

# Full-Text Federated Search in Peer-to-Peer Networks

**Jie Lu**

Language Technologies Institute  
School of Computer Science  
Carnegie Mellon University

## Abstract

Peer-to-peer (P2P) networks integrate autonomous computing resources without requiring a central coordinating authority, which makes them a potentially robust and scalable model for providing federated search capability to large-scale networks of text digital libraries. However, P2P networks have so far mostly used simple search techniques based on document names or controlled-vocabulary terms, and provided very limited support for full-text search of document contents.

This dissertation provides solutions to full-text federated search with relevance-based document ranking within an integrated framework of P2P network overlay, search, and evolution models. Previous notions of P2P network architectures are extended to define a network overlay model with desired content distribution and navigability. Existing approaches to federated search are adapted, and new methods are developed for resource representation, resource selection, and result merging in a network search model according to the unique characteristics of P2P networks. Furthermore, autonomous and decentralized algorithms to evolve the network topology into one with desired search-enhancing properties are proposed in a network evolution model to facilitate effective and efficient full-text federated search in dynamic environments.

To demonstrate that the proposed solutions are both effective and practical, two P2P testbeds consisting of thousands of real-content text digital libraries and hundreds of thousands of automatically generated queries are developed. Evaluation using these testbeds provides strong empirical evidence that the approaches proposed in this dissertation provide a better combination of accuracy, efficiency and robustness than more common alternatives.

## 1 Introduction

A very large number of text digital libraries<sup>1</sup> were developed during the last decade. Nearly all of them use some form of relevance-based ranking, in which term frequency information is used to rank documents by how well they satisfy each query. Many of them allow free search access to their contents via the Internet, but do not provide complete copies of their contents upon request. Many do not allow their contents to be crawled by Web search engines. In consequence, the

---

<sup>1</sup> A digital library is “a set of resources and associated technical capabilities for creating, searching and using information” (Borgman 1999). A text digital library consists of a collection of documents primarily in text form.

contents provided by these digital libraries cannot be accessed by Web search engines such as Google and AltaVista that only conduct search on materials that can be copied to centralized repositories. How best to provide federated search<sup>2</sup> across such independent digital libraries is an unsolved problem often referred to as the “Hidden Web” problem.

Many text digital libraries also reside in enterprise networks. Collecting and maintaining an internal centralized repository is not always practical for heterogeneous, multi-vendor, or lightly-managed enterprise networks. Federated search in these environments requires an effective, convenient and cost-efficient solution that is decentralized in nature.

*Peer-to-peer (P2P)* networks integrate autonomous computing resources without requiring a central authority, which makes them a good choice for providing federated search capability to a large number of digital libraries on the Internet and in enterprise networks. The decentralized nature of P2P networks also enables high robustness and high scalability, which are critical to federated search over large numbers of digital libraries. To capitalize on the power and scaling properties of large distributed P2P systems, we were motivated to explore federated search of text digital libraries in P2P networks.

The majority of the previous research on search in P2P networks has focused on P2P networks used for file-sharing or distributed information storage. As a result, the search techniques developed for P2P networks have so far mostly been limited to simple matching over document names, identifiers, or keywords from a small vocabulary for limited-domain contents (Tsoumakos and Roussopoulos 2003b) (Sakaryan et al. 2004) (Li and Wu 2005). In contrast, it has already become common practice for text digital libraries developed during the last decade to perform *full-text search*, in which the full body of each text document is searched. In addition, term frequency information is often used to rank documents by how well they satisfy each query, and the search result is presented with some form of relevance ranking (“*full-text ranked retrieval*”). We argue that most of the recent research on P2P networks offers little useful guidance for providing full-text search of current text digital libraries with open-domain contents. Thus we focus on developing solutions to full-text ranked retrieval for federated search of text digital libraries in P2P networks.

## 1.1 Challenges

Most search techniques developed for full-text ranked retrieval assume a centralized control. Either all the documents are stored in a centralized repository, or information about all the documents is gathered at a centralized directory service. Traditional federated search (“distributed information retrieval”) only requires the aggregate directory information about each collection instead of each individual document. However, a centralized directory is still assumed to store the directory information of all the collections. A central authority for search purpose may be undesired in P2P networks due to its susceptibility to become a performance bottleneck or the target of malicious attacks, or because it requires IT infrastructure and resources that are unavailable or impractical in

---

<sup>2</sup> Federated search provides a single interface to support “finding items that are scattered among a distributed collection of information sources or services, typically involving sending queries to a number of servers and then merging the results to present in an integrated, consistent, coordinated format” (Baeza-Yates and Ribeiro-Neto 1999).

the environment. Therefore, federated search in P2P networks requires new solutions to extend existing techniques designed for environments with a global control in order to address the problem of how multiple distributed resources work autonomously and collaboratively to accomplish the retrieval task.

In addition to the decentralized nature of P2P networks, another characteristic that distinguishes P2P networks from traditional search environments is their dynamic nature. When peers in a network are permitted to arrive and depart at will, the structure of the network is under constant change, which affects how contents are distributed in the network and how easy it is to navigate from a source peer to a target peer using peer connections. Because P2P networks are decentralized, peers must rely on dynamic self-organization to adjust network structures. New approaches are needed to guide peer organization to achieve desired content distribution and network navigability.

## 1.2 Contributions

We extend previous notions of P2P networks to define a P2P *network overlay model* with enhanced functionalities in network architecture, and desired content distribution and navigability in network topology. Based on the network architecture extended to support full-text federated search, we develop a *network search model* to conduct effective and efficient federated search of text digital libraries. A *network evolution model* is also proposed to describe how a P2P network can dynamically and autonomously evolve into one with the defined network topology to further improve search performance. Our network overlay model, network search model, and network evolution model provide an integrated framework for full-text federated search of text digital libraries that provides accurate, efficient, robust, and scalable search.

The network overlay model uses *hubs* (directory services) to define the upper level or backbone of the network and *leaves* (digital libraries and users) to define the lower level of the network in a two-level hierarchy. Different functionalities of peers lead to different types and properties of connections between them. At the upper level in the hierarchy, the network has locational proximity of similar content areas and short global separation of dissimilar content areas for good navigability. At the lower level in the hierarchy, connections between digital libraries and hubs are organized to form cohesive content-based clusters for desired content distribution. In addition, connections between users and hubs are established based on users' interests. The key contributions of our network overlay model are i) its explicit recognition of distinctive structural requirements for peers with different functionalities, and ii) its effective integration of several network properties that can enhance search performance in a single architecture, both of which play critical roles in the effort to optimize the overall federated search performance of the network.

The network search model utilizes the network architecture and topology defined in the network overlay model in designing a full-text search mechanism that can offer a better combination of accuracy and efficiency than previous approaches to federated search of text digital libraries in P2P networks. We show in detail that the network search model is not a simple adaptation of existing solutions to full-text ranked retrieval. Its significance lies in our new development for each of the main components (resource representation, resource selection, and result merging) in consideration

of the characteristics and requirements of federated search in P2P networks. Specifically, the concept of a neighborhood is defined, and exponentially decayed resource descriptions of neighborhoods are used for resource selection of hubs; unsupervised threshold learning methods are developed for resource selection of providers; user modeling with adaptive query clustering is proposed to improve resource selection performance for queries representing persistent and long-term user interests; and Kirsch’s algorithm for result merging is extended to effectively merge multiple ranked lists without requiring global corpus statistics.

The network overlay model and the network search model are most useful if we can show that there are decentralized algorithms capable of evolving the topology of a P2P network into one with the search-enhancing properties described in the network overlay model and desired by the network search model. For this reason, we propose the network evolution model. Our network evolution model works effectively with open-domain contents using an unstructured full-text representation, which distinguishes it from previous topology evolution approaches that are constrained to limited domains and representations with small or controlled vocabularies. It adjusts connections dynamically to reflect frequent changes in the network without relying on a central control. In addition, it puts extra effort into avoiding high system overhead on topology evolution, and balancing load to make the network more scalable and robust.

Extensive experimental results and analyses are included to provide strong empirical evidence for the effectiveness and practicality of the proposed models. The two P2P testbeds developed for evaluation are two of the largest so far consisting of real-content text digital libraries, showing our effort towards applying the newly developed approaches to real operational environments and verifying their effectiveness.

## 2 Background and Related Work

In a peer-to-peer network, a *peer* (also called a “*node*”) refers to an abstract notion of a participating entity in the network. There are three different types of *functional units* in an information-sharing P2P network, namely *provider* which provides information, *consumer* which requests information, and *service* which provides functionality to facilitate efficient and effective search of relevant information. A peer may function as a single functional unit or as a combination of multiple functional units (e.g., both as a provider and as a consumer). Peers are organized in the network using the logical *connections* between them established at a protocol layer. *Logical connections* serve as data channels by which information is exchanged in the form of messages between peers. These logical connections are not necessarily associated with the underlying physical connections in the network. In this dissertation, by default “connections” refers to logical connections.

Three components of a P2P network are essential to federated search: *network architecture*, *search mechanism*, and *network topology*. A *network architecture* defines peer functions and relations associated with federated search. It determines the search mechanisms and network topologies that can be supported in the network. Basic network architectures include brokered, completely decentralized, hierarchical, and structured P2P architectures. A *brokered P2P architecture* uses a single, centralized directory service, which is simple, efficient, and easy to control, but less robust and not appropriate for distributed environments without the support of a central IT infrastructure.

A *completely decentralized P2P architecture* requires each peer to provide directory services, which is robust and easy to deploy in distributed environments, but less efficient. A *hierarchical P2P architecture* relies on multiple collaborative regional directory services, which combines the strengths of brokered and completely decentralized P2P architectures, but requires extra system overhead for collaboration among directory services. A *structured P2P architecture* uses a distributed hash table for directory services, which is efficient, but restrictive and less flexible.

A *search mechanism* specifies the activities required for search, which mainly includes representing contents (*resource representation*), locating relevant resources (*resource location*), and integrating results (*result integration*). Representations with small sizes such as *name-based*, *free-text*, and *controlled-vocabulary* representations are widely used in all P2P architectures. Adopting *full-text* representations is common in completely decentralized and hierarchical P2P architectures, but rare in brokered P2P architectures, and it requires additional processing in structured P2P architectures. Resource location uses centralized mapping in brokered P2P networks, message passing in completely decentralized and hierarchical P2P networks, and distributed hash table lookup in structured P2P networks. Result integration in existing P2P networks has so far relied on simple methods based on the frequency of term matching or content-independent features, and hasn't provided any solution to relevance-based result integration.

A *network topology* describes how peers are connected in the network. It affects the effectiveness and efficiency of any particular search mechanism. Previous work has discovered three properties of network topologies that can enhance the performance of federated search: interest-based locality, content-based locality, and small-world. *Interest-based locality* puts a peer near to those peers whose contents are similar to its interests so that its typical queries only need to travel a short distance to locate relevant contents. *Content-based locality* keeps peers with similar contents near to one another to make locating most relevant contents efficient. *Small-world* properties enable short path lengths between any pair of peers and provide good navigability for efficient query routing.

Despite the development of several main ingredients for search mechanisms and network topologies in prior work, no existing work has provided a complete recipe for full-text ranked retrieval in P2P networks. The study of these approaches inspired our development of network architecture (network overlay model), search mechanism (network search model), and network topology (network overlay model and network evolution model) for full-text federated search.

The development of our search mechanism for full-text federated search in P2P networks also benefits from previous research on full-text ranked retrieval using a single, centralized directory service ("*distributed information retrieval*"). Viewing full-text federated search in P2P networks as distributed information retrieval in a particular type of environment with possibly multiple directory services, we use the techniques developed for traditional distributed information retrieval as a starting point, and further introduce new methods to fit the solutions to the characteristics and requirements of full-text federated search in P2P networks.

### 3 Network Overlay Model

A *network overlay* model describes the functionalities and organization of peers in the network at a protocol layer using a *network architecture* and a *network topology*. We develop our network overlay model for full-text federated search based on a hierarchical P2P architecture due to the following reasons. First, a hierarchical P2P architecture uses multiple regional directory services (hubs) to work collectively to cover the network without relying on a central authority. Therefore, it is more appropriate than a brokered P2P architecture for distributed environments that lack the support of a central IT infrastructure but need practical search solutions to full-text ranked retrieval. Second, hierarchical P2P architectures rely on peers with more power and bandwidth to conduct directory services. By concentrating most processor and bandwidth usage at a few heavy-duty peers, peers with limited computing and network resources can be relieved of the burden of directory services, and the overall network traffic can be reduced. Dedicating some peers to directory services also enables sophisticated techniques to be applied for effective and efficient search, and provides more opportunities for peers to learn about the network and self-organize into desired network topologies. Therefore, hierarchical P2P architectures provide a better choice than completely decentralized P2P architectures. Third, because structured P2P architectures use distributed hash tables to distribute inverted lists among peers, and require multiple inverted lists from multiple peers to be intersected for multi-term queries, it is difficult for them to support effective and efficient full-text search with relevance-based result integration. In contrast, the flexibility of hierarchical P2P architectures allows existing techniques to be adapted and new approaches to be developed for full-text ranked retrieval in a relatively straightforward manner.

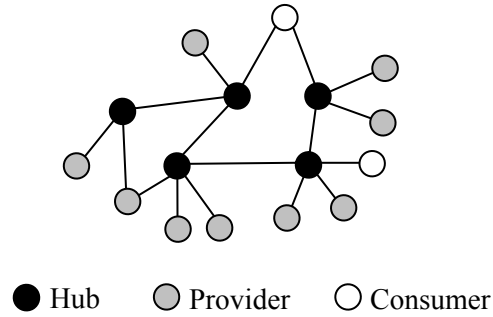
#### 3.1 Network Architecture

As in the basic hierarchical P2P architecture, our hierarchical P2P architecture consists of two types of peers, organized into two levels: a lower level of leaves and an upper level of hubs. A peer located at the leaf level can be a provider, a consumer, or a combination of both. A peer located at the hub level is a directory service.

A *consumer* represents a user with information requests. It initiates the search process by generating a query message and relaying the message to the selected hubs, and finalizes the search process by collecting the returned results and presenting them to the user.

A *provider* is a digital library that shares text documents in the network. It provides a full-text search service by running a document retrieval algorithm over a local document collection and returning a list of documents with relevance-based ranking in response to a query. Each provider can choose its own document retrieval algorithm. A provider also provides an accurate description of its content to its neighboring hubs upon request.

A *hub* is a resource that provides directory services to a region of the network including all the leaves that connect to it. It acquires and maintains content information about its neighboring hubs and providers, and uses it to provide resource selection (query routing) and result merging (integrating results returned by multiple providers) services to the network.



**Figure 3.1 Illustration of the network architecture.**

In previous P2P architectures, each connection is a data channel between a pair of peers. For a peer with multiple functional units (e.g., both as a consumer and as a provider), each of its connections to other peers is shared by these units. Because different functional units use the same connection for different purposes, it is difficult to optimize the utilities of all functional units of the same peer using a single set of connections since different functional units may desire different sets of connections. For example, it is quite likely that a peer’s utility as a provider is optimized when it connects to one set of peers, but its utility as a consumer is optimized by connecting to a different set of peers since its information need as a consumer may not be always related to the content it shares as a provider. To solve this problem, in our hierarchical P2P architecture each functional unit on a peer can have its own connections to other peers. In other words, a connection that links a pair of peers actually links a pair of functional units on these peers.<sup>3</sup> By doing so, connections to a peer with multiple functional units for different purposes can be established and adjusted independently so that it is easier to achieve an optimal setting for the utilities of all functional units simultaneously.

In our hierarchical P2P network architecture, leaves (providers and consumers) only connect to hubs. Hubs connect with leaves and other hubs. Each hub acts as a gateway between its connecting leaves and the rest of the network, so leaves are relieved of the responsibility to relay messages not related to their contents or interests. Figure 3.1 illustrates the defined network architecture.

### 3.2 Network Topology

For a network that adopts the hierarchical P2P architecture described in Section 3.1, its topology has the components of hub-provider topology, hub-hub topology, and hub-consumer topology, each of which serves different purposes and desires different search-enhancing properties.

The *hub-provider topology* exhibits *content-based locality* by connecting providers with similar contents to the same hub to form a content-based cluster. Each content-based cluster defines a *content area* in the network. Because contents relevant to a query are often similar to each other, with content-based locality, most relevant contents are expected to be covered by a few hubs so that query routing can be both efficient and effective. In contrast, a randomly generated hub-provider

<sup>3</sup> For the convenience of description, the multiple functional units of a single peer may be treated as multiple “virtual” peers so that a connection between two functional units is the same as a connection between two (virtual) peers.

topology produces an arbitrary content distribution in the network. In this case, queries must be routed to hubs all over the network in order to locate sufficient relevant contents.

Having content-based locality in the hub-provider topology can also increase the robustness of full-text federated search. By covering a cohesive content area, the representation of the contents each hub serves remains relatively stable even if the members of its connecting information providers may change over time due to the dynamic nature of P2P networks. By comparison, in a randomly generated hub-provider topology, the representation of the content area covered by a hub may change dramatically as a result of the arrivals and departures of information providers. Because the performance of full-text federated search largely depends on the effectiveness of resource location while resource location relies on the quality of resource representation, content-based locality reduces the susceptibility of full-text federated search to dynamic content change in P2P networks.<sup>4</sup>

The *hub-hub topology* has *content-based small-world properties* by requiring each hub to maintain connections both to hubs covering similar content areas (*local* connections) and to hubs serving dissimilar content areas (*long-range* connections). In addition, the similarities between each hub's content area and those of its long-range hub neighbors should be approximately uniformly distributed in "similarity scales" to guarantee good navigability (Kleinberg 2000).

Federated search in a hierarchical P2P network relies on message-passing to first locate hubs that cover relevant contents before these hubs further direct messages to the connecting providers. By keeping hubs with similar content areas near to one another, relatively homogeneous *content regions* (a collection of similar content areas) can be formed at the hub level so that query routing can be more effective once a query arrives at the right content region. The existence of hub-hub connections that link dissimilar content regions of the network assures that a query can be routed to the targeted content region efficiently irrespective of where it starts. Therefore, efficient and effective full-text federated search in a hierarchical P2P network requires the properties of both locational proximity of similar content areas to form content regions and short global separation of dissimilar content regions, which are exactly small-world properties with a definition of peer distance based on content similarity.

*Interest-based locality* for the *hub-consumer topology* is established by connecting each consumer to those hubs with content areas similar to its interests. By directly connecting a consumer to hubs that are more likely to cover contents relevant to its requests, the amount of query routing among hubs can be greatly reduced without sacrificing search accuracy.

A consumer can perform two different types of search activities: search related to the user's persistent, long-term interests ("*characteristic search*") and search aimed to satisfy transient, ad-hoc information needs ("*uncharacteristic search*"). Interest-based locality can improve search performance for queries of characteristic search ("*characteristic queries*") which are conceptually

---

<sup>4</sup> One may argue that content-based locality reduces the robustness of federated search because if a hub fails unexpectedly, the content area covered by this hub becomes unreachable. However, this problem can be easily solved by each hub maintaining a small amount of redundant information about neighboring hubs' connections, so that the responsibility of a failing hub can be quickly taken over by its hub neighbors (Renda and Callan 2004). In contrast, the susceptibility of resource location performance to dynamic content change in a randomly generated hub-provider topology cannot be easily alleviated.

related to each other, since it is likely that hubs covering content areas relevant to past characteristic queries also cover relevant contents for future queries expressing similar interests, especially when the network topology exhibits content-based locality. But interest-based locality does not provide an effective solution to queries of uncharacteristic search (“*uncharacteristic queries*”) since their relevant contents are unlikely to be covered by hubs with content areas related to characteristic queries. As a result, for uncharacteristic queries, the consumer must resort to a more extensive search using a larger search radius and rely on certain properties of the hub-provider and hub-hub topologies for effective and efficient query routing, trading efficiency for accuracy.

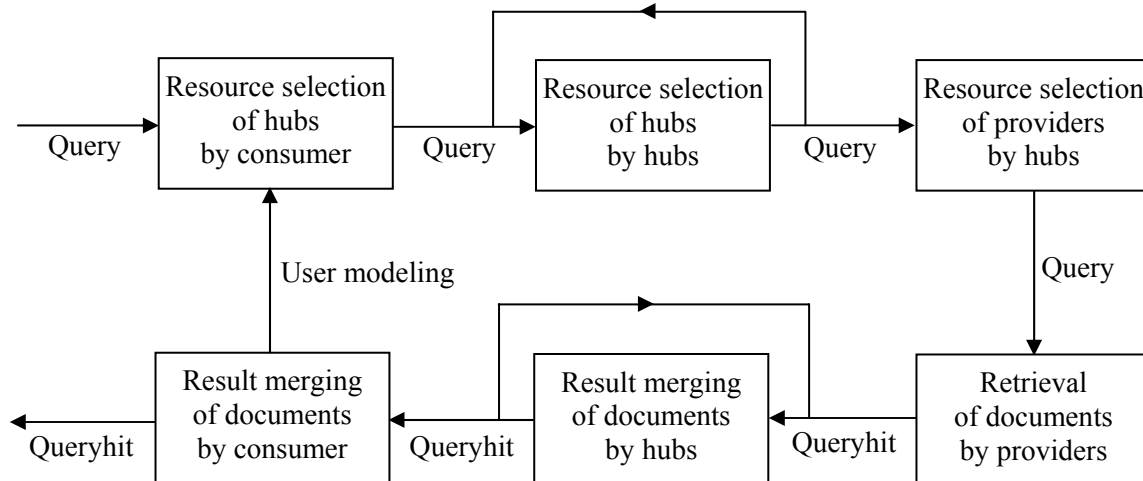
Compared with previous research, our network overlay model is unique in effectively incorporating all three search-enhancing properties (interest-based locality, content-based locality, and small-world properties) in a single framework to support full-text federated search, and making them suitable for distributed, dynamic environments with heterogeneous and open-domain contents.

## 4 Network Search Model

We define a network search model to describe a full-text federated search mechanism targeted at offering a better combination of accuracy and efficiency than existing common approaches for federated search of text digital libraries in P2P networks. The process of full-text federated search proceeds as follows. When a consumer has an information request, it sends a query message with an initial TTL (Time-To-Live) value to one or more hubs selected using interest-based selection (for characteristic search) or random selection (for uncharacteristic search). A hub that receives the query message uses its resource selection algorithm to rank and select one or more neighboring providers as well as hubs and routes the query to them with a decreased TTL until the message’s TTL reaches zero. A provider that receives the query message uses its full-text document retrieval algorithm to generate a relevance-based ranking of its documents and responds with a queryhit message that contains a list of the top-ranked documents. A hub is responsible for collecting the queryhit messages generated by multiple neighboring providers, using its result merging algorithm to merge multiple ranked lists of documents from these providers into a single, integrated ranked list, and returning it to the consumer. Finally, a consumer needs to merge results returned by multiple hubs. The search results for past queries are used by each consumer to construct a user model to improve the performance of interest-based selection for characteristic search. Figure 4.1 illustrates the interactions between the various components of the network search model. Solutions to the problems of resource representation, resource selection, and result merging required by full-text federated search are presented below.

### 4.1 Resource Representation

Resource descriptions required for resource selection by hubs use an full-text representation with term frequency information because it provides the most comprehensive description for text content among common representations and its large size is not an issue for hubs equipped with high connection bandwidth and processing power. The format of a resource description includes a list of terms with corresponding term frequencies (*collection language model*), and corpus statistics such as the total number of terms and documents provided or covered by the resource. The resource

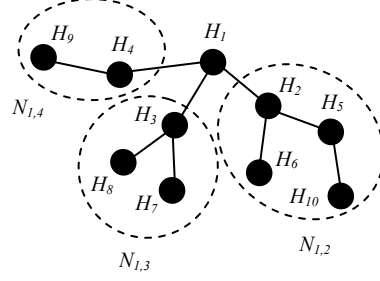


**Figure 4.1 Illustration of the network search model.**

could be a single provider (digital library), a hub that covers multiple neighboring providers, or a “neighborhood” that includes all the peers reachable from a hub.

Resource descriptions of providers are used by hubs for query routing (“*resource selection*”) among adjacent providers. Each provider provides an accurate resource description to its neighboring hubs upon request (e.g., using STARTS) (Gravano et al. 1997). The resource description of a hub is the aggregation of the resource descriptions of its neighboring providers. It describes the content area covered by the hub. Since hubs work collaboratively in hierarchical P2P networks, neighboring hubs can exchange with each other their aggregate resource descriptions. However, because hubs’ resource descriptions only have information for peers within one hop (providers directly connecting to them), if they are used by a hub to decide how to route query messages, the routing would not be effective when peers with relevant documents sit beyond this “horizon”. Thus for effective hub selection, a hub must have information about what contents can be reached if the query travels several hops beyond each hub neighbor. This kind of information is represented by the resource description of a *neighborhood*. A *neighborhood* of a hub  $H_i$  in the direction of its neighboring hub  $H_j$  is the set of hubs that a query message can reach by following the path from  $H_i$  to  $H_j$  and further traveling a number of hops. Each hub has its own view of neighborhoods near it, and each of its neighborhoods corresponds to one of its hub neighbors. Figure 4.2 illustrates the concept of neighborhood. Hub  $H_1$  has three neighboring hubs  $H_2$ ,  $H_3$  and  $H_4$ . Thus it has three adjacent neighborhoods, labeled  $N_{1,2}$ ,  $N_{1,3}$  and  $N_{1,4}$ . A neighborhood’s resource description provides information about the contents covered by all the hubs in this neighborhood. A hub uses resource descriptions of neighborhoods to route queries to its neighboring hubs.

A hub calculates the resource description of a neighborhood by aggregating the resource descriptions of all the hubs in the neighborhood decayed exponentially according to the number of hops so that contents located nearer are weighted more highly. For example, in the resource description of a neighborhood  $N_{i,j}$  (the neighborhood of  $H_i$  in the direction of  $H_j$ ), a term  $t$ ’s exponentially aggregated term frequency is calculated as:



**Figure 4.2 Three neighborhoods that can be reached from  $H_1$ .**

$$\sum_{H_k \in N_{i,j}} \frac{tf(t, H_k)}{F^{numhops(H_i, H_k)-1}} \quad (4.1)$$

where  $tf(t, H_k)$  is  $t$ 's term frequency in the resource description of hub  $H_k$ , and  $F$  is a factor for exponential decay, which can be the number of hub neighbors each hub has in the network.

The exponentially aggregated total number of documents in a neighborhood is calculated as below.

$$\sum_{H_k \in N_{i,j}} \frac{numdocs(H_k)}{F^{numhops(H_i, H_k)-1}} \quad (4.2)$$

The creation of resource descriptions of neighborhoods requires several iterations at each hub. A hub  $H_i$  in each iteration calculates and sends to its hub neighbor  $H_j$  the resource description of neighborhood  $N_{j,i}$  (denoted by  $ND_{j,i}$ ) by aggregating its hub description  $HD_i$  and the most recent resource descriptions of neighborhoods it received previously from all of its neighboring hubs excluding  $H_j$ . The calculation of  $ND_{j,i}$  is provided by Equation 4.3.

$$ND_{j,i} = HD_i + \sum_{H_k \in directneighbors(H_i) \setminus H_j} \frac{ND_{i,k}}{F} \quad (4.3)$$

The stopping condition could be either the number of iterations reaching a predefined limit, or the difference in resource descriptions between adjacent iterations being small enough.<sup>5</sup> Each hub can run the creation process asynchronously.

The process of maintaining and updating resource descriptions of neighborhoods is identical to the process used for creating them. The resource descriptions of neighborhoods could be updated when the difference between the old and the new value is significant, or periodically, or when a peer leaves the network.

For networks that have cycles, the frequencies of some terms and the number of documents may be overcounted, which may affect the accuracies of resource descriptions. Even so, empirical evidence

<sup>5</sup> The stopping condition used in our experiments was the difference in resource descriptions between adjacent iterations being smaller than a threshold, so that each hub could dynamically determine the number of iterations based on content and network conditions.

shows that resource selection using resource descriptions of neighborhoods in networks with cycles is still quite efficient and accurate (Lu and Callan 2005) (Lu and Callan 2006a).

Because the size of a full-text resource description is proportional to its vocabulary size (i.e., total number of unique terms in the description), the communication and storage costs associated with acquiring and maintaining full-text resource descriptions are much larger than other common forms of resource representations, which may become a problem in large-scale P2P networks. One straightforward solution is to reduce the size of a resource description by pruning terms that are not good representatives of a resource's content. Because rare terms and terms that are common in general English are unlikely to contribute significantly to the content of a digital library, they become natural candidates to be pruned. We use a simple, but effective approach of pruning full-text resource descriptions by removing stopwords and terms with frequencies below a threshold.

## 4.2 Resource Selection by Hubs

To achieve both efficiency and accuracy, each hub ranks its neighboring providers by their likelihood of satisfying the information request, and neighboring hubs by their likelihood of providing a short path to peers with relevant information, and only forwards the request to the top-ranked neighbors. The information each hub utilizes for resource selection is the resource descriptions of its neighboring providers and neighborhoods.

Direct comparison between providers and neighborhoods is difficult because it requires very accurate size normalization in order to compare resource descriptions of providers and those of neighborhoods that are not of the same magnitude in vocabulary size and term frequency. For this reason, a hub handles separately the selection of its neighboring providers and hubs. In theory, each hub can choose its own resource selection algorithm independently. In practice, it is common in most operational and research P2P systems to require all of the hubs to use the same resource selection methods. In our experiments, because the Kullback-Leibler (K-L) divergence-based method that incorporates size effects has been shown to be one of the most effective resource selection algorithms tested on various testbeds in distributed information retrieval (Si and Callan 2004), we use it for selection of both neighboring providers and neighboring hubs at each hub. However, one could easily use instead one of the more sophisticated resource selection algorithms such as ReDDE (Si and Callan 2003); the framework is sufficiently general that it is not constrained to use any specific algorithm.

### 4.2.1 Resource Selection of Providers

Each hub uses the K-L divergence resource selection algorithm to calculate  $P(P_i | Q)$ , the conditional probability of predicting the collection of provider  $P_i$  given the query  $Q$ , and uses it to rank different providers and select the top-ranked ones (Si and Callan 2004).  $P(P_i | Q)$  is calculated as follows:

$$P(P_i | Q) = \frac{P(Q | P_i) \times P(P_i)}{P(Q)} \propto P(Q | P_i) \times \frac{\text{numdocs}(P_i)}{\sum_j \text{numdocs}(P_j)} \quad (4.4)$$

$P(Q)$  is neglected because its value is independent of providers and doesn't affect the ranking of providers. The prior probability of a provider  $P(P_i)$  is estimated using the number of documents in the collection of provider  $P_i$  divided by the total number of documents from all the providers connecting to the hub.  $P(Q | P_i)$  is calculated using Equation 4.5:

$$P(Q | P_i) = \prod_{q \in Q} \frac{tf(q, P_i) + \mu \times P(q | G)}{\text{numterms}(P_i) + \mu} \quad (4.5)$$

where  $tf(q, P_i)$  is the term frequency of query term  $q$  in provider  $P_i$ 's resource description (collection language model). Dirichlet smoothing is used in the calculation of  $P(Q | P_i)$ , and  $\mu$  is the smoothing parameter (Zhai and Lafferty 2001). The background language model  $P(q | G)$  used for smoothing is calculated based on maximum likelihood estimation with Laplacian smoothing from the aggregation of all the resource descriptions of the hub's neighboring providers and neighborhoods:

$$P(q | G) = \frac{tf(q, G) + 1}{\text{numterms}(G) + \text{vocabularysize}(G)} \quad (4.6)$$

#### 4.2.2 Thresholding for Resource Selection of Providers

Thresholding for resource selection of providers in a hierarchical P2P network is the process for a hub to decide how many top-ranked neighboring providers to select for relaying each query message. In previous work of distributed information retrieval with a single directory service, typically the simple approach of selecting the top-ranked neighbors up to a predetermined number is used. We refer to this method as *resource selection of providers based on a fixed threshold*. The value of this fixed threshold is tuned empirically to optimize system performance. In a hierarchical P2P network, the number of each hub's neighboring providers is unknown in advance due to its dynamic nature, so it would be unclear how to set a fixed threshold that produces the optimal performance. In addition, different hubs may have different "optimal" threshold values, and the "optimal" threshold value of each individual hub may change over time. Therefore, it is not appropriate to use a static, predetermined fixed threshold for resource selection of providers in hierarchical P2P networks. It is desirable that hubs have the ability to learn their own thresholds automatically and autonomously. We refer to the method that selects the top-ranked neighboring providers whose relevance-based ranking scores are larger than a learned threshold as *resource selection of providers based on a learned threshold*.

It is preferable that threshold learning in P2P networks be conducted in an unsupervised manner because the user relevance feedback required as training data may not be easily available for federated search. Our goal is to develop a technique for each hub to learn its resource selection threshold without supervision based on the information and functionality it already has. Because each hub has the ability to merge multiple retrieval results into a single, integrated ranked list (more details in Section 4.4), as long as result merging has reasonably good performance, we could

assume that the top-ranked merged documents are “relevant”, or at least appropriate for the user to consider. Thus the distribution of the top-ranked merged documents over neighboring providers should provide useful hints about the number of relevant documents each provider is likely to return. If we further assume that a hub is permitted to flood its neighboring providers with a *small* number of queries, it can use the results of these queries as training data.

Each hub can compute its resource selection threshold for neighboring providers based on the thresholds it learned for individual training queries. Alternatively, a hub can also learn its resource selection threshold directly from the retrieval results of a set of training queries as a whole without first learning a separate threshold for each training query. We refer to the former approach as *individual-based threshold learning* and the latter as *set-based threshold learning*.

Each hub can either use the relevance-based ranking scores of its neighboring providers directly, or first normalize them into a fixed range (e.g., the lowest ranking score for a query is normalized to 0 and the highest ranking score is normalized to 1) and use the normalized scores instead. The advantages and disadvantages of these two approaches are complementary, i.e., the strength of one is the weakness of the other. On the one hand, because the range of original ranking scores is query-dependent, using original ranking scores for threshold learning requires training queries to be a good representative of future queries. In contrast, normalization makes ranking scores for different queries comparable and to some extent “query-independent” so that the threshold learned using normalized ranking scores is applicable to any queries. On the other hand, different ranges of original ranking scores for different queries may indicate a hub’s neighbors’ different overall degree of relevance (total amounts of relevant documents available) for these queries, which may indeed affect threshold learning for different queries. By normalizing original ranking scores for different queries into the same range, the learned threshold becomes reliant on the assumption that a hub’s neighboring providers have the same overall degree of relevance for all queries, which is not necessarily true.

Both original ranking scores and normalized ranking scores can be used in set-based threshold learning. However, individual-based threshold learning can only use normalized ranking scores because thresholds learned for different queries using original ranking scores are not comparable and therefore cannot be combined to generate a single threshold value.

For *individual-based threshold learning* with normalized ranking scores, a hub uses the following procedure to decide the threshold for selection of its neighboring providers with respect to a query:

1. Given a query, the hub uses its resource selection algorithm to calculate the ranking scores of its neighboring providers and sorts them in descending order;
2. The hub normalizes their scores using the following equation:

$$S' = \frac{S - S_{\min}}{S_{\max} - S_{\min}} \quad (4.7)$$

where  $S_{\max}$  is the maximum ranking score and  $S_{\min}$  is the minimum ranking score;

3. The hub forwards the query to its neighboring providers and merges the lists of documents returned by these providers (Section 4.4);
4. The hub uses up to the  $r$  top-ranked documents in the merged result as the set of “relevant” documents to calculate  $E(j)$  for each provider rank  $j$ , where  $r$  is a parameter of threshold learning (50 in our experiments) and  $E(j)$  is calculated as:

$$E(j) = 1 - \frac{1 + b^2}{\frac{b^2}{R(j)} + \frac{1}{P(j)}} \quad (4.8)$$

$$R(j) = \frac{\sum_{i=1}^j n_{rel}(i)}{N_{rel}} \quad (4.9)$$

$$P(j) = \frac{\sum_{i=1}^j n_{rel}(i)}{\sum_{i=1}^j n(i)} \quad (4.10)$$

where  $R(j)$  is the recall calculated based on the set of documents returned by providers ranked 1<sup>st</sup> to  $j^{\text{th}}$ ,  $P(j)$  is the precision of the set of documents returned by providers ranked 1<sup>st</sup> to  $j^{\text{th}}$ ,  $E(j)$  is the  $E$  evaluation measure corresponding to  $R(j)$  and  $P(j)$ ,  $b$  is a parameter which reflects the relative importance of recall and precision (with values of  $b$  greater than 1 indicating that precision is valued more than recall),  $n_{rel}(i)$  is the number of “relevant” documents returned by the  $i^{\text{th}}$  ranked provider,  $n(i)$  is the total number of documents returned by the  $i^{\text{th}}$  ranked provider, and  $N_{rel}$  is the total number of “relevant” documents for the query (which is equal to  $r$ ); and

5. The hub finds the rank  $j^*$  that gives the minimum  $E$  value and regards the normalized ranking score of the  $j^*$ <sup>th</sup> provider as the threshold for selection of neighboring providers with respect to the given query.

The individually learned thresholds for a set of training queries are averaged to get a single threshold at the hub.

For *set-based threshold learning*, the procedure a hub uses to learn the threshold for selection of its neighboring providers is the following:

1. Given a query, the hub uses its resource selection algorithm to calculate the ranking scores of its neighboring providers and sorts them in descending order;
2. If original ranking scores are used, go to the next step; otherwise, the hub normalizes ranking scores using Equation 4.7;

3. The hub forwards the query to its neighboring providers and merges the lists of documents returned by these neighbors;
4. The hub uses up to the  $r$  top-ranked documents in the merged result as the set of “relevant” documents to calculate for each provider neighbor how many of its returned documents are “relevant” in order to decide its relevance with respect to the query by comparing the number with  $n$ , where  $r$  is a parameter of threshold learning (50 in our experiments), and  $n$  is the minimum number of relevant documents a provider should provide in order to be considered relevant for a query;
5. After the hub finishes conducting the above steps for each training query, if original ranking scores are used, it defines non-overlapping groups spanning from 0 to the maximum term probability in the hub description so that all groups have roughly the same number of queries for training (at least 5 per group) and classifies each query into one of these groups based on the average probability of its terms in the hub description, which is a rough measure of the hub’s contents’ overall degree of relevance for the query; otherwise (if normalized ranking scores are used), go to the next step<sup>6</sup>;
6. The hub estimates the distributions of the ranking scores of relevant and non-relevant neighboring providers  $P(s | rel)$  and  $P(s | nonrel)$  for each query group using maximum likelihood estimation of Gaussian parameters if original ranking scores are used, or for a single query group consisting of all training queries using maximum likelihood estimation of empirical discrete distributions if normalized ranking scores are used; and
7. The hub uses Equations 4.11–4.14 to calculate  $\theta^*$  that gives the maximum value of a linear utility function  $U(\theta)$  and uses it as its threshold for selection of its provider neighbors for queries that belong to the same query group:

$$U(\theta) = N_{rel}(\theta) - N_{nonrel}(\theta) \quad (4.11)$$

$$\theta^* = \arg \max_{\theta} U(\theta) = \arg \max_{\theta} \{N_{rel}(\theta) - N_{nonrel}(\theta)\} \quad (4.12)$$

$$N_{rel}(\theta) = \alpha \times \int_{\theta}^{\max(s)} P(s \wedge rel) ds = \alpha \times \int_{\theta}^{\max(s)} P(s | rel) \times P(rel) ds \quad (4.13)$$

$$N_{nonrel}(\theta) = \alpha \times \int_{\theta}^{\max(s)} P(s \wedge nonrel) ds = \alpha \times \int_{\theta}^{\max(s)} P(s | nonrel) \times (1 - P(rel)) ds \quad (4.14)$$

where  $N_{rel}(\theta)$  and  $N_{nonrel}(\theta)$  are the numbers of relevant and non-relevant providers respectively whose ranking scores are above threshold  $\theta$ ,  $P(s \wedge rel)$  is the probability of a provider having score  $s$  and being relevant,  $P(s \wedge nonrel)$  is the probability of a provider having score  $s$  and being non-relevant,  $P(s | rel)$  is the probability of a relevant provider having score  $s$  (estimated in Step 6),  $P(s | nonrel)$  is the probability of a non-relevant provider having score  $s$  (estimated

---

<sup>6</sup> Because *normalized* ranking scores are somewhat “query-independent”, there is no need to classify training queries into different groups.

in Step 6),  $P(rel)$  is the probability of a provider being relevant,  $\alpha$  is the total number of provider neighbors, and  $\max(s)$  is the maximum relevance-based ranking score of a hub's neighboring providers for the training queries.

The integrals in Equations 4.13 and 4.14 are used when the corresponding probability distributions are represented with continuous probability density functions (e.g., Gaussian and uniform distributions). When discrete probability distributions are used, the integrals are replaced by sums.

We assume that each provider has equal probability of being relevant and non-relevant, i.e.,  $P(rel)$  has a uniform distribution. This is a reasonable assumption when each hub covers specific content area so that all of its neighboring providers have somewhat similar contents.

Since different threshold learning methods (individual-based and set-based) may overestimate or underestimate the number of neighboring providers to be selected in different cases, a hybrid approach may improve the quality. A straightforward way to combine these methods is to average the values determined by these methods for the number of providers to be selected and use this averaged value to decide how many top-ranked provider neighbors to select.

### 4.2.3 Resource Selection of Hubs

For resource selection of hubs, because selecting a neighboring hub is essentially selecting a neighborhood, the resource descriptions of neighborhoods are used to calculate the collection language models needed by the K-L divergence resource selection algorithm. The prior probability of a neighborhood  $P(N_i)$  is set to be proportional to the exponentially aggregated total number of documents in the neighborhood (Equation 4.2). Given the query  $Q$ , the probability of predicting the neighborhood  $N_i$  in the direction of a neighboring hub  $H_i$  is calculated as follows and used to rank neighboring hubs:

$$P(N_i | Q) = \frac{P(Q | N_i) \times P(N_i)}{P(Q)} \propto P(Q | N_i) \times \frac{\text{numdocs}(N_i)}{\sum_j \text{numdocs}(N_j)} \quad (4.15)$$

$P(Q)$  is neglected because its value is independent of neighborhoods and doesn't affect the ranking of hubs. The prior probability of a provider  $P(N_i)$  is estimated using the number of documents in the neighborhood  $N_i$  (Equation 4.2) divided by the total number of documents in the hub's neighborhoods.  $P(Q | N_i)$  is calculated using Equation 4.16:

$$P(Q | N_i) = \prod_{q \in Q} \frac{tf(q, N_i) + \mu \times P(q | G)}{\text{numterms}(N_i) + \mu} \quad (4.16)$$

where  $tf(q, N_i)$  is the term frequency of query term  $q$  in the resource description of neighborhood  $N_i$  (collection language model). Dirichlet smoothing is used in the calculation of  $P(Q | N_i)$ , and  $\mu$  is the smoothing parameter (Zhai and Lafferty 2001). The background language model  $P(q | G)$  for smoothing is calculated using Equation 4.6.

Although in theory the methods of threshold learning developed for resource selection of providers can be adapted to learn the threshold for resource selection of hubs, in practice it is more difficult to do so because i) the ranking of neighboring hubs is based on not only each hub neighbor's likelihood to cover relevant contents with its own providers, but also its potential to quickly reach other hubs with relevant contents, and ii) it is more challenging to effectively estimate the number of relevant documents in a neighborhood when distance to relevant documents must be taken into account. Since resource selection of hubs is based on neighborhood descriptions that are much more heterogeneous than provider descriptions on which resource selection of provider is based, it is easier to choose a hub selection threshold empirically that can work reasonably well for different hubs. Therefore, with the benefit of threshold learning for resource selection of hubs unlikely to offset its effort, a fixed threshold is used for resource selection of hubs.

### 4.3 Resource Selection by Consumers

Each consumer conducts initial hub selection to choose entry points where a query can be submitted to the network (Lu and Callan 2006b). The selected hub(s) use full-text resource selection to propagate the query among providers and other hubs. The consumer has the ability to distinguish different types of queries based on user modeling so that different search strategies can be applied to them to achieve the overall optimal performance. For characteristic queries representing the user's persistent, long-term interests, the consumer uses the user model it has learned from past search results to select hubs that are likely to locate relevant contents in their neighboring providers. For a network with content-based locality, this method can greatly reduce the amount of hub-hub query routing without degrading search accuracy. For uncharacteristic queries that express transient, ad-hoc information needs, because the limited (and often biased) information the consumer has learned as a byproduct of past search does not provide much of a clue about which hubs cover content areas relevant to these queries, it relies on a large search radius and hub-hub query routing to guarantee effectiveness since hubs maintain comprehensive information about the contents available in the network.

Non-hierarchical, incremental query clustering is used to group past queries in identifying a user's different interests, and each query cluster represents a *topic of interest*. New queries join existing query clusters if they are similar to any of them (determined by a clustering threshold  $T_{cluster}$ ); otherwise, new clusters are created. Because the small number of query terms does not provide a reliable basis for calculating similarities and clustering queries effectively, without assuming the availability of explicit or implicit user relevance feedback, the text of a small number of top-ranked merged documents for each query are used to generate query and cluster representations. Each cluster is associated with a table to record the interest-dependent performance of each hub with respect to this cluster, measured by how many documents returned by this hub to the consumer appear among the overall top-ranked merged documents for queries in the cluster.

Figure 4.3 provides an algorithmic description of how a consumer updates its user model using query clustering and measures the hubs' resource location performance for various interests based on the results they provided for queries in each cluster. To reduce clusters of uncharacteristic queries and effectively model the user's interest shift, the total number of query clusters is limited to

```

UPDATE_USER_MODEL( $q$ )
  /* Update query clusters with results for query  $q$  */
  get a set  $R$  of the  $D_{top}$  top-ranked merged documents for  $q$ 
  initialize  $N[\bullet] = 0$ 
   $q_d = \text{DocRepresentation}(R)$ 
  for each document  $d_j$  in  $R$ 
     $h_j = \text{GetSourceHub}(d_j)$ ;
     $N[h_j]++$ 
  end
  if exists at least one cluster  $c_i$  such that  $\text{KL}(c_i, q_d) < T_{cluster}$ 
    find the largest cluster  $c$  among all  $c_i$  with  $\text{KL}(c_i, q_d) < T_{cluster}$ 
  else
     $c = \text{NEWCLUSTER}()$ 
    initialize  $\text{NumTopDocs}[c][\bullet] = 0$ 
  end
  add  $q$  to cluster  $c$ 
  UpdateTimeStamp( $c$ )
  for each hub  $h_j$  that responds to  $q$ 
     $\text{NumTopDocs}[c][h_j] += N[h_j]$ 
  end

NEWCLUSTER()
  if the total number of clusters ==  $N_{max}$ 
    sort clusters by their time stamps
    delete the smallest cluster among the  $r$  least recently used clusters
  end
  return new cluster

```

**Figure 4.3 An algorithmic description of a consumer updating the user model and measuring the hubs' resource location performance based on the search results for a query  $q$**

$N_{max}$  and infrequently used clusters are removed. Small old clusters are removed before big old clusters because they are more likely to be clusters of uncharacteristic queries formed by chance.

Figure 4.4 describes in detail the initial hub selection conducted by a consumer for a query. When a query is issued, its query terms are used as its representation in determining which existing query clusters it is most similar to. To avoid classifying queries to clusters of uncharacteristic queries formed by chance and to make the description of the topic represented by each cluster more reliable, the comparison is restricted to clusters exceeding a certain size  $S_{min}$ . A classification threshold  $T_{classify}$  is used to distinguish characteristic queries representing long-term interests from uncharacteristic queries representing transient information needs. A characteristic query is issued to the hubs selected using *interest-based hub selection* with a small search radius, which selects hubs based on their measured resource location performance for the query clusters the query is most similar to. An uncharacteristic query is issued to randomly selected hub(s) with a default, larger search radius. A weighted  $k$ -nearest neighbor approach is used to increase the robustness of interest-based hub selection, where the value of  $k$  is determined by  $T_{classify}$  and the weights are related to the similarities between the query and the clusters. Each hub's weighted performance

```

INITIAL_HUB_SELECTION( $q$ )
  /* Compare query  $q$  to existing query clusters */
  characteristic = false
  initialize  $M[\bullet] = 0$ 
   $q_t = \text{TermRepresentation}(q)$ 
  for each cluster  $c_i$ 
    if  $\text{KL}(c_i, q_t) < T_{\text{classify}}$  AND  $|c_i| \geq S_{\text{min}}$ 
      characteristic = true
      UpdateTimeStamp( $c_i$ )
      for each hub  $h_j$  recorded by cluster  $c_i$ 
         $M[h_j] += \text{NumTopDocs}[c_i][h_j] / |c_i| \times \exp(-\text{KL}(c_i, q_t))$ 
      end
    end
  end
  /* Classify query  $q$  as characteristic or uncharacteristic for retrieval */
  if characteristic
    SetTimeToLive( $q, \text{ttl}_{\text{characteristic}}$ )
    Sort hubs by  $M[\bullet]$ 
    send  $q$  to the  $m$  top-ranked hubs
  else
    SetTimeToLive( $q, \text{ttl}_{\text{uncharacteristic}}$ )
    send  $q$  to randomly selected  $m$  hubs
  end
end

```

**Figure 4.4** An algorithmic description of initial hub selection by a consumer for a query  $q$

values for different clusters are accumulated and hubs are ranked and selected according to the accumulated performance.

## 4.4 Result Merging

In a hierarchical P2P network, each hub is responsible for merging results returned by its neighboring providers. If each provider can provide summary statistics (e.g., document length and how often each query term matches) for each of the retrieved documents, then a hub can recalculate very accurate normalized document scores and use them to generate an integrated ranked list of documents, which is essentially Kirsch's algorithm for result merging (Kirsch 1997). However, global corpus statistics are also required in recalculating document scores. To avoid the cost of acquiring and maintaining global corpus statistics at each hub which require aggregating information from *all* the hubs in the network, we propose for each hub to use the aggregation of the resource descriptions of its neighboring providers and neighborhoods to substitute for the corpus statistics. We refer to the algorithm that uses summary statistics of the returned documents (Kirsch's algorithm) and the aggregation of resource descriptions as corpus statistics to recalculate document scores as the *extended Kirsch's algorithm*.

A consumer may also need to merge results returned by multiple hubs. Because consumers don't maintain comprehensive information about the contents of other peers and corpus statistics as do hubs, they cannot use advanced result-merging algorithms. Thus only simple, but probably less

effective, merging methods can be applied at consumers. For example, results can be merged directly based on the document scores returned by hubs (“*raw score merge*”) or in a round robin fashion.

## 4.5 Evaluation

We created two new P2P testbeds consisting of 2,500 and 25,000 text digital libraries corresponding to real Web sites extracted from the TREC WT10g and .GOV2 datasets, which are among the largest testbeds to be used so far for research on P2P systems (Lu and Callan 2007). Our P2P testbeds also include tens of thousands of automatically generated queries and queries from a search engine query log. Based on our P2P testbeds, we progressively evaluate various components of our network search model against existing common alternatives: i) full-text resource selection was compared against flooding and random selection for query routing among hubs, and from hubs to providers, ii) the performance of resource selection of providers based on automatically learned thresholds was measured against the performance of search using a predetermined fixed threshold, iii) different types of resource descriptions were studied and compared in terms of their support for full-text resource selection of hubs, and iv) the extended Kirsch’s algorithm for result merging was evaluated and compared with merging using global corpus statistics and raw score merge. The overall conclusion is that the network search model provides more sophisticated search techniques and offers a better combination of accuracy and efficiency for full-text federated search in P2P networks.

## 5 Network Evolution Model

The network evolution model describes the process of dynamic self-organization in a P2P network, focusing on the evolution of network topology. The goal of our network evolution model is to establish and adjust the connections between peers dynamically and autonomously so that the resulting network topology exhibits the properties defined in the network overlay model to facilitate effective and efficient full-text federated search. The topology of a hierarchical P2P network has the components of hub-provider topology, hub-hub topology, and hub-consumer topology. Therefore, the topology evolution of a hierarchical P2P network includes the evolution of each of the three components. Because different components serve different purposes and desire different search-enhancing properties, the topology evolution of each component has its own objective. Specifically, the goal of hub-provider topology evolution is to establish content-based locality so that most contents relevant to a query are expected to be concentrated in a small part of the network at just a few hubs in order to improve the efficiency and effectiveness of query routing. Hub-hub topology evolution has the objective of having locational proximity of similar content areas and short global separation of dissimilar content areas (content-based small-world properties) in order to route a query quickly to its relevant content area no matter where it starts. The evolution of hub-consumer topology aims at reducing the effective search radius (TTL) by establishing permanent connections between consumers and hubs that cover content areas most similar to the characteristic interests of the consumers (interest-based locality).

## 5.1 Hub-Provider Topology

A hub-provider topology with content-based locality is constructed by requiring each hub's neighboring providers to form a cohesive content-based cluster so that each hub covers a content area. Since it is difficult to obtain a partition of the content space beforehand for digital libraries of unstructured text documents in open domains, the content area covered by each hub cannot be predetermined. Instead, it can only be determined implicitly by the contents of the providers already connecting to the hub. As the hub accepts into its content-based cluster more providers whose contents are similar to the content area it already covers, its content area may be updated dynamically to integrate the contents of these new members. Instead of solely relying on a single similarity threshold to determine whether to accept new providers and to control the granularity of a hub's content area, we use a more flexible clustering strategy which cultivates multiple sub-clusters within each hub's content-based cluster and spins off a sub-cluster to create a new content area when and only when the sub-cluster has grown to a certain size.<sup>7</sup> With this strategy, while the granularity of each sub-cluster may be relatively fixed by using a similarity threshold, the granularity of each hub's content area can be dynamically adjusted by increasing its number of sub-clusters or transferring its sub-clusters to other hubs.

An information provider's content is described using its full-text resource description. A sub-cluster is represented by the aggregation of the resource descriptions of its provider members. A hub's content area is represented by its resource description, generated by aggregating the representations of its sub-clusters. We use two similarity thresholds to distinguish among three levels of similarity between a provider's content and the contents covered by a sub-cluster: *high*, *marginal*, and *low*.<sup>8</sup> A provider's first priority is to join the most similar sub-cluster among all the sub-clusters to which it has a *high* similarity level. If it fails to find sub-clusters with *high* similarity, but has at least one sub-cluster with *marginal* similarity, it will join the content-based cluster of the hub that has the most similar sub-cluster with *marginal* similarity by initiating a new sub-cluster at this hub. The distinction between *high* and *marginal* similarity values is for controlling the granularity of each sub-cluster while allowing the granularity of a hub's content-based cluster to dynamically change by including more sub-clusters. If the provider has a *low* similarity level to all existing sub-clusters, it requests an empty<sup>9</sup> hub to initiate a new sub-cluster within a new content-based cluster. When all the hubs are non-empty, the provider initiates a new sub-cluster within the content-based cluster of the hub that has the most similar sub-cluster.

The join process of an information provider  $P$  proceeds as follows. If it was active in the network before, it first tries to connect to the hubs it previously connected to. If it fails or if it is completely new to the network, it finds an initial set of hubs by querying host-cache servers or by pinging the network. Then it arbitrarily chooses a hub from the list to connect to temporarily, which is

---

<sup>7</sup> If we assign a "virtual" hub to each sub-cluster, then having sub-clusters within a content-based cluster can be viewed as a mapping from multiple "virtual" hubs with similar contents to one physical hub.

<sup>8</sup> Although it is common and often desirable to use global similarity thresholds in cooperative P2P environments, in theory each hub can have its own similarity thresholds adjusted locally based on its workload, e.g., a busy hub can tighten its thresholds to accept fewer newcomers.

<sup>9</sup> A hub is "empty" if it has an empty content-based cluster, i.e., if it doesn't connect to any information provider. Otherwise, it is non-empty.

responsible for acquiring  $P$ 's resource description and starting its propagation among the hubs within a certain radius specified by the TTL of the message. Each non-empty hub that receives  $P$ 's resource description executes two operations. First, the hub computes the similarity between  $P$ 's resource description and the description of each of its sub-clusters, and sends back to  $P$  a message containing the value of the highest similarity along with the similarity level based on its thresholds. Second, the hub selects some neighboring hubs based on the similarity between  $P$ 's resource description and the description of each of its neighborhoods, and only propagates  $P$ 's resource description to those hubs most likely to have matched contents. Each empty hub that receives  $P$ 's resource description directly forwards it to its hub neighbors.  $P$  collects the messages from the hubs and chooses a hub to connect to based on the strategies described in the previous paragraph. It is straightforward to extend the strategies if  $P$  is allowed to join multiple content-based clusters.

If the size of a sub-cluster within a hub's content-based cluster exceeds a certain limit to indicate the emergence of a new or popular content area, the hub propagates a message among the hubs requesting an empty hub to take over. The transfer of the sub-cluster is conducted by connecting the provider members of the sub-cluster to the chosen empty hub to generate a new content-based cluster, and disconnecting them from the original hub. The attempt of initiating a new content area fails if no empty hub is available and the hub may try again sometime later.

If a hub becomes heavily connected due to the large number or sizes of its sub-clusters, it may propagate the descriptions of its sub-clusters towards selective directions at the hub level based on neighborhood descriptions to request other hubs covering similar content areas to share the load, and transfer one or more of its sub-clusters to the willing hubs. Alternatively, a hub may pose a limit on the size of its content-based cluster and stop accepting new providers once the limit has been reached. The former approach may result in higher degree of content-based locality at the expense of higher communication costs than the latter approach.

## 5.2 Hub-Hub Topology

Content-based small-world properties are small-world properties with a content-based definition of peer distance inversely related to the similarity between hubs' content areas. A hub-hub topology with content-based small-world properties can be constructed dynamically by each hub establishing *local* connections to hubs with similar content areas and a few *long-range* connections to hubs with dissimilar content areas. Due to the decentralized nature of a P2P network, no global information about either the graph distance (number of hops) or the content distance (the inverse of content similarity) between peers is readily available and acquiring such global information is inadmissible because of high cost. Therefore, each hub can only utilize the content information of other hubs in its local neighborhood to find similar (close) and dissimilar (remote) hubs to connect to. By iteratively adjusting its connections, a hub is able to connect to "globally" close and remote hubs because connection adjustment keeps bringing new hubs to its local network region. Experimental results indicate that the number of iterations required is only a small constant larger than the resulting (small) diameter of the hub-hub topology, which means that the hub-hub topology can converge to one with small-world properties fairly quickly.

The dynamic evolution of hub-hub topology proceeds as follows. When a hub  $H$  joins the network, if it was active in the network before, it first tries to connect to the hubs it previously connected to. If it fails or if it is completely new to the network, it obtains a list of existing hubs in the network by querying host-cache servers or by pinging the network. Because it doesn't have any providers connecting to it yet, it has an empty resource description. Since the similarity between a hub with an empty resource description and any other hub is undefined,  $H$  can only randomly choose its hub neighbors at the moment.  $H$ 's resource description is initialized when it responds to a provider's join request or a hub's transfer request to start a new content-based cluster. Then  $H$  connects to the joining provider or the providers in the transferred sub-cluster.

Each non-empty hub operates independently in a decentralized manner to select its own hub neighbors based on its local view of the content areas available in the network. Given the dynamic conditions of a P2P network, a hub  $H$  periodically evaluates its content similarity to its direct hub neighbors and their direct hub neighbors (i.e., hubs within two hops from it) using the hubs' resource descriptions exchanged among them, and adjusts its outgoing connections to link to several most similar hubs and a few dissimilar hubs using the following procedure:

1.  $H$  uses a threshold  $\rho^*$  to distinguish between similar and dissimilar hubs among the hubs within two hops from it;
2.  $H$  connects to  $M_{ol}$  most similar hubs whose incoming hub connection capacities have not reached their maximally allowed values  $M_i$ , where  $M_{ol}$  is the maximum number of outgoing local hub connections  $H$  can have;
3. If all of  $H$ 's similar hubs have reached their maximum incoming hub connection capacities  $M_i$ ,  $H$  requests them to recommend their similar hub neighbors, which may be repeated recursively until  $H$  establishes at least one local hub connection or the number of requests reaches a limit before it succeeds in finding any similar hub available for connection;
4.  $H$  selects  $M_{og}$  dissimilar hubs that have not reached their maximum incoming hub connection capacities  $M_i$  with probability:

$$P(H_{i \in \{j: GD(H, H_j) \leq 2\}} | CD(H, H_i) \geq \rho^*) = c CD(H, H_i)^{-\beta} \quad (6.1)$$

where  $M_{og}$  is the maximum number of outgoing long-range ("global") hub connections  $H$  can have,  $GD$  is the graph distance (number of hops) between hubs,  $CD$  is the content distance (the inverse of content similarity) between hubs, calculated based on the K-L divergence between hubs' resource descriptions,  $\beta$  is an exponent that essentially controls the "distance scale" of long-range connections (i.e., smaller  $\beta$  biases towards greater content distance and thus more dissimilar hubs, and larger  $\beta$  biases towards smaller content distance and thus less dissimilar hubs), and  $c$  is a normalizing constant.

By dynamically adapting each hub’s outgoing<sup>10</sup> connections at the hub level, hub-hub topology effectively maintains content-based small-world properties. Since each hub adjusts its connections only based on its local knowledge of the hubs that are located within two hops from it and possibly their recommended local contacts, no global information or control are necessary for the evolution of hub-hub topology. The step of limiting each hub’s connection capacity and recommending other similar hubs when its own connection capacity becomes full helps in distributing connections at the hub level in a less skewed manner to avoid concentrating a large number of connections at a few hubs for load balancing. Establishing long-range connections based on a power-law distribution of content similarity enables each hub to have nearly uniformly distributed long-range hub connections over all “distance scales” (“similarity scales”), which allows hubs to route any query efficiently towards its targeted content area (Kleinberg 2000).

### 5.3 Hub-Consumer Topology

A consumer  $C$  may perform *characteristic search* for which information requests are closely related to the user’s persistent interests in specific topics, and *uncharacteristic search* for which information requests are ad-hoc, transient in nature. While a consumer cannot do much with regard to hub-consumer topology to optimize the performance of uncharacteristic search, for the benefit of characteristic search,  $C$  should establish permanent connections to hubs that cover content areas most similar to its interests in order to take advantage of interest-based locality. In addition, if  $C$  is interested in several different topics, then the optimal set of hubs it should connect to may vary by topic. Based on the dynamic observation of initial hub selection conducted by the consumer for characteristic queries (Section 4.3), the consumer can establish permanent connections to those hubs that are most frequently selected, and periodically adjust its connections to adapt to the change in user’s interests and hubs’ contents.

### 5.4 Evaluation

Using the testbeds and the settings consistent with those used for evaluating the network search model, we progressively evaluated various components of our network evolution model by studying the properties of the dynamically evolved topology and measuring its effectiveness in enhancing federated search performance. Experimental results show that the topology evolution algorithms developed are effective in constructing a network topology with the desired search enhancing properties (interest-based locality, content-based locality, and content-based small-world properties) and load balance without relying on central coordination and control. The resulting network topology is capable of not only further improving the effectiveness of full-text federated search, but also increasing its robustness and scalability in dynamic environments. It also provides an environment where user modeling of a person’s characteristic information needs can lead to greater search accuracy and efficiency.

---

<sup>10</sup> The directions of hub-hub connections are only used for topology evolution. They are ignored when hub-hub connections are used as data channels to exchange messages between hubs.

## 6 Conclusions and Future Work

In this section, we summarize the research presented in this dissertation, discuss our major contributions and their significance, and point out directions for future work.

### 6.1 Contributions

Today, vast amounts of useful information contents exist in text form in distributed environments, many of which are hidden from conventional search engines. Effective and practical techniques must be developed to retrieve distributively located relevant contents for satisfying information needs. Federated search in peer-to-peer networks for accessing distributed information has become an important research topic that draws the attention of practitioners from multiple research areas, especially database management and networking. Although it can be regarded as a particular type of information retrieval activity in a particular type of environment, federated search in P2P networks has largely been explored independently from the research area of information retrieval. Previous work for federated search in P2P networks has mostly been targeted for known-item search of documents with representations based on names, annotations, or keywords from small, controlled vocabularies. P2P networks have so far provided very limited support for efficient full-text search of document contents with relevance-based document ranking. In contrast, full-text ranked retrieval has become common practice for information retrieval in traditional search environments, and is widely used for search over unstructured text documents of heterogeneous, open-domain contents.

The objective of this dissertation is to study federated search in P2P networks from an information retrieval perspective, and to develop new techniques to complement existing approaches in P2P networks that are mostly only applicable to limited domains. Particularly, we aim at providing comprehensive full-text ranked retrieval capability for federated search of text digital libraries in P2P networks. A *network overlay model* is proposed to extend previous notions of hierarchical P2P network overlays by enhancing the functionalities of peers and their connections, and explicitly defining the properties of a network topology capable of supporting effective, efficient, robust and scalable full-text federated search. The components of query routing, document retrieval, and result merging required for federated full-text ranked retrieval are incorporated in a *network search model*, which utilizes the functionalities and search-enhancing properties of the network overlay model to optimize search performance. The problem of constructing the proposed network overlay dynamically and autonomously is tackled with a *network evolution model*, which enables effective, efficient, and scalable topology evolution in decentralized, open-domain environments. Our development offers one of the first sets of practical solutions to enable full-text federated search in P2P networks, which can be deployed *now* for networks of at least a few tens of thousands of small- and medium-sized digital libraries. It not only broadens the application territory of sophisticated information retrieval techniques, but also expands the use of federated search in P2P networks to more domains.

Application areas of full-text federated search using P2P networks include but are not restricted to the “Hidden Web” and enterprise networks. The “Hidden Web” consists of independent or loosely affiliated text digital libraries on the Internet that provide search access to their contents via their

own search interfaces, but do not allow their contents to be crawled by Web search engines for centralized search. Using a P2P network to organize these digital libraries, “wrap” them in a standard P2P protocol, and conduct full-text federated search across them offers a single interface to access the “hidden” Web contents that cannot be reached using conventional Web search engines. Full-text federated search using P2P networks also provides an effective, convenient and cost-efficient solution to federated search of heterogeneous, multi-vendor, and lightly-managed distributed collections of text documents in enterprise networks or other environments where it is not feasible or practical to have a strong central IT infrastructure to support centralized search.

The models developed as integrated parts of the framework for full-text federated search in P2P networks can find their uses in other applications as well. One application for the network overlay and evolution models is large-scale centralized search. Due to the vast amount of information that needs to be stored and processed, even a centralized architecture has to rely on multiple connected computing and storage resources (the server farm) to provide the services of a central authority and control. Since the organization of these resources can be regarded as a particular type of network environment, the network overlay defined in our network overlay model can be used to organize them in order to provide regulated content distribution for more efficient query processing. The algorithms developed for the network evolution model can help to dynamically manage the resources in the face of constant changes in contents, requests and workload.

In recent years, with millions of individuals publishing contents and interacting with one another through the Internet, the analysis and management of large-scale social networks have drawn increasing attention. The distributed adaptive clustering approach proposed in the network evolution model can be adapted to more effectively organize groups or communities in social networks, and the network search model can be quite useful for distributed search in this highly dynamic environment.

Additional application areas that may benefit from our work are meta-search and personalized search. Our approach to user modeling, which recognizes and distinguishes long-term and short-term information needs and applies different search strategies accordingly, may provide useful insight into the development of effective techniques to optimize the overall search performance in these applications.

In addition to the models, the dissertation also provides valuable resources for the evaluation of federated search in large-scale P2P networks with realistic settings. Two P2P testbeds with large numbers of text collections and queries have been developed and used for evaluating existing and new approaches to full-text federated search and providing useful guidance for future research. Both testbeds have been published in a form that allows them to be used by other researchers.<sup>11</sup> Hopefully, our effort of building a useful evaluation platform with a flavor of large-scale P2P environments in the real world will benefit future research in this field.

---

<sup>11</sup> <http://www.cs.cmu.edu/~jielu/testbed.html>

## 6.2 Future Work

To minimize the computation and bandwidth usage of peers with limited resources, the network overlay defined in the proposed framework relies on hubs at the upper level of the network to act as communication gateways so that leaf peers (providers and consumers) don't need to connect directly among themselves. However, some lightweight communications at the lower level of the network may greatly improve search performance without significantly increasing costs. For instance, similar to content-based clusters formed by providers with similar contents, consumers having similar interests can form interest-based clusters and share their user models within each cluster so that the performance of resource selection conducted by consumers can be improved collectively. It would be interesting to extend our framework to incorporate collaborative search.

The proposed framework for full-text federated search in P2P networks currently includes models focusing primarily on search accuracy measured in precision/recall and efficiency measured in the percentage of the network reached for query messages. However, the framework is sufficiently flexible to allow more factors to be considered in addition to accuracy and efficiency in measuring search performance. For example, a utility-based approach can be used in the network search and evolution models to replace the similarity-based approach in ranking resources or establishing connections. The utility function can combine multiple factors such as accuracy, efficiency, authority, reliability, latency, monetary cost, etc., which will certainly make full-text federated search more practical and useful in a wider range of distributed environments.

Although our work has provided some solutions and analysis to address the issue of load balance for full-text federated search, more studies on dynamic load balancing are still needed to ensure a smooth execution of federated search and network evolution. New developments are required for hubs to dynamically and cooperatively monitor the load in the network to detect hotspots, and to alleviate the burdens of peers in the hotspots by using techniques such as result caching and query traffic redirection.

The topology evolution algorithms proposed in our network evolution model enable the network to recover from hub failures through dynamic adaptation, but the recovery may be slow without maintaining certain redundancy. One simple redundancy scheme to facilitate fast fault handling is for each peer to store the time-stamped information it obtains during topology evolution (with a time-out schedule), which may include the identities of other peers, their resource descriptions, and/or the similarities between resource descriptions. Because this information is the byproduct of topology evolution, this redundancy scheme does not require extra communication and coordination between peers. However, the simple redundancy scheme is not sufficient to avoid long-distance migration of peer locations in network topology when network connectivity and peer accessibility are restored from hub failures by establishing new connections. Because long-distance changes in network topology cause changes in the distribution of contents, and dramatic changes in content distribution result in costly updates in the resource descriptions required by full-text resource selection, a more complex redundancy scheme with extra communication and coordination among hubs should be developed in future work to minimize dramatic changes in topology and content distribution. For example, each hub can store information about local topological structure in a range larger than its immediate neighbors, so that the connections of a failing hub can be quickly taken over by its nearby hubs.

Our algorithms for federated search and topology evolution all take into consideration the highly dynamic nature of P2P networks. However, our work doesn't directly address several issues in dynamic environments such as the effect of content drift in collections, and the departures of providers and hubs. Future research to study the responsiveness of the current algorithms, and perhaps to extend them for fast-changing environments, might be required.

## References

- Baeza-Yates R and Ribeiro-Neto B (1999) *Modern Information Retrieval*. ACM Press/Addison Wesley, New York, NY.
- BearShare, <http://www.bearshare.com>.
- Borgman C (1999) What are digital libraries? Competing visions. *Information Processing & Management*, 35(3): 227-243.
- Edutella, <http://edutella.jxta.org>.
- eDonkey, <http://www.edonkey2000.com>.
- eMule, <http://www.emule-project.net>.
- Gnucleus, <http://www.gnucleus.com>.
- Gnutella v0.4, [http://www9.limewire.com/developer/gnutella\\_protocol\\_0.4.pdf](http://www9.limewire.com/developer/gnutella_protocol_0.4.pdf).
- Gnutella v0.6, <http://rfc-gnutella.sourceforge.net>.
- Gnutella2, <http://www.gnutella2.com>.
- Gravano L, Chang C, Garcia-Molina H and Paepcke A (1997) STARTS: Stanford proposal for internet meta-searching. In *Proceedings of the ACM-SIGMOD International Conference on Management of Data*.
- IRIS, <http://www.project-iris.net/>.
- JXTA, <http://www.jxta.org>.
- Javasim, <http://javasim.ncl.ac.uk>.
- KaZaA, <http://www.kazaa.com>.
- Kirsch S (1997) Document retrieval over networks wherein ranking and relevance scores are computed at the client for multiple database documents. U.S. Patent 5,659,732.
- Kleinberg J (2000) The small-world phenomenon: an algorithmic perspective. In *Proceedings of 32<sup>nd</sup> ACM Symposium on Theory of Computing*.
- Limewire, <http://www.limewire.com>.
- Li X and Wu J (2005) Searching techniques in peer-to-peer networks. To appear in Wu J ed. *Handbook of Theoretical and Algorithmic Aspects of Ad Hoc, Sensor, and Peer-to-Peer Networks*. CRC Press.
- Lu J and Callan J (2005) Federated search of text-based digital libraries in hierarchical peer-to-peer networks. In *Proceedings of the 27<sup>th</sup> European Conference on Information Retrieval Research (ECIR 2005)*.
- Lu J and Callan J (2006a) Full-text federated search of text-based digital libraries in peer-to-peer networks. *Journal of Information Retrieval, Volume 9, Number 4*.

- Lu J and Callan J (2006b) User modeling for full-text federated search in peer-to-peer networks. In *Proceedings of the 29<sup>th</sup> Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.
- Lu J and Callan J (2007) Content-based peer-to-peer network overlay for full-text federated search. In *Proceedings of 8<sup>th</sup> RIAO Conference on Large-Scale Semantic Access to Content*.
- Morpheus, <http://www.morpheus.com>.
- MusicNet, <http://www.musicnet.com>.
- Renda M E and Callan J (2004) The robustness of content-based search in hierarchical peer to peer networks. In *Proceedings of the 13<sup>th</sup> International Conference on Information and Knowledge Management (CIKM'04)*.
- RevConnect, <http://www.revconnect.com>.
- Sakaryan G, Wulff M and Unger H (2004) Search methods in P2P networks: a survey. In *Proceedings of I2CS-Innovative Internet Community Systems (I2CS 2004)*.
- Shareaza, <http://www.shareaza.com>.
- Si L and Callan J (2003) Relevant document distribution estimation method for resource selection. In *Proceedings of the 26<sup>th</sup> Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.
- Si L and Callan J (2004) The effect of database size distribution on resource selection algorithms. *Distributed Multimedia Information Retrieval*. LNCS 2924, Springer.
- Swapper.NET, <http://www.revolutionarystuff.com/swapper/>.
- Tsoumakos D and Roussopoulos N (2003b) A comparison of peer-to-peer search methods. In *Proceedings of the 6<sup>th</sup> International Workshop on the Web and Databases*.
- Yaga, <http://www.yaga.com>.
- Zhai C and Lafferty J (2001) A study of smoothing methods for language models applied to ad hoc information retrieval. *Research and Development in Information Retrieval*, pp. 334-342.