An Interest-based P2P File Sharing System

Haiying Shen
Department of Computer Science and Computer Engineering
University of Arkansas, Fayetteville, AR 72701
Email: hshen@uark.edu

***Abstract*—Efficient file query is important to the overall performance of Peer-to-Peer (P2P) file sharing systems. In this paper, we introduce an interest-based P2P file sharing system based on a structured P2P. It groups peers based on both interest and proximity. The proposed system is able to support sophisticated routing and clustering strategies based on the file attribute and topology used. Simulation results demonstrate the efficiency of the proposed system in comparison with another P2P file sharing system. It dramatically reduces the overhead and yields significant improvements in file query efficiency.**

I. Introduction

Peer-to-Peer (P2P) system is a distributed system without any centralized control or hierarchical organization, in which each node has equal functionality. A key criterion to judge a P2P file sharing system is file location efficiency. To improve the efficiency, numerous methods have been proposed. Recently, a new wave of P2P systems is advancing an architecture of centralized topology embedded in decentralized systems; such a topology forms a super-peer network [1, 2, 3]. A super-peer topology consists of supernodes with fast connections and regular nodes with slower connections. A supernode connects with other supernodes and some regular nodes at the same time, and a regular node connects with a supernode.

Another class of methods to improve file location efficiency is proximity-aware structure [4, 5]. Recall that P2P overlay network is a logical structure constructed upon a physical network. That is, logical proximity abstraction derived from a P2P doesn't necessarily match the physical proximity information in reality. The shortest path (the least hop count routing) according to the routing protocol is not necessarily the shortest physical path. This mismatch becomes a big obstacle for the deployment and performance optimization of P2P file sharing systems. A P2P system should utilize proximity information to reduce file query overhead and improve its efficiency. Proximity-aware clustering to group physically close peers is an effective technique to improve the efficiency. The third class of methods to improve file location efficiency is to cluster nodes based on their interests [6, 7, 8, 9, 10, 11, 12, 13, 14]. They lead to clusters of peers with similar interests, and in turn allows to limit the latency of searches required to find files.

This paper presents an interest-based P2P file sharing system on a structured P2P. It groups peers with the same interests. It also places files semantically together, and organize them in a fashion similar to a Yellow Pages. More importantly, it keeps all advantages of DHTs over unstructured P2Ps. Relying on DHT lookup policy rather than broadcasting, the construction of the proposed system consumes much less cost in mapping nodes to clusters and mapping clusters to semantic descriptions.

II. Interest-based P2P File Sharing System

The interest-based p2p file sharing system is developed based on Cycloid [15] structured P2P network. Cycloid is a lookup efficient constant-degree overlay with $n{=}d \cdot 2^d$ nodes, where d is its dimension. It achieves a time complexity of $O(d)$ per lookup request by using $O(1)$ neighbors per node. Each Cycloid node is represented by a pair of indices $(k, a_{d-1}a_{d-2}\dots a_0)$, where k is a cyclic index and $a_{d-1}a_{d-2}.....a_0$ is a cubical index. The cyclic index is an integer ranging from 0 to $d-1$, and the cubical index is a binary number between 0 and 2^d-1. The nodes with the same cubical index are ordered by their cyclic index mod d on a small cycle, which we call *cluster*. All clusters are ordered by their cubical index mod 2^d on a large cycle. The Cycloid DHT assigns keys onto its ID space by a consistent hashing function [16]. For a given key, the cyclic index of its mapped node is set to its hash value modulated by d and the cubical index is set to the hash value divided by d. A key will be assigned to a node whose ID is closest to its ID. Cycloid has self-organization mechanisms to deal with node joins, departures and failures. It has APIs, including `Insert(key,object)`, `Lookup(key)`, `Join()` and `Leave()`. Cycloid routing algorithm involves three phases. A file request is routed along the cluster of the requester, between clusters, and along the cluster in the destination's cluster. For more details of Cycloid, please refer to [15].

Without loss of generality, node interests can be uniquely identified. A node's interests are described by a set of attributes with globally known string description such as "image" and "music". The interest attributes are fixed in advance for all participating peers. The strategies that allow to describe content in a peer with metadata [9, 10, 11, 12, 13, 14] can be used to derive the interests of each peer. Due to the space limit, we don't explain the details of the strategies. The system uses cubical indices to distinguish nodes in different physical location, and uses cyclic indices to further classify physically close nodes based on their interests. Hilbert number represents the closeness of nodes. Specifically, the proposed system uses node i's Hilbert number, $\mathcal{H}_i$, as its cubical index, and the consistent hash value of node i's interest mod d, $S_i\%d$, as its cyclic index to generate node i's ID, denoted by $(S_i\%d, \mathcal{H}_i)$.

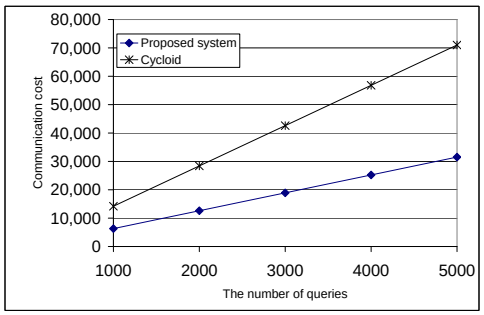

Fig. 1. Communication cost for file query.

If a node has a number of interests, it generates a set of IDs with different cyclic indices. Using this ID determination method, the physically close nodes with the same $\mathcal{H}$ will be in a cluster, and those with similar $\mathcal{H}$ will be in close clusters in the proposed system. Physically close nodes with the same interest have the same ID, and they constitute a sub-cluster.

The clusters in the proposed system function as super-peer network. The server in a sub-cluster acts as a centralized server to a subset of clients. Clients submit queries to their server and receive results from it, as in a hybrid system. Servers are also connected to each other as peers in a Cycloid, routing messages over this overlay network and submitting and answering queries on behalf of their clients and themselves.

III. Performance Evaluation

In the experiments, the DHT's dimension was set to 8 and it had 2048 nodes. We assumed there were 200 interest attributes (i.e. file keys), and each attribute had 500 files. We assumed a bounded Pareto distribution for the capacity of nodes. The number of queried files was set to 50, and the number of queries per file was set to 1000, unless otherwise specified. The file requesters and the queried files were randomly chosen.

The cost of file searching is directly related to message size and physical distance in hops of the message travelled; we use the product of these two factors of all file queries to represent communication cost. It is assumed that the size of a file query is 1 unit. Figure 1 plots the file searching communication cost of the proposed system and Cycloid. From the figure, we can see that the cost increases as the number of file queries increases, and Cycloid incurs considerably higher cost than the proposed system. There are two reasons for the observation. First, the proposed system reduces the lookup path length of Cycloid. Second, because Cycloid neglects proximity, file query messages travel long physical distances. In contrast, the proposed system proactively considers proximity in P2P construction for file query, such that the messages only travel between physically close nodes. Its shorter lookup path length and shorter physical message travel distance result in low-overhead and timely file queries.

Acknowledgements: This research was supported in part by U.S. NSF grants CNS-0834592 and CNS-0832109.

References

[1] Fasttrack. http://www.fasttrack.nu/index_int.html.

[2] Gnutella development forum. The Gnutella Ultrapeer Proposal, 2002.

[3] B. Yang and H. Garcia-Molina. Designing a super-peer network. In *Proc. of ICDE*, 2003.

[4] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker. Topologically-aware overlay construction and server selection. In *Proc. of INFOCOM*, 2002.

[5] M. Waldvogel and R. Rinaldi. Efficient topology-aware overlay network. In *Proc. of HotNets-I*, 2002.

[6] M. K. Ramanathan, V. Kalogeraki, and J. Pruyne. Finding Good Peers in Peer-to-Peer Networks. In *Proc. of IPDPS*, 2002.

[7] C. Hang and K. C. Sia. Peer Clustering and Firework Query Model. In *Proc. of WWW*, 2003.

[8] A. Crespo and H. Garcia-Molina. Routing indices for peer-to-peer systems. In *Proc. of ICDCS*, 2002.

[9] W. Nejdl, W. Siberski, M. Wolpers, and C. Schmnitz. Routing and clustering in schema-based super peer networks. In *Proc. of IPTPS*, 2003.

[10] P. A. Bernstein, F. Giunchiglia, A. Kementsietsidis, J. Mylopoulos, L. Serafini, and I. Zaihrayeu. Data management for peer-to-peer computing: A vision. In *Proc. of WebDB*, 2002.

[11] A. Y. Halevy, Z. G. Ives, P. Mork, and I. Tatarinov. Piazza: Data management infrastructure for semantic web applications. In *Proc. of WWW*, 2003.

[12] K. Aberer, P. Cudrè-Mauroux, and M. Hauswirth. The chatty web: Emergent semantics through gossiping. In *Proc. of WWW*, 2003.

[13] A. Löser, W. Nejdl, M. Wolpers, and W. Siberski. Information integration in schema-based peer-to-peer networks. In *Proc. of CAiSE*, 2003.

[14] W. Nejdl, M. Wolpers, W. Siberski, A. Löser, I. Bruckhorst, M. Schlosser, and C. Schmitz. Super-peer-based routing and clustering strategies for rdf-based peer-to-peer networks. In *Proc. of WWW*, 2003.

[15] H. Shen, C. Xu, and G. Chen. Cycloid: A scalable constant-degree lookup-efficient P2P overlay network. *Performance Evaluation*, 63(3):195–216, 2006.

[16] D. Karger, E. Lehman, T. Leighton, M. Levine, D. Lewin, and R. Panigrahy. Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web. In *Proc. of STOC*, pages 654–663, 1997.