. Our strategy aims to minimize the number of messages passing through the network, reduce the workload of each computer and maximize the ability of retrieving relevant data from the peer-to-peer network.

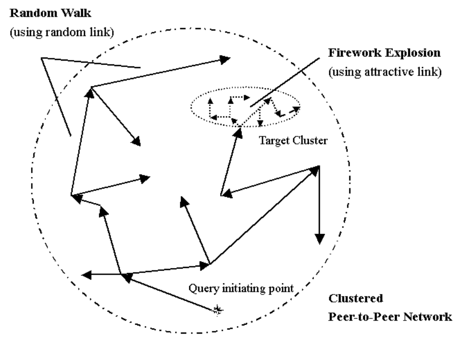

Figure 2: Illustration of firework query.

Here, we introduce the algorithm to determine when and how a query message is propagated like a firework in Algorithm 2. When a peer receives the query, it needs to carry out two steps:

1. **Shared File Look Up**–The peer looks up its shared information for those matched with the query. Let q be the query, and $\vec{q}$ be its vector representation, $REL(c_v, q)$ is the relevance measure between the query and the information c_v shared by peer v, it depends on a L_2 norm defined as,

$$REL(c_v, q) = \begin{cases} 1 & ||\vec{c_v} - \vec{q}|| \leq T \\ 0 & ||\vec{c_v} - \vec{q}|| > T, \end{cases}$$

   where T is a threshold defining the degree of result similarity a user wants. If any shared information within the matching criteria of the query, the peer will reply the requester. In addition, we can reduce the number of $REL(c_v, q)$ computations by performing local clustering in a peer, thus speeding up the process of query response.

2. **Route Selection**–The peer calculates the distance between the query and each signature value of its local clusters, sig_v , which is represented as,

$$D_q(sig_v, q) = \sum_i \frac{q_i - \mu_i}{\delta_i}, \quad sig_v = (\mu, \delta). \qquad (5)$$

If none of the distance measure between its local clusters' signature value and the query, $D_q(sig_v, q)$, is smaller than a preset threshold, θ, the peer will propagate the query to its neighbors through random connections. Otherwise, if one or more $D_q(sig_v, q)$ is within the threshold, it implies the query has reached its target cluster. Therefore, the query will be propagated through corresponding attractive connections much like an exploding firework.

In our model, we retain two existing mechanisms in Gnutella network for preventing query messages from looping forever in the distributed network, namely, the Gnutella replicated message checking rule and Time-To-Live (TTL) of messages. There is a modification on DISCOVIR query messages from the original Gnutella messages. In our model, the TTL value is decremented by one with a different probability when the message is forwarded through different types of connection. For random connections, the probability of decreasing TTL value is 1. For attractive connections, the probability of decreasing TTL value is an arbitrary value in [0, 1] called Chance-To-Survive (CTS). This strategy can reduce the number of messages passing outside the target cluster, while more relevant information can be retrieved inside the target cluster because the query message has a greater chance to survive depending on the CTS value.

Algorithm 2 Algorithm for the Firework Query Model

Firework-query-routing (peer v, query q)
for all $sig_v \in SIG_v$ **do**
  if $D_q(sig_v, q) < \theta$ (threshold) **then**
    if $rand() > CTS$ **then**
      $q_{ttl} = q_{ttl} - 1$
    end if
    if $q_{ttl} > 0$ **then**
      propagate q to all $e_a(a, b, c, d)$ where $a = v, c = sig_v$ or $b = v, d = sig_v$ (attractive link)
    end if
  end if
end for
if Not forwarding to attractive link **then**
  $q_{ttl} = q_{ttl} - 1$
  if $q_{TTL} > 0$ **then**
    forward q to all $e_r(a, b)$ where $a = v$ or $b = v$ (random link)
  end if
end if

4 Experiments and Results

Two main goals of routing algorithm in P2P network are to increase the percentage of desired result retrieved (Recall - R) and decrease the percentage of peers visited (Visited - V) for a query. Therefore, we define the query efficiency as R/V and we investigate how this quantity varies with different number of total peers. In our experiments, we generate a certain number of peers and randomly assign two to four classes of data points to each of them. Then, We initiate a query starting from a randomly selected peer and the retrieved data point is treated as desired result if it belongs to the same class as the query point. We simulate the environment using both controlled data and real data. For the controlled data, each class of data points follows a Gaussian distribution and we generate 200 classes of data points totally. For the real data, each class of data points is extracted from a category in CorelDraw's Image Collection and we select 200 different categories also.

We measure the query efficiency against the number of peers with four different routing methods or data sets: (1) Brute Force Search (BFS) with controlled data, (2) FQM with 1 signature value per peer and controlled data, (3) FQM with 3 signature values per peer and controlled data, and (4) FQM with 3 signature values and real data. As seen in Fig. 3, FQM outperforms BFS algorithm and it can perform even better if an appropriate number of signature values per peer is used. As expected, recall and visited peers percentage in BFS are more or less equal because data classes are evenly distributed among peers, the more peers visited, the more desired data retrieved. The curve of FQM follows a bell shape with a long tail. Query efficiency increases at first due to two reasons: (1) the percentage of peers visited is inversely proportional to the number of peers when TTL is fixed and (2) FQM advances the recall percentage when the query message reaches the target cluster. When the number of peers increases further, a query might not reach its target cluster, so query efficiency starts to drop. The result shows that the improvement in FQM is reduced when real data is used. This is when the real data cannot be clustered appropriately.

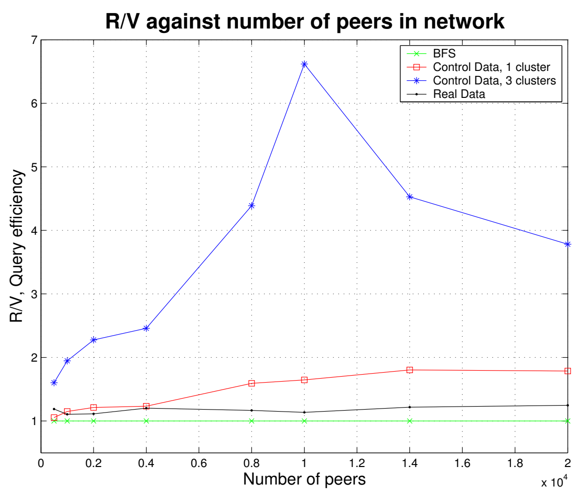

Figure 3: R/V against number of peers.

5 Conclusion

In this paper, we propose a peer clustering and content-based routing strategy to retrieve information based on their content efficiently over the P2P network. We verify our proposed strategy by simulations with different parameters to investigate the performance changes subject to different network size. We show that our FQM outperforms the BFS method in both network traffic cost and query efficiency measure.

Acknowledgments

DISCOVIR is built based on LimeWire open source project. This research work is supported in part by an Earmarked Grant (#2150317) from the Research Grants Council of Hong Kong, SAR.

References

[1] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. Adison Wesley, first edition, 1999.

[2] W. Bolosky, J. Douceur, D. Ely, and M. Theimer. Feasibility of a Serverless Distributed File System Deployed on an Existing Set of Desktop PCs. In *Proceedings of SIGMETRICS 2000 (Santa Clara, CA, June 2000)*.

[3] G. Coulouris, J. Dollimore, and T. Kindberg. *Distributed Systems Concepts and Design*. Addison-Wesley, third edition, 2001.

[4] DIStirbuted COntent-based Visual Information Retrieval. http://www.cse.cuhk.edu.hk/∼miplab/discovir.

[5] The Gnutella homepage. http://www.gnutella.com.

[6] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao. OceanStore: An Architecture for Global-Scale Persisten Storage. In *Proceedings of ASPLOS 2000 (Cambridge, Massachusetts, Nov. 2000)*.

[7] Modern Peer-to-Peer File-Sharing over the Internet. http://www.limewire.com/index.jsp/p2p.

[8] The Napster homepage. http://www.napster.com.

[9] C. H. Ng and K. C. Sia. Peer Clustering and Firework Query Model. In *Poster Proc. of The 11th International World Wide Web Conference*, May 2002.

[10] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable Content-Addressable Network. In *In Proc. ACM SIGCOMM*, August 2001.

[11] A. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. In *Proc. of the 18th IFIP/ACM International Conference on Distributed Systems Platforms*, November 2001.

[12] The Search for Extraterrestrial Intelligence homepage. http://www.setiathome.ssl.berkeley.edu.

[13] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In *Proc. of ACM SIGCOMM*, pages 149–160, August 2001.

[14] I. H. Witten, A. Moffat, and T. C. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann Publishers, Inc., second edition, 1999.

[15] B. Zhao, J. Kubiatowicz, and A. Joseph. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical report, Computer Science Division, U.C. Berkeley, April 2001.