

Search in Peer-to-Peer File-Sharing System: Like Metasearch Engines, But Not Really

Wai Gen Yee, Dongmei Jia, Linh Thai Nguyen
Information Retrieval Lab
Illinois Institute of Technology
Chicago, IL 60616
Email: {yee, jiadong, nguylin}@iit.edu

Abstract

Peer-to-peer information systems have gained prominence of late with applications such as file sharing systems and grid computing. However, the information retrieval component of these systems is still limited because traditional techniques for searching and ranking are not directly applicable. This work compares search in peer-to-peer information systems to that in metasearch engines, and describes how they are unique. Many works describing advances in peer-to-peer information retrieval are cited.

1 Introduction

File-sharing is a major use of peer-to-peer (P2P) technology. CacheLogic estimates that one-third of all Internet bandwidth is consumed by file-sharing applications [2]. Although much of this bandwidth consumption can be attributed to the large sizes of the shared files (i.e., media files), the fact that the usage has been consistent suggests its popularity. Increasing system size increases the importance of search technology that helps rank query results.

The task of effective ranking is exactly the goal of information retrieval (IR). However, traditional IR ranking does not function effectively in a P2P environment. Some work in the area of Web IR, however, is similar: the metasearch engine whose goal is to dispatch queries to other search engines and then rank their results. The goal of this paper is to explain the shortcomings of traditional IR in the P2P environment as well as the similarities and differences between metasearch engines and search in the P2P environment.

The impact of improved ranking should be significant in terms of resource usage as well. The popular Gnutella-based P2P file sharing systems basically flood the network with queries, which is bandwidth intensive. By improving

ranking effectiveness, fewer queries need to be issued to find a particular data object. Furthermore, effective ranking reduces the likelihood that a user will accidentally find and download interesting, but unrelated data.

2 Peer-to-Peer File Sharing Model

Our model is based on that which exists in common P2P file sharing systems, such as Gnutella and Kazaa [8]. Peers of a P2P system collectively **share** a set of **data objects** by maintaining local **replicas** of them. Each replica¹ (of a data object) is a file (e.g., a music file), which is identified by a **descriptor**. A descriptor is a **metadata set**, which is composed of **terms**. Depending on the implementation, a term may be a single word or a phrase. (A metadata set is technically a *bag* of terms, because each term may occur multiple times.)

A peer acts as a **client** by initiating a **query** for a *particular* data object (as opposed to any one of a *category* of data objects). A query is also a metadata set, composed of terms that a user thinks best describe the desired data object. A query is routed to all reachable peers, which act as **servers**. Query **results** are references to data objects that fulfill the **matching criterion**

$$D_O \supseteq Q, \text{ where } Q \neq \emptyset, \quad (1)$$

where D_O is the descriptor of data object O , and Q is the query. In other words, by design, the data object's descriptor must contain all the query terms [14].

A query result contains the data object's descriptor as well as the identity of the source server. The descriptor helps the user distinguish the relevance of the data object to the query, and the server identity is required to initiate the data object's download.

¹We use the term replica and data object interchangeably.

Once the user selects a result (for download), a local replica of the corresponding data object is made. In addition, the user has the option of manipulating the replica's descriptor. He may manipulate it for personal identification or to better share it in the P2P system.

The set of peers in a P2P file sharing system is connected in a general graph topology. Generally, peers join the system at arbitrary points, creating a random graph, although other topologies are possible and may yield performance benefits [13, 15, 18, 20].

Note that one major variation to the model is in what data are shared. We assume that data are binary objects, and, to be effectively shared, need to be identified via metadata in descriptors. Shared data, however, may be text. In this case, the data are self-describing, containing text that can be searched directly. This distinction may be important because the ranking scores of self-describing data objects are consistent, not being dependent on user-tunable descriptors. Furthermore, self-describing data objects are easier to rank because the information contained in descriptors tends to be more sparse and less consistent.

Another variation is in the way the network and data are organized. We assume a random graph, but more structure may be introduced, such as a ring or mesh, as suggested above. Furthermore, in such systems, data are often restricted in where they can be placed. For example, in the DHT described in [20], a ring network topology is enforced, and a data object is placed on a node with a node identifier that most closely matches the data object's object identifier. Consequently, replication of data is not allowed.

3 Similarity to Metasearch Engines

Metasearch engines' main selling points are their ability to search a larger data repository and return results that are ranked better. These features stem from the fact that different data sources (other search engines) may index different data repositories, and, if their data repositories overlap, they can improve overall ranking by corroborating or contradicting each others' rankings.

The main tasks carried out by a metasearch engine include source selection, query dispatching, result selection, and result merging [11]. Source selection is the process of selecting the search engines to query. Query dispatching is the process of translating a query to the search engine's local format, preserving the semantics of the final results. Result selection is the selection of the results returned by a search engine for consideration in the final results. Result merging is the ranking of the selected documents.

These tasks have analogs in P2P file sharing systems because they and metasearch engines both work in an environment where there are many independent, heterogeneous data sources. The difference, as we shall see below, is in the

dynamism of the P2P environment.

3.1 Source Selection

Source selection in metasearch engines is done by maintaining statistics on the contents of each search engine. This is often done by sampling [5]. Terms are extracted from a pre-defined corpus, and the contents of a search engine are deduced based on the results.

Source selection in P2P file sharing systems is related to the task of query routing because of the topology of the network. Because of the size of the network, two peers may be connected, but only through intermediate peers. The most general form of source selection, used by Gnutella [8], is through flooding, where queries are routed to all neighbors in a breadth-first fashion, until a certain query time-to-live has expired.

Alternatives to flooding include the use previous query responses and the publication of content signatures to intelligently route queries. A peer may learn how responsive its neighbors are based on its responses to past queries and may route future queries accordingly [6, 16, 19]. Another way a peer can control routing is by looking at signatures that servers generate to describe their shared content [4, 9]. Queries are routed to servers whose signatures are the best matches.

Finally, many P2P routing algorithms are based on distributed hash tables [13, 15, 20], which efficiently route single keys to nodes with the closest-matching node identifiers. This problem searching for data objects described with multiple terms has been addressed in various ways, such as by generating a query for each term in the query or by using unhashed queries and unhashed signatures to describe content [3, 17, 21].

The sampling technique used by metasearch engines does not in general work for P2P file sharing systems because of the dynamism and topology of the latter. Because all peers are autonomous, they can leave the network at any time. This can render any collected statistics obsolete.

3.2 Query Dispatching

Query dispatching in metasearch engines has received little attention because it is considered straightforward. For example, to express term weights in a query, certain terms may have to be repeated in the translated query. A certain number of results may be desired from each search engine so that the total number of results returned to the metasearch engine is fixed; this can be adjusted as well.

Little attention has been paid in the literature to query dispatching in P2P file sharing systems as well. In general, all peers that have been "selected" are given the same query and are assumed to use the same ranking function. This is

generally the case in practical P2P file sharing systems. Furthermore, their results are not ranked—results basically have to conform to the matching criterion, described in Section 2. It therefore makes little sense to modify a query.

Attempts to improve performance by query transformation have been limited. One attempt is to use query expansion by creating graphs that connect related terms as synonyms [12]. This term graph is generated dynamically using the data stored locally on each peer. We are currently also using a process we call **query masking** to grow and shrink queries at the client or server to tune the results that eventually reach the client [24]. The idea behind query masking is to control the recall and precision of query results by selecting a subset of the terms in the query.

Applying traditional query transformation techniques in a P2P file sharing environment is also made difficult by the scale of the system. To effectively transform a query, the client must maintain statistical information about each server. The fact that the number of potential servers is in the millions obviates the use of traditional methods.

3.3 Result Selection

Result selection in metasearch engines is performed using knowledge of each search engine’s relevance to a particular query. In general, the more relevant search engine is asked to return more results, so that the metasearch engine can tune the final number of results to return to the user.

Result selection requires that the search engine rank results so that the top few can be returned. Ranking is generally not supported by servers in P2P file sharing systems. Recent research efforts, however, have incorporated ranking into the servers. In [9], for example, specialized nodes (known as **ultrapeers**) in the P2P system function as servers for particular content, and all peers that have such content are directly connected with it. All queries are routed to the relevant ultrapeer which, knowing the contents and ranking function of each of its attached peers (K-L divergence, in this case), can perform effective document selection. In [4], the client locally ranks results from servers keeping the top ones for the final result.

In general, however, metasearch engine result selection is inapplicable in P2P file sharing environments because it is difficult to control the servers to which a query is sent, not to mention to maintain knowledge of every potential server’s contents and ranking functions. Furthermore, it is difficult to maintain a P2P network that effectively clusters peers based on their shared content. This complicates the implementation of a hybrid architecture containing ultrapeers.

3.4 Result Merging

Result merging in metasearch engines generally employs ranking scores returned in the result set of each of the selected search engines. Each of these scores is normalized using knowledge of relevance of each search engine to the query and then a final result set is created containing the results with the highest normalized scores. Alternatively, if results refer to text documents, all top-ranked documents from each search engine can be downloaded by the metasearch engine to perform local ranking. Duplicate results can be handled by maintaining only the maximum, the sum, or the average rank score.

Result merging in P2P file sharing systems poses two fundamental problems. First, it assumes that the client has knowledge of the ranking process of the servers, which is unlikely, considering the heterogeneity and dynamism of the system. Second, it assumes that ranking can be done at all by any peer—not a certainty, considering the lack of global statistics.

Traditional IR techniques have been adapted to result merging in P2P environments with reasonable levels of effectiveness [4, 7, 10]. In [4], servers are ranked and then sequentially searched until a server’s result set does not affect the current top N results. In [10], semi-supervised learning and Kirsch’s algorithm are used for result merging in ultrapeers. A novel result in [22], however, is that group size—the number of results that refer to the same data object for a given query—is a better ranking metric than tf-idf. Furthermore, [23] shows that different ranking functions can be effectively used to find data of varying popularity in a P2P file sharing system.

4 Conclusion

Information retrieval in P2P file sharing systems is complicated due to the fact that global statistics are hard to collect due to the dynamism of the system in terms of the shared content, the availability of peers, and the topology of the network. One fundamental question therefore is to see whether IR can be performed at all in P2P systems.

The works cited in this paper present solutions that put various levels of constraints on the system (e.g., from a random to a fixed network topology). These constraints affect the applicability of a P2P system (e.g., a fixed topology would be appropriate for a grid system, but inappropriate for today’s file sharing systems). As systems become more constrained, it seems, traditional IR becomes more applicable, because more global statistics can be harvested. In unconstrained environments, the work becomes more challenging, as fewer assumptions can be made and less information is available; pre-existing IR techniques lose relevance. In effect, we propose that work be done carefully

considering the parameters of the P2P system, including the network topology, the autonomy of the peers, type of data shared, and the distribution of data. Contributions can still be made in constrained environments, but fundamental advances, such as link analysis in Web IR [1], can only be made in unconstrained ones.

References

- [1] S. Brin and L. Page. The anatomy of a large scale hypertextual web search engine. In *Proc. World Wide Web Conf.*, 1998.
- [2] CacheLogic. Cachelogic home page. Web Document. www.cachelogic.com.
- [3] A. Crainiceanu, P. Linga, J. Gehrke, and J. Shanmugasundaram. Querying peer-to-peer networks using p-trees. In *Proc. Wkshp. Web and Database*, Paris, France, 2004.
- [4] F. M. Cuenca-Acuna and T. D. Nguyen. Text-based content search and retrieval in ad hoc p2p communities. In *Proc. Intl. Wkshp Peer-to-Peer Comp*, May 2002.
- [5] P. G. Ipeirotis and L. Gravano. Distributed search over the hidden web: Hierarchical database sampling and selection. In *Proc. VLDB*, pages 394–405, 2002.
- [6] V. Kalogeraki, D. Gunopulos, and D. Zeinalipour-Yazti. A local search mechanism for peer-to-peer networks. In *Proc. ACM Conf. on Information and Knowledge Mgt. (CIKM)*, 2002.
- [7] I. A. Klamponos, J. J. Barnes, and J. M. Jose. Evaluating peer-to-peer networking for information retrieval within the context of meta-searching. In *Proc. Euro. Conf. on Inf. Ret.*, pages 528–536, 2003.
- [8] T. Klingberg and R. Manfredi. Gnutella protocol 0.6. Web Document, 2002. rfc-gnutella.sourceforge.net/src/rfc-0.6-draft.html.
- [9] J. Lu and J. Callan. Content-based retrieval in hybrid peer-to-peer networks. In *Proc. ACM Conf. on Information and Knowledge Mgt. (CIKM)*, pages 199–206, Nov. 2003.
- [10] J. Lu and J. Callan. Federated search of text-based digital libraries in hierarchical peer-to-peer networks. In *Proc. Euro. Conf. on Inf. Ret.*, 2005.
- [11] W. Meng, C. Yu, and K.-L. Liu. Building efficient and effective metasearch engines. *ACM Comp. Surveys*, 34(1):48–84, Mar. 2002.
- [12] K. Nakauchi, Y. Ishikawa, H. Morikawa, and T. Aoyama. Peer-to-peer keyword search using keyword relationship. In *Proc. Wkshp. Global and Peer-to-Peer Comp. Large Scale Dist. Sys (GP2PC)*, pages 359–366, Tokyo, Japan, 2003.
- [13] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *Proc. ACM SIGCOMM*, 2001.
- [14] C. Rohrs. Keyword matching [in gnutella]. Technical report, LimeWire, Dec. 2000. www.limewire.org/techdocs/KeywordMatching.htm.
- [15] A. Rowstron and P. Druschel. Storage management and caching in past, a large-scale, persistent, peer-to-peer storage utility. In *Proc. SOSP*, 2001.
- [16] Y. Shao and R. Wang. Buddynet:history-based p2p search. y. shao, r. wang. in ecir-05. In *Proc. Euro. Conf. on Inf. Ret.*, 2005.
- [17] S. Shi, G. Yang, D. Wang, J. Yu, S. Qu, and M. Chen. Making peer-to-peer keyword searching feasible using multi-level partitioning. In *Intl. Wkshp. on P2P Sys. (IPTPS)*, 2004.
- [18] A. Singla and C. Rohrs. Ultrapeers: Another step towards gnutella scalability. Technical report, Limewire, LLC, 2002. rfc-gnutella.sourceforge.net/src/Ultrapeers_1.0.html.
- [19] K. Sripanidkulchai, B. Maggs, and H. Zhang. Efficient content location using interest-based locality in peer-to-peer systems. In *Proc. IEEE INFOCOM*, 2003.
- [20] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proc. ACM SIGCOMM*, 2001.
- [21] C. Tang, Z. Xu, and S. Dwarkadas. Peer-to-peer information retrieval using self-organizing semantic overlay networks. In *Proc. ACM SIGCOMM*, Aug. 2003.
- [22] W. G. Yee and O. Frieder. On search in peer-to-peer file sharing systems. In *Proc. ACM SAC*, Santa Fe, NM, Mar. 2005.
- [23] W. G. Yee, D. Jia, and O. Frieder. Finding rare data objects in p2p file-sharing systems. In *Proc. IEEE P2P Conf.*, Constance, Germany, Sept. 2005.
- [24] W. G. Yee, L. T. Nguyen, and O. Frieder. Improving search performance in p2p file sharing systems by query masking. In *Under Review*, June 2005.