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

Jie Lu

Thesis Proposal

Language Technologies Institute
School of Computer Science
Carnegie Mellon University

# 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 provided very limited support for full-text search of document contents.

This proposal 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, existing approaches to federated search are adapted, and new methods are developed for resource representation, resource selection, and result merging 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 to facilitate effective and efficient full-text federated search in dynamic environments. Evaluation using the P2P testbed we developed demonstrates that our approaches to constructing and searching the P2P network provide a better combination of accuracy and efficiency than more common alternatives for federated search of text digital libraries in P2P networks. Solutions to additional problems for search in P2P networks such as load balancing and fault handling will be provided using a network administration model, and a P2P testbed of larger scale together with a prototype P2P application will be used for evaluation in more varied P2P environments as part of our effort to make full-text federated search in P2P networks a practical solution that can benefit real users.

# TABLE OF CONTENTS

# *Chapter 1*

# INTRODUCTION

A very large number of text digital libraries[1] 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 an unstructured text 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 contents provided by these digital libraries cannot be accessed by Web search engines such as Google and AltaVista that only conduct search on centralized repositories. How best to provide federated search[2] across such independent digital libraries is an unsolved problem often referred to as the "Hidden Web" problem.

In contrast to the text digital libraries on the Internet, a lot of distributed collections of text documents 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 in large scales. 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.

## 1.1 Motivation

To date, P2P networks are primarily used for file-sharing of popular music, videos, and software, or for distributed storage of digital archives. The types of digital objects in these systems have relatively obvious or well-known naming conventions and descriptions, making it convenient to represent them with just a few words from a name, title, or manual annotation. Search in these systems is typically *known-item search*, in which the goal is to find a single instance of a known object (e.g., a particular song by a particular artist). For known-item search, the user is familiar with the object being requested, and any copy is as good as any other. Known-item search of digital objects with well-known naming conventions is a task for which simple solutions suffice. For example, it is common in music file-sharing applications to use matches between query terms and file names or identifiers to determine which files can satisfy the information request.

To use P2P networks as a federated search layer for text digital libraries is a different scenario. First, text documents do not have well-known naming conventions and it is typically difficult to represent the content of a text document by using just a few words. Second, search is mostly no longer known-item search because the user is usually only having an information need in his/her mind instead of the identity of any particular document. The principal goal of search becomes locating documents that contain relevant contents to satisfy the information need, not finding a copy of a specific document. Therefore, more sophisticated solutions to search based on content are required.

---

[1] 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.

[2] Federated search provides a single interface to "support for 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 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 (Tsoumakos and Roussopoulos 2003b) (Sakaryan et al. 2004) (Li and Wu 2005). In contrast, it has already become common practice for a large number of 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 an unstructured text query, and the search result is presented with some form of relevance ranking ("*full-text ranked retrieval*"). We would argue that most of the recent research on P2P networks offers little useful guidance for providing full-text search of current text digital libraries. Thus we focus on developing solutions to full-text ranked retrieval for federated search of text digital libraries in P2P networks.

## 1.2 Challenges

Most existing search techniques developed for full-text ranked retrieval typically assume a centralized control. Either all the documents are stored in a centralized repository, or the directory information of all the documents is gathered at a centralized directory service. Traditional federated search ("distributed information retrieval") enables integrated search over distributed collections of documents by only requiring the aggregate directory information of 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 is undesired in P2P networks due to its susceptibility to become a performance bottleneck or the target of malicious attacks. 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 its dynamic nature. Due to frequent peer arrivals and departures, 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 peers rely on dynamic self-organization to adjust network structure, for effective and efficient full-text ranked retrieval, new approaches are needed to guide peer organization to achieve desired content distribution and network navigability.

## 1.3 Contributions

In this proposal, we extend previous notions of P2P networks to define a P2P *network overlay model* with desired content distribution and navigability, based on which we further develop a *network search model* to conduct effective and efficient full-text federated search of text digital libraries. A *network evolution model* is also proposed to describe how a P2P network using the defined network overlay model can evolve into one with desired search-enhancing properties dynamically and autonomously. Our network overlay model, network search model, and network evolution model provide an integrated framework for full-text federated search of text digital libraries with the promise of accuracy, efficiency, robustness, and scalability.

The network overlay model we propose in Chapter 3 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. 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 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, which is described in Chapter 4, fully 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 in consideration of new characteristics and requirements of federated search in P2P networks.

Both the network overlay model and the network search model would be less useful unless 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 required by the network search model. For this reason, we propose the network evolution model in Chapter 6. Our network evolution model works effectively with open-domain content 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, while putting extra effort into avoiding high system overhead on topology evolution and making the network scalable and robust.

By developing a P2P testbed which is one of the largest so far consisting of real-content text digital libraries, and using it to evaluate our models, we show our effort towards applying our approaches to real operational environments and verifying their effectiveness. In spite of this, we recognize that the models we propose are still a simplification of federated search in real P2P networks. To make our approaches to federated search in P2P networks more practical, we plan to tackle problems such as fault handling and load balancing using a *network administration model*. In addition, evaluation in more varied P2P environments is scheduled for the dissertation research. We discuss future work in Chapter 7.

# *Chapter 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.

The type of a P2P network is determined by three components that are essential to federated search, namely *network architecture*, *search mechanism,* and *network topology*. A *network architecture* defines the functionality and responsibility of each type of functional unit as well as the relations between peers with different types of functional units. A *search mechanism* describes the process of federated search with a set of protocols to specify how contents are represented and used for search ("*resource representation*"), how peers with relevant information can be located given an information request ("*resource location*"), and how search results from multiple peers are presented to the user ("*result presentation*"). A *network topology* specifies a particular instantiation of peer organization under a specific network architecture, which can be modeled as a graph with nodes representing peers and edges representing connections between peers.

In this chapter we begin by providing background knowledge in Section 2.1 about network architectures, search mechanisms and network topologies for federated search in P2P networks. Section 2.2 describes various search mechanisms developed for P2P networks based on different network architectures, both research-based and in popular use. Section 2.3 discusses related work on constructing network topologies with certain properties to enhance the performance of federated search.

In addition to existing work on federated search in P2P networks, previous research in distributed information retrieval has also developed solutions to full-text ranked retrieval of text digital libraries using a single, centralized directory service and a static structure (network topology). Because P2P networks can be viewed as a particular type of distributed information retrieval environment, we describe related approaches to federated search in distributed information retrieval in Section 2.4.

## 2.1 Basic Components of Federated Search in P2P Networks

The architecture, search mechanism and topology of a P2P network are closely related. The network architecture determines what search mechanisms and network topologies can be supported, and the network topology affects how efficient and effective any particular search mechanism can be carried out. In this section we present an overview of previously developed P2P network architectures, search mechanisms and network topologies in order to set the stage for the descriptions of different approaches used by existing P2P systems to federated search.

### 2.1.1 Network Architecture

Various P2P network architectures can be classified into four basic types of *brokered*, *completely decentralized*, *hierarchical*, and *structured* P2P architectures based on the functionalities of peers and the connections between them. Brokered, completely decentralized, and hierarchical P2P architectures are sometimes referred to as *unstructured* P2P architectures in order to contrast with structured P2P architectures.

**Brokered P2P architecture** A group of peers (possibly located in the same setting) function as a single, centralized logical directory service ("*broker*"). Other peers function as information providers and/or consumers. Information providers independently store their contents without relying on any system-wide resources and contact the centralized directory service to provide information about their contents. Information consumers contact the centralized directory service to locate providers with contents relevant to their requests and directly connect to the providers to download contents.

**Completely decentralized P2P architecture** All peers in the network provide the same functions. Each peer functions as a consumer, a provider and a directory service. Peers independently store their contents. Peers can connect with one another with minimal constraints. A completely decentralized P2P architecture is sometimes called a "*pure*" or "*flat*" P2P architecture.

**Hierarchical P2P architecture** Peers are typically organized into a two-level hierarchy of a lower level of leaves (providers and consumers) and an upper level of hubs (directory services).[3] Each information provider independently stores its contents. Each hub provides directory service to a region of the network and multiple hubs work collectively to cover the whole network. Leaves only connect to hubs. Hubs connect with leaves and other hubs. It is worth noting that a hierarchical structure is *not* necessarily a tree structure. Hierarchy here only refers to the division of peers into different classes.

**Structured P2P architecture** Each document (or document reference/pointer) is associated with a key identifier and each peer is associated with a peer identifier, taken from the same space as the keys (i.e., same number of digits). Documents or pointers to documents with a certain range of keys are distributed to each peer. Distributing documents to peers based on their keys implies that peers do not necessarily store their own contents and must store others' contents irrespective of their interests. Storing pointers to a set of documents in each peer's part of the key space adds an extra layer of indirection for resource location, but lowers the cost of being a directory service and allows documents to be only stored at the original peers that provide them. The connections between peers are determined by the locations of their identifiers in the key space. Because systems having structured P2P architectures are generally realized through Distributed Hash Table abstractions, they are often referred to as DHT-based systems.

### 2.1.2 Search Mechanism

For centralized search, the locations where the search results in response a query are generated are fixed and known to users since documents are stored in a centralized repository. For federated search in a P2P network, where the search results are generated can be different for different queries and unknown in advance because content dissemination is distributed. Therefore, locating the resources that are most likely to contain relevant documents becomes the main problem of federated search. Although blindly relaying queries using connections between peers in hope that peers with relevant documents could be reached ("*flooding*") is an admissible approach, a huge volume of network traffic will be generated for every single query, making it extremely inefficient and not scalable. For effective and efficient resource location, information about the contents covered by each resource needs to be discovered. Unstructured P2P architectures gather this information either at a centralized directory service, or at multiple regional directory services for query routing. Structured P2P architectures use this information to redistribute contents among peers so as to fix particular contents to specific locations. Once relevant resources are located, an additional problem of federated search is how to integrate the search results returned by multiple resources and present them to the user. In summary, three basic problems need to be addressed for federated search in a P2P network:

1. *Resource representation*: Discovering the contents covered by each resource;

2. *Resource location*: Locating resources most appropriate for an information need based on resource representations; and

[3] In theory, there can be multiple levels of hubs for a large-scale P2P network, with each level of hubs providing directory services to the peers at the next lower level.

3. *Result presentation*: Deciding how search results from multiple resources are presented to the user issuing the information request.

We briefly describe below existing common approaches for each of these problems.

**Resource representation** For federated search in P2P networks, the basic unit of information contents is a document. The commonly used representations of a document's content are i) terms from its name or title ("*name-based representation*"), ii) keywords that are automatically extracted from the (text) document or manually assigned to the document ("*keyword-based representation*"), iii) terms from a controlled vocabulary with a hierarchical structure based on an ontology ("*schema-based representation*"), and iv) all the terms that occur in the (text) document ("*full-text representation*"). For a full-text representation, the frequency information of how many times each term occurs in the document can be included. In this case, it is essentially a maximum likelihood unigram document language model.

Similar representations can be used to describe the contents of an information provider that contains multiple documents by ignoring document boundaries and treating the collection as one big document. Another way to represent a provider's contents is to associate terms from a controlled vocabulary or from the full collection vocabulary with the number of documents each term occurs in.

Name-based representations are simple and work well for audio, video, and software documents that have well-known naming conventions. Keyword-based and schema-based representations have small sizes and they can be used for both text and non-text documents. However, they require automatic or manual annotations of documents, which are often difficult and inaccurate for text documents that are long and have heterogeneous contents. Full-text representations provide a much more comprehensive description for text documents than other representations, but their sizes are significantly larger. In a few words, different representations are appropriate in different situations. There is no single representation that works best in all P2P environments.

When a resource representation is simply a list of terms without frequency information, it can be converted into numerical digits using hash functions for easy storage, transfer, and keyword matching (Gnutella v0.6) (Rohrs 2001) (Cuenca-Acuna and Nguyen 2002). For example, a *Bloom filter* is an array of bits commonly used to represent a set of terms. Multiple independent hash functions map each term into a set of indices and the bits at the corresponding indices of the array are set to 1. Whether a term appears in the representation can be tested quickly by applying the same set of hash functions to the term and checking whether the bits of the Bloom filter at the generated indices are set (Bloom 1970). Bloom filter provides a space-efficient data structure for set membership, but may have a small probability of false positives, i.e., claiming that a term is a member of the set while it actually isn't.

**Resource location** A network with a brokered P2P architecture uses a single, logical directory service for resource location. The directory service maintains a catalog of all the providers' contents in the network. In response to a request issued by a consumer, it provides a list of possible matches along with the addresses of the providers offering the matched content.

Resource location in both completely decentralized and hierarchical P2P networks rely on message passing between peers ("*query routing*"). The difference is that in a completely decentralized P2P network every peer is responsible for relaying the query message it receives to its neighbors (Gnutella v0.4), but in a hierarchical P2P network query routing is restricted to hubs (Gnutella v0.6). A peer can relay a query message to all of its neighbors ("*flooding*"), or to a subset of its neighbors selected according to certain criteria. Each message in the network has a time-to-live (TTL) field that determines the maximum number of hops it can be relayed in the network. The TTL is decreased by 1 each time the message is routed to a peer. When the TTL reaches 0, the message is no longer routed. Each peer discards duplicate messages it receives.

A structured P2P network uses distributed hash table lookup for resource location (Ratnasamy et al. 2002). Given a key, the task of resource location is to route the message to the peer responsible for that key (i.e., containing the document or a pointer to the document associated with that key). Each peer maintains a routing table consisting of a small subset of peers (neighbors) according to the network's distributed hash table architecture. When a peer receives a

query for a key for which it is not responsible, it routes the query to the neighboring peer that is "nearest" in terms of some distance between peer identifier and the identifier of the queried key.

**Result presentation** Although presenting search results in some form of relevance-based ranking is already common practice for search in a centralized repository (e.g., using Google), the search results provided by most P2P networks do not have relevance-based rankings. This is because document retrieval at an information provider in these networks typically uses Boolean keyword matching, which can only return a list of matched documents in response to a query with a simple ranking based on how often the keywords are matched in each document's representation, or other content-independent document features such as publishing dates. In contrast, full-text ranked retrieval can provide a relevance-based ranking of documents by using term frequency information and sophisticated term weighting schemes to estimate how well they satisfy a text query, but it requires a full-text representation of documents. Furthermore, if multiple ranked search results are returned by providers, merging them into a single, integrated relevance-based ranking requires more complex result merging techniques.

### 2.1.3 Network Topology

Under a specific network architecture, a network topology specifies a particular instantiation of peer organization in the logical protocol layer of the network. Because network topologies greatly affect where peers with relevant contents are located ("*content distribution*") as well as how easy it is to find them ("*navigability*"), even if the same search mechanism is used, network topologies with different properties may lead to different federated search performance.

Since a network topology can be modeled as a graph with nodes representing peers and edges representing connections between peers, the distance between peers in the graph induces one concept of peer distance, which we refer to as *graph distance*. Graph distance can be measured by the (shortest) path length (number of hops) from one peer to another in the graph. Besides graph distance, attributes of peers such as contents or locations in the underlying physical layer of the network can induce other concepts of peer distance, for example, *content distance* based on the similarity between peers' contents, or *latency distance* based on the latency in the underlying physical network. Certain characteristics of these concepts of peer distance can be used to describe some properties of network topologies that enable effective and efficient federated search. Below we discuss three search-enhancing properties of network topologies recognized by previous work, namely *interest-based locality*, *content-based locality*, and *small-world*.

**Interest-based locality** Peers in the network exhibit interest-based locality if a peer with a particular piece of content that one is interested in is very likely to have other pieces of content that one is also interested in (Sripanidkulchai et al. 2003). Interest-based locality has been explored in the context of Web browsing and search for problems such as content distribution, proxy positioning, server replication, and Web caching. By keeping contents closer to users (consumers) that are more likely to request them, user perceived latency as well as Web server load can be reduced (Krishnamurthy and Wang 2000). Interest-based locality can be utilized to improve the efficiency of federated search by keeping each peer's location near to those peers whose contents are similar to its interest. If we refer to the distance between two peers based on the similarity between one's interest and the other's content as *interest distance*, then interest-based locality means that peers with short graph distance are in short interest distance as well. This way queries do not need to travel far in order to locate relevant contents.

**Content-based locality** A network topology is said to exhibit content-based locality if peers with similar contents are located near to one another at the protocol layer. Content-based locality can be described as the high co-occurrence of short content distance and short graph distance between peers. Because documents relevant to a given query tend to be similar to one another, content-based locality makes locating most relevant contents efficient since they are mostly near to one another.

**Small-world** The small-world phenomenon refers to the phenomenon that any two individuals in the network are likely to be connected through a short sequence of intermediaries (Kleinberg 2003). Previous study has shown that the small-world phenomenon is common to many large-scale sparse networks in the real world, including the popular file-sharing P2P network Gnutella (Stutzbach and Rejaie 2005). We refer to these networks as *small-world networks*.

The topology of a small-world network ("*small-world topology*") has the properties of sparseness, short global separation (small diameter), and high local clustering of nodes (Watts and Strogatz 1998). These properties are achieved by sparsely connecting nodes that belong to different densely connected local clusters. Compared with a complete graph with the same number of nodes, a small-world topology has a much smaller number of edges (closer to $O(n)$ than to $O(n^2)$ where $n$ is the number of nodes in the graph). The diameter of a small-world topology increases logarithmically with the number of nodes, which indicates that there exist short paths between every pair of nodes even for a large-scale network. Compared with a random graph, a small-world topology has a much higher average probability of connecting two nodes given that both of them connect to the same third node ("*clustering coefficient*").

A small-world network topology in general has short graph distance between any pair of peers. The significance of having a small-world topology in the network is that it not only guarantees the existence of short paths between peers without requiring a large number of connections, but also provides the potential of using a decentralized algorithm with only local information to find these short paths (Kleinberg 2003). In contrast, in a P2P network with a random topology, although there exist short paths between peers, no decentralized algorithm is capable of finding them with a high probability. Therefore, navigating from source to target can be efficient for federated search in a P2P network with a small-world topology, but not in a P2P network with a random topology.

It is worth noting that a network with a small-world topology also exhibits good content-based locality if peers that form a local cluster have short content distance to each other. Therefore, when local clustering is based on content distance, content-based locality may be inferred from small-world properties.

## 2.2 Related Work on Search Mechanisms in P2P Networks

In this section we describe different search mechanisms developed for research or for real operational P2P systems, grouped by the architectures of the systems.

### 2.2.1 Search Mechanisms in Brokered P2P Networks

In a brokered P2P network, content dissemination is distributed, but search occurs in a centralized manner. The original Napster[4] was a representative brokered P2P network that used a name-based representation and Boolean keyword matching for music files. Each peer in the original Napster registered itself with a centralized server and provided a list of its shared files to the server. All requests were sent to the centralized server in the form of keywords. The centralized server responded to each request with a list of matched files along with the addresses of the providers offering these files. Consumers directly contacted the providers in the results to download files.

Centralized search ensures relatively consistent coverage and speed, but it suffers from a single point of failure. It also requires providers to be cooperative in providing accurate and detailed information about the contents they share. Furthermore, although in theory any resource representation can be used in brokered P2P networks, in practice typically only name-based representations are used due to their simplicity and efficiency, which limits the allowed retrieval models and result presentations. Despite these disadvantages, brokered P2P architectures are still in active use by today's P2P systems because they are easy to implement and control (Yaga) (MusicNet) (Intel 2003).

### 2.2.2 Search Mechanisms in Completely Decentralized P2P Networks

Gnutella v0.4 is an early example of a completely decentralized P2P architecture (Gnutella v0.4). Peers connect to one another with minimal constraints. Each peer offers a minimal directory service by blindly relaying each request it receives to all of its neighbors ("*flooding*", "*breadth-first search*") until the request has traveled a maximally allowed distance from the initiating peer. Responses are sent back along the query path in reverse direction. A consumer peer contacts directly the providers that have responded to its request to download the matched files.

[4] The original Napster refers to the Napster peer-to-peer music-sharing service that was started in 1999 and shut down in 2001.

A completely decentralized P2P network is simple to build and maintain. It easily reacts to the high dynamics of frequent peer arrivals and departures, which makes it quite robust. However, because query flooding with a limited search horizon is used for resource location and there is no special consideration in network topology, a large search radius is required to guarantee a high likelihood of locating relevant content, which leads to low efficiency and high overhead. The compromise between search accuracy and efficiency limits the self-scaling properties which motivate distributed P2P systems.

Recent research provides a variety of solutions to the flaws of the basic completely decentralized P2P architecture by enhancing the functionality of peers or by adding constraints to the connections between peers. Most approaches can be divided into three categories. The first category of approaches rely on *random walks* to reduce query traffic (Lv et al. 2002). The requesting peer sends out several query messages to a number of randomly chosen neighbors. Each of these messages follows its own path, having intermediate peers relay it to a randomly selected neighbor at each step. The relay of a message stops when it reaches a peer with relevant content or when the termination condition (e.g., based on TTL) has been satisfied. The performance of random walks is highly variable, depending on network topology and the random choices made. Although content replication can be used to increase the chance of locating relevant contents, it does not help much for requests of contents that are not so popular in the network.

The second category of approaches improve the performance of resource location by requiring each peer to select a subset of its neighbors based on certain criteria to further route query messages ("*directed breadth-first search*"). Various criteria use different knowledge or history about peers and their behaviors. For example, a peer's degree (its number of direct neighbors) is used as a criterion in (Adamic et al. 2001) to bias query routing towards high-degree peers, based on the intuition that a large number of peers can be reached quickly through high-degree peers and hence relevant content is likely to be located efficiently. Each peer can also select neighbors based on other statistics such as the number of results received through each neighbor for past queries, the latency of the connection with each neighbor, or the query load at each neighbor (Yang and García-Molina 2002). In Adaptive Probabilistic Search, the probability of choosing a neighboring peer depends on the successes and failures of previous searches that were routed through that peer (Tsoumakos and Roussopoulos 2003a). In (Kalogeraki et al. 2002), each peer uses the past query responses from its neighbors to build run-time profiles for them (essentially a keyword-based representation), which are used to select those neighbors that are most likely to reach content relevant to a new query by comparing the query with past queries in the profiles ("*intelligent search*"). Another approach to query-dependent selection is for each peer to record in its *routing index* the numbers of documents for a set of topics that may be found along the path led by each neighbor, and use the routing index to decide where to route query messages (Crespo and García-Molina 2002b). Query Routing Protocol proposes a similar approach as routing indices but a Bloom filter is used to summarize document keywords (Rohrs 2001). In PlanetP, each peer selects the peers to contact for a query based on the similarities between the query and the compact summaries it collects about other peers' inverted indices (Cuenca-Acuna and Nguyen 2002). Similarly, *local indices* requires each peer to maintain a local index of the contents of other peers that exist within a predetermined range and use it to select peers most likely to have contents relevant to the query (Yang and García-Molina 2002).

Approaches that belong to the third category construct network topologies with certain properties to facilitate more efficient and effective resource location. (Sripanidkulchai et al. 2003) proposes to use interest-based "shortcuts" based on the presence of interest-based locality to improve the efficiency of search in a completely decentralized P2P network. Each peer builds a shortcut list of peers that answered its previous queries. To find relevant content, a peer first queries peers on its shortcut list and turns to a default search mechanism such as flooding only if these peers cannot answer the query. In Crespo and García-Molina's work on semantic overlay network (SON), peers connect to other peers with similar contents to form a SON so that the network topology exhibits content-based locality. Queries are routed to the appropriate SONs, increasing the chances that matching files will be found quickly and reducing the search load on peers with unrelated contents (Crespo and García-Molina 2002a). Another similar method organizes peers into concept clusters based on a global ontology with the hypercube network topology (Schlosser et al. 2002). There is also research that explores small-world properties under completely decentralized P2P architectures. For instance, (Merugu et al. 2004) shows that using a small-world topology based on latency distance can improve the chances of locating files while decreasing the traffic load for file-sharing. (Sakaryan and Unger 2003) defines the content distance between two peers using the similarity between their content-based "informational profiles" and establishes peer connections based on small-world properties for efficient resource location.

Besides the popular name-based representations for music file-sharing P2P networks, other resource representations are also in active use by various completely decentralized P2P networks. The system described in (Kalogeraki et al. 2002) using "intelligent search" mechanism and routing indices (Crespo and García-Molina 2002) use keyword-based representations. Query Routing Protocol uses a keyword-based representation with a Bloom filter (Rohrs 2001). A full-text representation plus a Bloom filter is used in PlanetP (Cuenca-Acuna and Nguyen 2002). Semantic overlay networks (Crespo and García-Molina 2002a), the system of (Sakaryan and Unger 2003), and Hypercube P2P (Schlosser et al. 2002) adopt schema-based representations.

### 2.2.3 Search Mechanisms in Hierarchical P2P Networks

Hierarchical P2P architectures provide another approach to alleviate search overhead due to query flooding. Peers with more processing power and connection bandwidth provide distributed directory services for efficient and effective resource location without relying on a central authority. Each directory service (hub) maintains information about other hubs and providers that connect to it in order to direct query messages appropriately to those peers that are likely to provide or reach relevant contents, and shield the rest from irrelevant query traffic. This is similar to directed breadth-first search in completely decentralized P2P networks. The advantage of using a hierarchical P2P architecture is that resource location can be more efficient when it is conducted by peers with more computing and network resources so that peers that are limited in these resources won't become bottlenecks of federated search. Furthermore, instead of requiring digital libraries to cooperatively provide accurate descriptions of their contents, hierarchical P2P networks enable directory services to automatically discover the contents of (possibly uncooperative) digital libraries, which is well-matched to networks that are dynamic, heterogeneous, or protective of intellectual property.

The network topology of a hierarchical P2P network can also take advantage of the search-enhancing properties described in Section 2.1.3 to further improve federated search performance. For example, (Löser et al. 2003) suggests that in a schema-based hierarchical P2P network, using semantic overlay clusters to construct a network topology with content-based locality can enhance search efficiency and reduce flooding of the network with messages. Each semantic overlay cluster is defined as a link structure from a set of providers to a particular hub based on schema-based clustering policies. To the best of our knowledge, no work has been done to explicitly use interest-based locality and small-world properties in determining how hubs connect with leaf peers and how hubs interconnect with each other in hierarchical P2P networks.

Hierarchical P2P architectures avoid a single point of failure and thus are more robust than brokered P2P architectures, but the coordination between multiple directory services costs more overhead compared with using a single directory service. It easily supports sophisticated search techniques that are not constrained to representations using controlled or small vocabularies. However, hierarchical P2P networks typically have higher communication costs and are more complex than structured P2P systems. Existing protocols or applications using hierarchical P2P architectures include BearShare, Edutella, Gnucleus, Gnutella2, GUESS (Daswani and Fisk), JXTA, KaZaA, Limewire, Morpheus, Shareaza, and Swapper.NET.

### 2.2.4 Search Mechanisms in Structured P2P Networks

Various structured P2P systems (DHTs) can be distinguished by the mechanisms they use to generate document and peer identifiers and to perform distributed hash table lookup for resource location. The most well-known structured P2P systems include CAN (Ratnasamy et al. 2001), Chord (Stoica et al. 2001), Pastry (Rowstron and Druschel 2001), and Tapestry (Zhao et al. 2004). All of them associate each document (or document reference/pointer) with a key identifier and each peer with a peer identifier, and distribute documents (or pointers to documents) with a certain range of keys to each peer. Identifiers are strings of digits of some length. The search request for a document is specified by a key and routed through the network based on the routing information maintained at each peer about a subset of other peers (neighbors) until the peer responsible for that key is found. Different DHTs explore different key spaces, which lead to different measures for distance between identifiers, and therefore affect how a peer's neighbors are determined during topology construction as well as how to choose which neighbor to relay a query during search. Most of these systems have $O(\log n)$ neighbors for each peer and require $O(\log n)$ hops for resource location, except that CAN has $O(d)$ neighbors for each peer and needs longer paths of $O(dn^{1/d})$ hops, where $n$ is the number of peers in the network and $d$ is the dimension of the key space used by CAN (Ratnasamy et al. 2002). Additional examples of structured P2P

networks include CFS (Dabek et al. 2001), IRIS, (Maymounkov and Mazières 2002), eDonkey, eMule, RevConnect, pSearch (Tang et al. 2003), and Symphony (Manku et al. 2003).

Structured P2P networks are scalable and efficient, and they can be quite effective if documents are described using name-based or keyword-based representations. However, because it is difficult to adopt full-text representations in structured P2P architectures, full-text search cannot be easily supported. In addition, the strict content placement enforced by structured P2P architectures may limit peer autonomy.

## 2.3 Related Work on Network Topologies in P2P Networks

This section discusses related work on P2P networks constructing network topologies that have the search-enhancing properties described in Section 2.1.3.

### 2.3.1 Network Topologies with Interest-Based Locality

In the work described in (Sripanidkulchai et al. 2003), a loose topology structure on top of the existing topology of a completely decentralized P2P network is constructed to utilize interest-based locality. Each peer establishes "interest shortcuts" to other peers based on their responses to its earlier information requests. Interest-based locality is also used in (Shao and Wang 2005) to construct a BuddyNet topology structure on top of the topology of a hierarchical P2P network. The "buddies" of each peer are peers that have the highest probabilities of answering its future queries based on past query statistics. Because both approaches assume that a peer's interest is focused to some degree, if a peer has diverse interests on multiple topics, their constructed topologies may be less efficient since queries and connections are not distinguished by topics.

### 2.3.2 Network Topologies with Content-Based Locality

One approach to constructing a network topology with content-based locality is to cluster peers into explicitly defined content-based clusters and establish connections based on cluster memberships. For example, (Crespo and García-Molina 2002a) proposes to use a global classification hierarchy to cluster peers into one or more semantic overlay networks (SONs) and peers connect to other peers that belong to the same SONs in a completely decentralized P2P network. Another similar method partitions the contents in the network into concept clusters based on a global ontology and organizes peers using the hypercube network topology (Schlosser et al. 2002). In a hierarchical P2P network, each hub can be associated with a content-based cluster and providers that belong to a cluster connect to the corresponding hub. For instance, in (Löser et al. 2003), each hub in a schema-based hierarchical P2P network is associated with a semantic overlay cluster and matches an explicit clustering policy predefined by a human expert against the content model of a provider in order to decide independently whether to accept the provider into its cluster. Both the provider's content model and clustering policies are schema-based. The content model of each provider is broadcast to all the hubs in the network.

An alternative way to establish content-based locality is to use implicit content-based clusters, each of which is formed by peers with similar contents. For example, the algorithm proposed in (Khambatti et al. 2002) enables each peer in a completely decentralized P2P network to discover a sufficient number of other peers with similar contents by collecting the content information of its direct neighbors and their direct neighbors (one level of indirection). An algorithm that enables peers to self-organize into content-based clusters in a hierarchical P2P network is proposed in (Asvanund 2004). Starting from a randomly constructed network topology, each peer (hub or provider) actively seeks out new peers and evaluates its content-based similarity to them and replaces its existing neighbors with those peers that are more similar in content.

Using a global classification hierarchy or ontology to explicitly define content-based clusters assumes that the content space can be partitioned exhaustively into a number of content areas, and the annotations of document collections are available, which may be difficult to satisfy for text digital libraries containing heterogeneous, open-domain contents. Constructing a network topology with content-based locality by linking members of implicit content-based clusters does not restrict the contents in the network to be limited-domain. However, compared with explicitly defined content-based clusters, the discovery of implicit content-based clusters generally consumes more bandwidth, which may

become a burden for peers with limited connection bandwidth. In addition, the discovery may be slow when it relies on random walks to locate peers with similar contents.

### 2.3.3 Network Topologies with Small-World Properties

Given a concept of peer distance, a network topology with small-world properties can be constructed by connecting each peer to several peers in short distance ("*close*" peers) and a few peers in relatively long distance ("*remote*" peers). Different methods vary in how peers discover their close and remote peers, and how they determine which close and remote peers to connect to. Using Watts and Strogatz's "re-wired ring lattice" model, each peer is assumed to know its content distance to any other peer so that it chooses its $k$ closest peers for a small constant $k$ to establish *local* connections and randomly chooses from its remote peers with a uniform distribution for *long-range* connections (Watts and Strogatz 1998). Kleinberg argues that a network topology with a uniform distribution over remote peers for long-range connections does not provide sufficient latent navigational "cues" for a decentralized algorithm to find the short paths between peers using only local information (Kleinberg 2003). In his $d$-dimensional lattice model, the probability to establish a long-range connection between two peers is inversely related to their content distance using an "inverse $r^{th}$-power distribution". The resulting power-law distribution of connection lengths[5] provides the right mix of long-, medium-, and short-range connections in the topology for the decentralized algorithm to quickly navigate from source to target.

A network evolution algorithm that constructs a small-world topology adaptively using only local information at each peer is proposed in (Merugu et al. 2004). By selecting "better" neighbors (several closest peers and a few random peers) repeatedly according to its local view of the network, each peer adaptively moves to the appropriate location in the topology. Peers obtain their local knowledge of the network by measuring their latency distances with peers that are located within two hops. In the topology evolution algorithm described in (Sakaryan and Unger 2003), each peer obtains local knowledge about the network by analyzing previous search message chains and uses this information to update its local and long-range connections.

Another approach to constructing a small-world topology is to rewire existing connections in the network so that the topology can converge to one with a desired distribution of connection lengths. For example, inspired by surfers' behavior on the Web to update the outgoing links from their home pages, (Clauset and Christopher 2004) describes a rewiring process to update during each round the long-range connection of a random source peer if its path to a random destination peer is longer than a threshold. Based on the same finite-dimensional lattice model as used by Kleinberg, the paper claims that the network can converge to a topology with a power-law distribution of connection lengths from a range of initial distributions. The rewiring process proposed in (Manna and Kabakcioglu 2003) repeatedly selects a random pair of connections and rewires them to reduce the total length of the pair but with the constraint that the out-degree and in-degree of each peer are precisely maintained. The resulting network has a small diameter as well as high clustering and the distribution of connection lengths has a stretched exponential tail.

Among the algorithms described above, (Watts and Strogatz 1998) and (Kleinberg 2003) require global knowledge of the content distance from each peer to any other peer, which may be difficult to acquire in real P2P environments. (Merugu et al. 2004) and (Sakaryan and Unger 2003) select remote peers uniformly instead of using a power-law distribution, so the topology constructed using either algorithm does not guarantee a desired distribution of connection lengths in the network for good navigability. Although the rewiring processes proposed in (Clauset and Christopher 2004) and (Manna and Kabakcioglu 2003) can converge to a small-world topology with good navigability, the convergence may be quite slow for large-scale P2P networks.

## 2.4 Related Work on Full-Text Search of Text Digital Libraries

Prior research on full-text federated search of text digital libraries (also called "distributed information retrieval" in the research literature) identifies three problems that must be addressed:

[5] The length of a connection is defined as the content (or latency) distance between the two connected peers.

1. *Resource representation*: Discovering the contents covered by each digital library ("*resource description*");

2. *Resource selection*: Deciding which digital libraries are most appropriate for an information need based on their resource descriptions; and

3. *Result merging*: Merging ranked retrieval results from a set of selected digital libraries.

A single, centralized directory service is responsible for acquiring resource descriptions of the digital libraries it serves, selecting the appropriate digital libraries for a given query, and merging the retrieval results from selected digital libraries into a single, integrated ranked list. Therefore, federated search in traditional distributed information retrieval is essentially search in a brokered P2P network that has just one directory service. Solutions to all these three problems have been developed in distributed information retrieval. We briefly review them below.

### 2.4.1 Resource Representation

Different techniques for acquiring resource descriptions require different degrees of cooperation from digital libraries. STARTS is a cooperative protocol that requires every digital library to provide an accurate resource description to the directory service upon request (Gravano et al. 1997). STARTS is a good solution in environments where cooperation can be guaranteed. However, in some environments where digital libraries may not cooperate or may have an incentive to cheat (*uncooperative environments*), STARTS cannot be used to acquire accurate resource descriptions.

Query-based sampling is an alternative approach to acquiring resource descriptions without requiring explicit cooperation from digital libraries. The resource description of a digital library is constructed by sampling its documents via the normal process of submitting queries and retrieving documents. Query-based sampling has been shown to acquire fairly accurate resource descriptions using a small number of queries and documents in distributed information retrieval environments (Callan and Connell 2001).

The typical format of a resource description includes a list of terms with corresponding collection term frequencies ("*collection language model*"), and corpus statistics such as the total number of terms and documents in the collection. Capture-Recapture (Liu at al. 2002) and Sample-Resample (Si and Callan 2003a) are two methods of estimating the total number of documents of an uncooperative digital library. Experimental results show that in most scenarios, Sample-Resample is more accurate and has less communication costs than the Capture-Recapture method (Si and Callan 2003a).

### 2.4.2 Resource Selection

Resource selection aims to select a small set of resources that contain many documents relevant to the information request. Typically resource selection includes *resource ranking* which ranks resources by their likelihood of returning relevant documents, and *thresholding for resource selection* which selects the top-ranked resources to process the information request.

Resource selection algorithms such as CORI (Callan 2000), gGlOSS (Gravano et al. 1994) (Gravano and García-Molina 1995), and Kullback-Leibler (K-L) divergence-based (Xu and Croft 1999) algorithms use techniques adapted from document retrieval for resource ranking. These algorithms have been shown to work well with resource descriptions provided by cooperative digital libraries or acquired using query-based sampling.

Other resource selection algorithms including ReDDE (Si and Callan 2003a) and DTF (the decision-theoretic framework for resource selection) (Nottelmann and Fuhr 2003) rank resources by directly estimating the number of relevant documents from each resource for a given query. ReDDE relies on sampled documents obtained using query-based sampling for such estimation. DTF has three variants DTF-rp, DTF-sample and DTF-normal. DTF-rp estimates the number of relevant documents from a resource by assuming a linearly decreasing recall-precision function and calculating the expected precision and recall from the resource. DTF-sample uses sampled documents to estimate how relevant documents are distributed among the available resources. DTF-normal models the distribution of document

scores from a resource with normal distribution and map document scores to probability of relevance using a function learned with user relevance feedback.

Deciding how many top-ranked resources to be selected ("*thresholding for resource selection*") is a problem that is usually simplified. Most resource selection algorithms use heuristic values such as 10 and 20 for the number of selected resources.

### 2.4.3 Result Merging

Many result-merging algorithms have been proposed in distributed information retrieval. One approach is based on normalizing resource-specific document scores into resource-independent document scores. The CORI merging algorithm uses a heuristic linear combination of digital library scores and document scores to normalize the scores of the documents from different digital libraries. The intuition is to favor documents from digital libraries with high scores and also to enable high-scoring documents from low-scoring digital libraries to be ranked highly. It is effective when used together with the CORI resource ranking and INQUERY document retrieval algorithms (Callan 2000). The Semi-Supervised Learning result-merging algorithm uses the documents obtained by query-based sampling as training data to learn score normalizing functions on a query-by-query basis. It is shown to work well with a variety of resource selection and document retrieval algorithms and is the current state-of-the-art for result merging in distributed information retrieval (Si and Callan 2003b). There has been some work on using logistic regression to learn merging models to normalize document scores but relevance judgments are required for training (Le Calv and Savoy 2000).

Another approach to result merging is recalculating document scores at the directory service. Document scores can be recalculated at the directory service by downloading all the documents in the retrieval results from selected resources, indexing them, and re-ranking them using a document retrieval algorithm. Downloading documents is not necessary if all the statistics required for score recalculation can be obtained in another way. Kirsch's algorithm (Kirsch 1997) requires each resource to provide summary statistics for each of the retrieved documents. It allows very accurate normalized document scores to be determined without the high communication cost of downloading. The corpus statistics required for recalculating document scores could also be substituted by a reference statistics database containing all the relevant statistics for some set of documents. This method is explored in (Craswell et al. 1999) and shown to be effective compared with using the corpus statistics provided by cooperative digital libraries.

## 2.5 Summary

This chapter first provides background knowledge on the basic components of federated search in P2P networks, namely 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. A search mechanism specifies the activities required for search, which mainly includes representing contents, locating relevant resources, and presenting results. A network topology describes how peers are connected in the network. It affects the effectiveness and efficiency of any particular search mechanism.

Following the overview of the basic components of federated search, previously developed approaches to each component are reviewed, and their strengths and weaknesses for search in different P2P environments are pointed out. 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, which is described in later chapters.

The development of our search mechanism for full-text federated search in P2P networks also benefits from previous research in traditional distributed information retrieval on full-text ranked retrieval using a single, centralized directory service. Viewing full-text federated search in P2P networks as distributed information retrieval in a particular type of environment, 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 new characteristics and requirements of full-text federated search in P2P networks.

# *Chapter 3*

# NETWORK OVERLAY MODEL

A *network overlay* model describes the 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 to work collectively to cover the network without relying on a central authority. Therefore, it is more robust and scalable than a brokered P2P architecture with a single, centralized directory service. Second, because full-text search requires a full-text representation, which is more expensive in terms of storage and communication costs, peers need more processing power and connection bandwidth to perform the duties of directory services. Hierarchical P2P architectures relieve peers with limited computing and network resources of the burden of conducting directory services, which makes them a better choice than completely decentralized P2P architectures. Third, it is difficult to use full-text representations with term frequency information in structured P2P architectures due to their complete reliance on distributed hash tables for content placement and resource location. In contrast, existing techniques developed for full-text federated search can be adapted to hierarchical P2P architectures in a straightforward manner.

In this chapter, we present the network overlay model, based on which our approaches to full-text federated search are developed.

## 3.1 Network Architecture

In this section, we describe in detail the extended hierarchical P2P architecture designed for full-text federated search, emphasizing the enhanced functionality of each type of functional unit (consumer, provider, directory service), as well as the new characteristics of network connections.

### 3.1.1 Functional Units

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. As is common in most operational and research P2P systems, peers are assumed to be honest and cooperative.[6]

A *consumer* represents a user with information requests. It initiates the search process by generating an unstructured text query message (which may include user and system settings such as the number of returned documents and the search radius) and relaying the message to the hubs it connects to, and finalizes the search process by collecting the returned results and presenting them to the user.

A *provider* is a text 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 matched documents in response to an unstructured text query. For document retrieval algorithms that support relevance-based document rankings, documents are ranked by how well they satisfy the query and the response is a list of the top-ranked documents. In addition to responding to incoming queries, a provider also provides an accurate description of its content to its neighboring hubs upon request.

[6] The general framework described in this proposal for full-text federated search also applies to environments where digital libraries (information providers) are not cooperative or have an incentive to cheat, although different approaches are required for acquiring resource descriptions and result merging (Lu and Callan 2005).

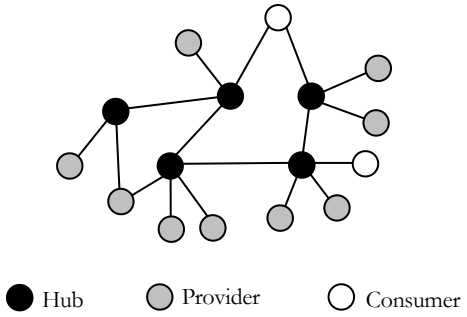


**Figure 3.1 Illustration of a hierarchical P2P network.**

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 leaves, and uses it to provide resource selection (query routing) and result merging (integrating results returned by multiple peers) services to the network.

### 3.1.2 Connections

Connections in an information-sharing P2P network are data channels established at the protocol layer through which information is exchanged in the form of messages. These logical connections are not necessarily associated with the underlying physical connections in the network. In this proposal, by default "connections" refers to the logical connections at the protocol layer used to exchange messages between peers.

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 find a setting in terms of where to connect and connection capacity that can optimize the utilities of all functional units at the same time. 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.[7] 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 the hierarchical P2P architecture, leaves (providers and consumers) only connect to hubs. Hubs connect with leaves and other hubs. This way each hub acts as a gateway between its connecting leaves and the rest of the network, so leaves with limited connection bandwidth are relieved of the responsibility to relay messages that are not related to their contents or interests. Figure 3.1 illustrates a hierarchical P2P network.

## 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-consumer topology, and hub-hub topology, each of which serves different purposes and desires different search-enhancing properties. In this section we describe our network topology in terms of its properties for different components and why they can support effective and efficient full-text federated search.

[7] 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.

### 3.2.1 Hub-Provider Topology with Content-Based Locality

A randomly generated hub-provider topology produces an arbitrary content distribution at each hub. In this case, queries must be routed to hubs all over the network in order to locate enough relevant contents to provide a desired recall. In contrast, if hub-provider topology exhibits content-based locality by connecting providers with similar contents to the same hub to form a content-based cluster (i.e., by requiring each hub to cover a specific *content area*), then most contents relevant to a query are expected to be covered by a few hubs so that query routing can be both efficient and effective.

### 3.2.2 Hub-Consumer Topology with Topic Diversity and Interest-Based Locality

A consumer can perform two different types of search activities: *ad-hoc search* and *focused search*. For ad-hoc search, successive information requests from a consumer are not closely related conceptually. For focused search, information requests during a period of time are closely related to one another, showing the consumer's persistent interests in specific topics. If a consumer mainly performs ad-hoc search, because there is no steady, homogenous interest model, there is no single "best" hub that it can connect to in order to optimize search performance at all times. The best it can do is to connect to hubs that can reach diverse content areas in short graph distances ("*topic diversity*"). In contrast, if a consumer mainly performs focused search during a period of time, the best hubs it can connect to are those that cover content areas most similar to the consumer's interests. In this case, hub-consumer topology exhibits interest-based locality. If a consumer equally needs both ad-hoc search and focused search, then it may need to maintain separate sets of hub connections for queries of different purposes and topics. This way it can forward each of its queries along selective hub-consumer connections (with possibly different search radiuses) for optimal search performance.

### 3.2.3 Hub-Hub Topology with Content-Based Small-World Properties

Federated search in a hierarchical P2P network relies on message-passing using hub-hub connections to first locate hubs that cover relevant contents before these hubs further direct messages to the connecting providers. If hubs covering similar content areas are located 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 is also indispensable in assuring that a query can be routed to the targeted content region efficiently irrespective of where it starts. In a few words, 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. Therefore, we refer to these properties as *content-based small-world properties* at the hub level.

## 3.3 Summary

In this chapter, we describe a network overlay model for full-text federated search of text digital libraries. By enhancing the functionalities of information consumers (users), information providers (digital libraries), and hubs (directory services) of the basic hierarchical P2P architecture, our network architecture explicitly supports full-text search over document contents to generate relevance-based rankings of documents in response to unstructured text queries. In addition, it is the first to recognize the distinctions between network connections that link different types of functional units (i.e., consumers, providers, and hubs) or serve different search purposes (i.e., ad-hoc search and focused search of different topics), and their importance in achieving an optimal setting for the utilities of all functional units simultaneously.

Under the defined hierarchical P2P architecture, desirable properties of a network topology for efficient and effective full-text federated search are described. How a network can dynamically evolve into one with these properties is detailed in Chapter 6. Although the search-enhancing properties of interest-based locality, content-based locality, and small-world properties and their uses in P2P networks are not our new development, our network overlay model is unique in effectively incorporating all these properties in a single framework to support full-text federated search.

*Chapter 4*

# NETWORK SEARCH MODEL

Search is the central activity in information-sharing peer-to-peer networks. Due to the lack of central authority to provide global coordination among peers, federated search in hierarchical P2P networks relies on local coordination, which is typically less efficient than using a centralized control. Therefore, compared with search using a centralized index or a central directory service, decentralized search is more concerned with the trade-off between accuracy and efficiency. In this chapter, we define a network search model to describe a full-text federated search mechanism in a hierarchical P2P network (with the network overlay model described in Chapter 3) targeted at offering a better combination of accuracy and efficiency than existing common approaches for federated search of text digital libraries in hierarchical P2P networks.

## 4.1 Overview of Full-Text Federated Search

The quality of federated search in P2P networks is measured not only by its accuracy, but also by its efficiency. Search mechanisms commonly used in file-sharing P2P applications such as flooding and random walks cannot be efficient and effective at the same time. The flooding technique guarantees to reach peers with relevant information but requires an exponential number of query messages; randomly forwarding the request to a small subset of neighbors can significantly reduce the number of query messages, but the reached peers may not be relevant. Directed breadth-first search has been shown to be a promising approach in providing a better combination of accuracy and efficiency (Yang and García-Molina 2002) (Kalogeraki et al. 2002) (Crespo and García-Molina 2002b), but no previous methods of using directed breadth-first search in P2P networks have been targeted at full-text ranked retrieval.

Because P2P networks can be viewed as a particular type of distributed information retrieval environment, we seek inspirations from previous research on full-text federated search in distributed information retrieval. Since resource selection techniques developed for a single directory service essentially conduct a one-level directed breadth-first search for the directory service to select digital libraries based on their resource descriptions, we can apply them to hub-provider query routing in hierarchical P2P networks so that query messages are only relayed to providers that are most likely to generate relevant responses. The efficiency of federated search can be further improved if we extend resource selection techniques to hub-hub query routing so as to propagate queries only to those network regions that cover related content areas. In addition to resource selection, existing result merging techniques can also be extended in hierarchical P2P networks to provide search results with integrated relevance-based document rankings.

Although the development of our search mechanism can start by extending some existing approaches to resource representation, resource selection, and result merging, new development is still in demand to fit the solutions to the unique characteristics of the hierarchical P2P network defined by our network overlay model. For example, selection of a neighboring hub should be based on not only this hub's likelihood of providing relevant documents with its own providers, but also its potential to provide a path to peers that are likely to satisfy the information request since the query message routed to it may further travel multiple hops. Thus a new representation for the contents covered by the available resources in the network is needed. Peer autonomy and lack of central control also pose new challenges to resource selection and result merging at regional directory services.

Our search mechanism does not adopt the decentralized search algorithm proposed by Kleinberg for efficient navigation in a P2P network (Kleinberg 2003), because Kleinberg's search algorithm assumes that given a query, the location of the target peer that has relevant content is known, and the task of search is quickly finding a path that leads from source to target. However, for full-text search in a real P2P network, until a real-time search is performed, the location of relevant content typically remains unknown to the requester. Therefore the graph distance from the source or any other peer to the target, which is required by the algorithm for navigation, cannot be directly measured. As a result, our search mechanism only relies on local content information for navigation to accommodate the network's decentralized, dynamic and uncertain nature.

Before presenting solutions to the problems of resource representation, resource selection (resource ranking and thresholding for resource selection), and result merging, we briefly describe in the rest of this section how our search mechanism works for full-text federated search in P2P networks. When a consumer has an information request, it sends a query message with an initial TTL value to selected neighboring hubs based on the purpose of the search (i.e., ad-hoc search or focused 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 if the message's TTL hasn't reached 0. 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 to include a list of the top-ranked documents. A hub is responsible for collecting the queryhit messages generated by multiple 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.

## 4.2 Resource Representation

In order for resource selection to determine which resources are more likely to contain documents relevant to the query, a resource description should include term frequency information in its full-text representation. Therefore, we adopt the format of a resource description used by previous resource selection algorithms (Callan 2000) (Gravano et al. 1994) (Gravano and García-Molina 1995) (Xu and Croft 1999) in distributed information retrieval, which 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 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. Although resource descriptions for different types of resources have the same format, different methods are required to acquire them, which we introduce below.

### 4.2.1 Resource Descriptions of Providers

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).

### 4.2.2 Resource Descriptions of Hubs

The resource description of a hub is the aggregation of the resource descriptions of its neighboring providers. It describes the content area covered at 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 neighbor. This kind of information is referred to as the resource description of a *neighborhood* and is introduced in the following subsection.

### 4.2.3 Resource Descriptions of Neighborhoods

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 4.1 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.

Resource descriptions of neighborhoods provide similar functionality as routing indices (Crespo and García-Molina 2002b). An entry in a routing index records the number of documents that may be found along a path for a set of topics represented by a small set of topic keywords. The key difference between resource descriptions of neighborhoods and routing indices is that resource descriptions of neighborhoods represent contents with unigram language models (terms

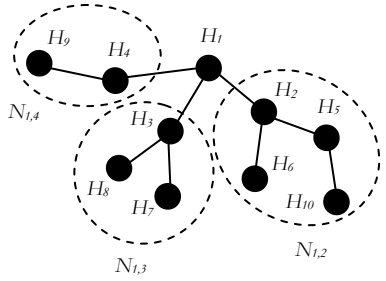


**Figure 4.1 3 neighborhoods that can be reached from $H_1$.**

with their frequencies), while routing indices represent them with a set of keywords for various topics. Thus by using resource descriptions of neighborhoods, there is no need for hubs and providers to cluster their documents into a set of topics and it is not necessary to restrict queries to topic keywords.

Similar to exponentially aggregated routing indices (Crespo and García-Molina 2002b), 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:

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

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.

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

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.

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

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

### 4.2.4 Reducing Sizes of Resource Descriptions

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.

The pruning method of cutting off the rarest terms in resource descriptions has been explored in our earlier work for full-text federated search in hierarchical P2P networks of text digital libraries (Lu and Callan 2003a). Experimental results demonstrate that on average it can reduce the size of a resource description by half without detriment to search accuracy. Therefore, pruning terms of low frequencies in resource descriptions is a simple but effective technique.

Pruning terms that are common in general English is equivalent to first extracting the core collection language model which gives the probability of each term being generated by the content of the collection rather than by general English, and then discarding terms with low probabilities. Given the maximum likelihood collection language model and the general English model, the core collection language model can be estimated using the method described in (Zhang, Y et al. 2002). This method is effective, but quite complex and expensive to use in practice for resource descriptions of large vocabularies.

In this proposal, we refer to the resource descriptions before pruning as *full* resource descriptions and those after pruning as *pruned* resource descriptions.

## 4.3 Resource Ranking

Resource ranking is the first step of resource selection for query routing. 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 path to peers with relevant information, and only forwards the request to the top-ranked neighbors. The information each hub utilizes for resource ranking is the content information about its neighboring providers as well as neighborhoods represented by resource descriptions.

Because resource descriptions of providers and those of neighborhoods are not of the same magnitude in vocabulary size and term frequency, direct comparison between providers and neighborhoods is difficult since it requires better size normalization. For this reason, a hub handles separately the selection of its neighboring provider peers and hubs. In theory, each hub can choose its own resource ranking 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 ranking algorithms tested on various testbeds in distributed information retrieval (Si and Callan, 2004), we use it for ranking of both neighboring providers and neighboring hubs at each hub.

### 4.3.1 Resource Ranking of Providers

Each hub uses the K-L divergence resource ranking algorithm to calculate $P(P_i \mid Q)$, the conditional probability of predicting the collection of provider $P_i$ given the query $Q$ and uses it to rank different providers. The prior probability of a provider $P(P_i)$ is set to be proportional to the total number of documents in the collection of this provider. $P(P_i \mid Q)$ is calculated as follows:

$$\mathrm{P}(P_i \mid Q) = \frac{\mathrm{P}(Q \mid P_i) \times \mathrm{P}(P_i)}{P(Q)} \propto \mathrm{P}(Q \mid P_i) \times numdocs(P_i) \tag{4.4}$$

$$\mathrm{P}(Q \mid P_i) = \prod_{q \in Q} \frac{tf(q, P_i) + \mu \times \mathrm{P}(q \mid G)}{numterms(P_i) + \mu} \tag{4.5}$$

where $tf(q \mid P_i)$ is the term frequency of query term $q$ in provider $P_i$'s resource description (collection language model), $\mathrm{P}(q \mid G)$ is the background language model used for smoothing and $\mu$ is the smoothing parameter in Dirichlet smoothing.

### 4.3.2 Resource Ranking of Hubs

For ranking 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 ranking algorithm. The prior probability of a neighborhood $\mathrm{P}(N_i)$ is set to be proportional to the exponentially aggregated total number of documents in the neighborhood. Given the query $Q$, the probability of predicting the neighborhood $N_i$ that a neighboring hub $H_i$ represents is calculated as follows and used to rank neighboring hubs:

$$\mathrm{P}(N_i \mid Q) = \frac{\mathrm{P}(Q \mid N_i) \times \mathrm{P}(N_i)}{P(Q)} \propto \mathrm{P}(Q \mid N_i) \times numdocs(N_i) \tag{4.6}$$

$$\mathrm{P}(Q \mid N_i) = \prod_{q \in Q} \frac{tf(q, N_i) + \mu \times \mathrm{P}(q \mid G)}{numterms(N_i) + \mu} \tag{4.7}$$

where $tf(q \mid N_i)$ is the term frequency of query term $q$ in the resource description of neighborhood $N_i$ (collection language model), $\mathrm{P}(q \mid G)$ is the background language model used for smoothing and $\mu$ is the smoothing parameter in Dirichlet smoothing.

## 4.4 Thresholding for Resource Selection

Thresholding for resource selection in a hierarchical P2P network is the process for a hub to decide how many top-ranked neighboring providers or hubs to select for relaying the 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 based on a fixed threshold*. The value of this fixed threshold is tuned empirically to optimize system performance. In a hierarchical P2P network, there are many hubs and providers, and their numbers are 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 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 neighbors whose relevance-based ranking scores are larger than a learned threshold as *resource selection based on a learned threshold.*

The problem of learning a threshold to convert relevance-based ranking scores into a binary decision has mostly been studied in information filtering and text categorization (Zhai et al.1998) (Zhai et al. 2000) (Zhang and Callan 2001). However, the user relevance feedback required as training data may not be as easily available for federated search in P2P networks as for the task of information filtering. Therefore, it is preferable that threshold learning in P2P networks be conducted in an unsupervised manner. Our goal is to develop a technique for each hub to learn resource selection thresholds 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 4.5), as long as result merging has reasonably good performance, we could assume that the top-ranked merged documents are

"relevant". Thus the distribution of the top-ranked merged documents over neighboring peers should provide useful hints on the number of relevant documents each neighbor is likely to return. If we further assume that a hub is permitted to flood its neighbors with a small number of queries, it can use the results of these queries as training data. This is analogous to query expansion with pseudo-relevance feedback which treats the top-ranked documents retrieved initially as relevant documents and uses them to improve the quality of the query. The key differences are i) our approach uses the information about which top-ranked merged documents are from which neighbors and ignores the actual contents of these documents, and ii) the direct goal here is not to improve immediately the retrieval quality for the current query, but to learn a resource selection threshold that is specific to each hub and sometimes even specific to different types of queries and improve the overall search performance for a set of queries in the future.

Each hub learns two resource selection thresholds of relevance-based ranking scores, one for selecting its neighboring providers, the other for selecting its neighboring hubs. A hub can compute its resource selection threshold for neighboring providers or for neighboring hubs 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 neighbors 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 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 neighbors 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.

We introduce in Section 4.4.1 individual-based threshold learning using normalized ranking scores. Set-based threshold learning using original (unnormalized) ranking scores, and set-based threshold learning using normalized ranking scores are introduced in Section 4.4.2. Section 4.4.3 describes our attempt to take advantage of the complementary strengths of the three methods by using a hybrid approach.

### 4.4.1 Individual-Based Threshold Learning with Normalized Ranking Scores

For a training query, after a hub merges the documents returned by its neighbors of a particular type (e.g., providers), it can calculate how many "relevant" documents are returned by each neighbor by using the top-ranked documents in the merged result as the set of "relevant" documents with respect to the query. Given this information, one simple method to decide the threshold of ranking scores for the query is to go down the list of neighbors sorted by their normalized ranking scores until a sufficiently large percentage of "relevant" documents have been returned, i.e., a sufficiently high value of recall is obtained and use the last ranking score before stopping as the threshold. However, because this method doesn't measure the amount of "non-relevant" documents returned by the top-ranked neighbors, the learned threshold will not work well when the top-ranked neighbors that return some "relevant" documents return even more "non-relevant" documents, in which case a high value of recall comes together with a low value of precision. To balance between recall and precision, it is more appropriate to use a measure to combine recall and precision such as the *E* evaluation measure proposed by van Rijsbergen (van Rijsbergen, 1979). The *E* measure is defined under this circumstance as follows:

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

where $R(j)$ is the recall for documents returned by neighbors ranked 1st to $j^{th}$, $P(j)$ is the precision for documents returned by neighbors ranked 1st to $j^{th}$, $E(j)$ is the E evaluation measure relative to $R(j)$ and $P(j)$, and $b$ is a parameter which reflects the relative importance of recall and precision. Values of $b$ greater than 1 indicate that precision is valued more than recall while values of $b$ smaller than 1 indicate that recall is valued more than precision. To decide the threshold of ranking scores for the query, the hub goes down the list of neighbors sorted by their normalized ranking scores, stops at the neighbor that has the maximum $E$ value and uses its ranking score as the threshold.

To summarize, 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 (hubs) with respect to a query:

1. Given a query, the hub uses its resource ranking algorithm to calculate the ranking scores of its neighboring providers (hubs) and sorts them in descending order;

2. The hub normalizes their scores using the following equation:

$$S' = \frac{S - S_{\min}}{S_{\max} - S_{\min}} \tag{4.9}$$

   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 (hubs) and merges the lists of documents returned by these neighbors;[8]

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 rank $j$, where $r$ is a parameter of threshold learning (50 in our experiments); and

5. The hub finds the rank $j^*$ that gives the maximum $E$ value and regards the normalized ranking score of the $j^{*th}$ neighbor as the threshold for selection of neighboring providers (hubs) 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.

### 4.4.2 Set-Based Threshold Learning with Original or Normalized Ranking Scores

Set-based threshold learning takes a similar approach as using maximum likelihood estimation to learn dissemination threshold for information filtering (Zhang and Callan 2001). For information filtering, an optimal dissemination threshold is one that maximizes a given utility function based on the distributions of the scores of relevant and non-relevant documents. For our task of resource selection, an optimal selection threshold is the one that maximizes a given utility function based on the distributions of the ranking scores of a hub's relevant and non-relevant neighbors. To take this approach, we need to solve three problems: i) Defining a utility function; ii) Determining the criterion for a peer to be considered relevant with respect to a query; and iii) Deciding how to estimate the distributions of the ranking scores of relevant and non-relevant peers.

[8] A neighboring hub needs to collect all the documents returned by its downstream peers, not just the documents returned by its own providers. This is also the case for set-based threshold learning described in the next section.

A linear utility function $U(\theta)$ is defined as below, and the optimal value $\theta^*$ that maximizes $U(\theta)$ at a hub can be used as this hub's threshold for selection of its provider (hub) neighbors:

$$U(\theta) = N_{rel}(\theta) - N_{nonrel}(\theta) \tag{4.10}$$

$$\theta^* = \arg\max_{\theta} U(\theta) = \arg\max_{\theta} \{N_{rel}(\theta) - N_{nonrel}(\theta)\} \tag{4.11}$$

$$N_{rel}(\theta) = \alpha \times \int_{\theta}^{\max(s)} \mathrm{P}(s \wedge rel) ds = \alpha \times \int_{\theta}^{\max(s)} \mathrm{P}(s \mid rel) \times \mathrm{P}(rel) ds \tag{4.12}$$

$$N_{nonrel}(\theta) = \alpha \times \int_{\theta}^{\max(s)} \mathrm{P}(s \wedge nonrel) ds = \alpha \times \int_{\theta}^{\max(s)} \mathrm{P}(s \mid nonrel) \times (1 - \mathrm{P}(rel)) ds \tag{4.13}$$

where $N_{rel}(\theta)$ and $N_{nonrel}(\theta)$ are the numbers of relevant and non-relevant neighbors respectively whose ranking scores are above threshold $\theta$, $\mathrm{P}(s \wedge rel)$ is the probability of a neighbor having score $s$ and being relevant, $\mathrm{P}(s \wedge nonrel)$ is the probability of a neighbor having score $s$ and being non-relevant, $\mathrm{P}(s \mid rel)$ is the probability of a relevant neighbor having score $s$, $\mathrm{P}(s \mid nonrel)$ is the probability of a non-relevant neighbor having score $s$, $\mathrm{P}(rel)$ is the probability of a neighbor being relevant, $\alpha$ is the total number of neighbors, and $\max(s)$ is the maximum relevance-based ranking score of a hub's neighbors for the training queries. The integrals in Equations 4.12 and 4.13 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.

For a given query, a peer is considered relevant if it returns at least $n$ relevant documents. When real relevance judgments are not available due to lack of user relevance feedback, a hub can estimate the relevance of a neighboring peer with respect to the query using the top-ranked merged documents at this hub for the query as the set of "relevant" documents. $n$ is a parameter of set-based threshold learning.

The difference between set-based threshold learning using original ranking scores and using normalized ranking scores lies in their different ways to estimate the distributions of the ranking scores of relevant and non-relevant neighbors $\mathrm{P}(s \mid rel)$ and $\mathrm{P}(s \mid nonrel)$ at a hub, which we describe in the following paragraphs. Briefly speaking, using original ranking scores requires maximum likelihood fittings of continuous Gaussian models for different groups of training queries, while using normalized ranking scores estimates empirical discrete distributions for a single group containing all training queries.

Using original ranking scores, because the score range is query-dependent which may indicate a hub's neighbors' overall degree of relevance (total amounts of relevant documents available) for the query, the hub needs to divide training queries into groups based on its neighbors' different levels of overall degree of relevance for these queries and estimate $\mathrm{P}(s \mid rel)$ and $\mathrm{P}(s \mid nonrel)$ for each group. Queries can be classified based on their contents or statistical properties. Classifying queries by content is more difficult because it requires more training data and hence imposes higher system overhead. Therefore, we focus on classifying queries by their statistical properties. Because the average probability of a query's terms in the aggregate resource description at the hub is a rough measure of the hub's neighbors' overall degree of relevance for the query, we use it as a feature for query classification. Given a set of training queries, probability values ranging from 0 to the maximum term probability in the aggregate resource description are divided into non-overlapping groups so that all groups have roughly the same number of queries for training. A query is classified into one of these groups based on the average probability of its terms in the aggregate resource description. For each group of training queries, the empirical distributions of $\mathrm{P}(s \mid rel)$ and $\mathrm{P}(s \mid nonrel)$ can be fitted using Gaussian distributions.

Because normalized ranking scores are somewhat "query-independent", there is no need to classify training queries into different groups. A single pair of $\mathrm{P}(s \mid rel)$ and $\mathrm{P}(s \mid nonrel)$ can be estimated from the aggregate results of all training queries. However, unlike the case with original ranking scores, score distributions of $\mathrm{P}(s \mid rel)$ and $\mathrm{P}(s \mid nonrel)$ cannot

be fitted by Gaussian or exponential distributions. For this reason, instead of fitting continuous distributions to the training data, the hub directly uses the empirical discrete score distributions learned from training queries.

Usually P(*rel*) is estimated by maximum likelihood estimation using training data. However, because using the top-ranked merged documents as the set of "relevant" documents for each query yields very unbalanced amounts of training data for relevant and non-relevant neighbors at the hub (very few relevant neighbors but a lot of non-relevant neighbors), maximum likelihood estimated P(*rel*) using training data is not likely to be a good estimation of P(*rel*) for future queries. Therefore, here we assume that each neighbor 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 connecting providers have somewhat similar contents.

In summary, for set-based threshold learning, the procedure a hub uses to decide the threshold for selection of its neighboring providers (hubs) is the following:

1. Given a query, the hub uses its resource ranking algorithm to calculate the ranking scores of its neighboring providers (hubs) 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 4.9;

3. The hub forwards the query to its neighboring providers (hubs) 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 neighbor how many of its returned documents are "relevant" and decide its relevance with respect to the query, where *r* is a parameter of threshold learning (50 in our experiments);

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 aggregate resource description at the hub 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 aggregate resource description; otherwise (if normalized ranking scores are used), go to the next step;

6. The hub estimates the distributions of the ranking scores of relevant and non-relevant neighbors 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 4.10−4.13 to calculate $\theta^*$ that gives the maximum $U(\theta)$ value and uses it as its threshold for selection of its provider (hub) neighbors for queries that belong to the same query group.

Although in theory set-based threshold learning can be applied to learn the threshold for resource selection of hubs, in practice it is more difficult to do so compared with using it to learn the threshold for resource selection of providers due to the difficulty of determining the criterion for a hub to be considered relevant with respect to a query. This is because the hub's likelihood of quickly reaching peers with relevant contents is measured not only by the number of "relevant" documents returned by its own providers, but also by the number of "relevant" documents returned by its downstream peers as well as how fast it can reach each of these peers. Effectively incorporating all of the above information in the relevance criterion is not a trivial task, which will further increase the complexity of threshold learning.

### 4.4.3 Combinations of Individual-Based and Set-Based Threshold Learning

Different methods for threshold learning have different weaknesses. Set-based threshold learning with original ranking scores assumes that the range of ranking scores for a query (as an indication of a hub's neighbors' overall degree of

relevance) correlates quite well with the average probability of this query's terms in the aggregate resource description at the hub for classifying queries into different groups. When such correlations do not exist for "outlier" queries, set-based threshold learning with original ranking scores performs badly. On the other hand, the problem of threshold learning with normalized ranking scores is that by normalizing ranking scores, a big difference in original ranking scores between peers is exaggerated and thus some peers seem much less relevant judged by their normalized ranking scores although their original ranking scores are quite high (but not at the same level as the highest one). In this case, threshold learning with normalized ranking scores tends to underestimate the number of relevant neighbors. For individual-based threshold learning, in addition to the aforementioned problem of using normalized ranking scores, it is not effective when the quality of resource ranking is not reliable. If resource ranking fails to rank most relevant neighbors on top of non-relevant neighbors, the top-ranked non-relevant neighbors will quickly decrease $E$ value, which can hardly be recovered even when relevant neighbors are included later on. Therefore, the optimal, low-valued E will occur early in the ranking and the method tends to select fewer neighbors than what is required to obtain a sufficient amount of relevant documents.

Since different threshold learning methods overestimate or underestimate the number of neighbors 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 neighbors to be selected and use this averaged value to decide how many top-ranked neighbors to select.

## 4.5 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 required in recalculating document scores. To avoid the cost of acquiring and maintaining global corpus statistics at each hub, we propose to use the aggregation of the hub's resource description and the resource descriptions of neighborhoods for all of its neighboring hubs 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 information about the contents of other peers and corpus statistics as hubs do in a hierarchical P2P network, 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.

## 4.6 Summary

In this chapter, we define a network search model to describe a full-text federated search mechanism in a hierarchical P2P network with the network overlay model described in Chapter 3. Although adapting existing approaches for a single, centralized directory service in distributed information retrieval to multiple, regional directory services in a hierarchical P2P network already gives the network new capability to support efficient query routing (resource selection) and result merging, the network search model goes beyond simple adaptation by introducing new methods in resource representation, resource selection, as well as result merging in view of new characteristics and requirements of full-text federated search in P2P networks. Specifically, the new features of our network search model are: i) we define the concept of a neighborhood and propose to use resource descriptions of neighborhoods for resource ranking of hubs in order to avoid "shortsighted" query routing that cannot see beyond the horizon of direct neighbors at the hub level; ii) we investigate pruning techniques that can effectively reduce the sizes of resource descriptions to lower system costs without significantly degrading retrieval accuracies; iii) we develop unsupervised methods for each hub to learn its selection threshold of neighbor ranking scores autonomously and adaptively so that extensive heuristic threshold tunings are no longer needed for resource selection in decentralized and dynamic environments; and iv) we modify the Kirsch's algorithm for result merging to generate integrated relevance-based rankings of documents without the cost of acquiring and maintaining global corpus statistics at each hub.

# *Chapter 5*

# EVALUATION

This chapter evaluates the performance of our proposed approaches to full-text federated search in P2P networks. The dataset and evaluation methodology are first described in Sections 5.1 and 5.2, followed by the experimental results in Section 5.3.

## 5.1 Dataset

There has been no standard data for evaluating the performance of full-text federated search in P2P networks, so we developed a *P2P testbed* (http://www.cs.cmu.edu/~callan/Data) based on the TREC WT10g Web test collection. We briefly describe below how we used the WT10g collection to generate the contents, topology, and queries for simulating federated search in a hierarchical P2P network of text digital libraries (Lu and Callan 2003a) (Lu and Callan 2003b).

### 5.1.1 Contents

TREC WT10g is a 10 gigabyte, 1.69 million document subset of the VLC2 collection (Hawking 2000). By combining all documents crawled from a single website into a single collection, the WT10g data was divided into 11,485 collections. 2,500 collections were randomly selected from them, consisting of a total number of 1,421,088 documents. The maximum, minimum, and average numbers of documents in a selected collection are 26,505, 8, and 568 respectively. The HTML title fields of the documents were used as document names. Each of the 2,500 collections defined a provider (text digital library) in a hierarchical P2P network.

### 5.1.2 Topology

The number of hubs was chosen to be 25 for the evaluation of resource ranking and result merging. For evaluating thresholding for resource selection, in order to test the effectiveness of automatic unsupervised threshold learning on more hubs, the number of hubs was increased to 50.

The network with 25 hubs was constructed to have content-based locality for its hub-provider topology. The 2,500 providers were clustered using a similarity-based soft clustering algorithm (Lin and Kondadadi 2001) into 25 clusters, each of which was associated with a hub. All the providers that belonged to the same cluster connected to the associated hub. The connections between the 25 hubs were generated randomly. Each hub could have no more than 7 and no less than 1 hub neighbors. A hub has on average 4 hub neighbors.

The network with 50 hubs was constructed to have content-based locality for its hub-provider topology and content-based small-world properties for its hub-hub topology. We used the topology evolution algorithm we developed (details in Chapter 6) to construct the network topology. A provider on average connects to 3 hub neighbors. A hub has on average 4 hub neighbors and 136 provider neighbors.

### 5.1.3 Queries

The queries provided by the U. S. National Institute for Standards and Technology (NIST) for the TREC WT10g Web test collection are TREC topics 451-550 with the standard TREC relevance assessments, which were used for the TREC-8 and TREC-9 Web Tracks. We refer to the title fields of these queries used in our experiments as *TREC queries*.

To study the performance of different methods for federated search in various P2P networks, sometimes a large number of queries is desired and clearly 100 queries is far from enough. Although Web logs from some search engines could provide a large amount of queries, there is no way to guarantee that there are relevant documents in the WT10g collection for these queries. One way to generate a large amount of queries in a controlled manner is to extract key terms from the documents in the WT10g collection and use them as queries. Prior research shows that 85% of the queries posted at Web search engines have 3 or less query terms (Jansen et al. 2000), so to be realistic, for most documents, we should only extract a few key terms as queries. We tried a variety of approaches to rank and extract key terms from documents. The best approach (judged manually) was to use a combination of unigram and bigram document language models, and some heuristic rules to rank document terms or term pairs for use as query terms. We describe this approach in more detail below.

We regard the probability $P_{emp}(t \mid d)$ that a term occurs in a document as a linear interpolation of the probability $P_{core}(t \mid d)$ that the term is generated by the unigram document language model, and the probability $P(t \mid background)$ that the term is generated by the background (general English) model:

$$P_{emp}(t \mid d) = \lambda P_{core}(t \mid d) + (1 - \lambda) P(t \mid background) \tag{5.1}$$

where $\lambda$ is the smoothing weight in this mixture model, and use $P_{core}(t \mid d)$ to evaluate how important a term is to the document. Given $P_{emp}(t \mid d)$ and $P(t \mid background)$, $P_{core}(t \mid d)$ can be calculated using the algorithm described in (Zhang, Y et al. 2002). Maximum likelihood estimation with simple Laplacian smoothing is used to calculate $P_{emp}(t \mid d)$. The value of $P(t \mid background)$ is based on the term frequency of term $t$ in the entire collection of WT10g.

The bigram document language model approach uses $P(t_1, t_2 \mid d)$ to measure the importance of a "phrase"[9] to the document. It is calculated as a mixture of maximum likelihood estimates:

$$P(t_1, t_2 \mid d) = 0.5P(t_1 \mid d)\frac{c(t_1, t_2 \mid d)}{c(t_1 \mid d)} + 0.5P(t_2 \mid d)\frac{c(t_1, t_2 \mid d)}{c(t_2 \mid d)} \tag{5.2}$$

where $c(\bullet)$ denotes count, $P(t_1 \mid d)$ and $P(t_2 \mid d)$ are smoothed maximum likelihood estimates of the probabilities that document $d$ generates terms $t_1$ and $t_2$ respectively, and $c(t_1, t_2 \mid d) / c(t_1 \mid d)$ and $c(t_1, t_2 \mid d) / c(t_2 \mid d)$ are un-smoothed empirical estimates of $P(t_2 \mid t_1, d)$ and $P(t_1 \mid t_2, d)$ respectively.

The unigram and bigram document language models are combined to rank document terms or term pairs and the top-ranked ones are selected as query terms based on the following heuristic rules:

1. The k-stem stemmer (Krovetz 1993) is used because the stemmed terms it generates are easier for people to understand, and because stemming a term more than once does not change it further;

2. Single-character terms are eliminated because it is rare to have single-character query terms;

3. Terms that begin with numbers are eliminated;

4. Terms that belong to a set of Web-specific stopwords such as "please", "thank", "previous" and "next" are eliminated;

5. Terms occurring in the title of the document are emphasized by a weight of 1.5;

[9] A phrase here refers to a pair of adjacent (non-stopword) terms in the document.

**Table 5.1 Distribution and randomly selected sample queries of different lengths.**

| Length | Distribution | Sample Query |
|---|---|---|
| 1 | 6.91% | sdtech |
| 2 | 39.79% | malignant hyperthermia |
| 3 | 29.16% | cardiac surgery; anesthesia |
| 4 | 22.66% | trade remedy; nafta law |
| 5 | 1.22% | drug drive collision police investigate |
| 6 | 0.26% | quarter company revenue increase sybase cash |

6. If the two top-ranked terms based on the unigram document language model appear to be a "phrase" in the top-ranked "phrases" based on the bigram document language model, these two terms are replaced by this "phrase"; and

7. The number of terms selected from a document is proportional to the length of this document, with an upper bound of 6.

A total number of 1,655,765 queries were generated from the WT10g using the approach described above. Table 5.1 shows the distribution of query lengths and randomly selected examples of the automatically-generated queries for different query lengths. We refer to these automatically-generated queries as *WT10g queries*.

Because it is expensive to obtain relevance judgments for over 1.6 million automatically-generated queries, we use the retrieval results from a single large collection as the baseline to measure how well federated search in P2P networks could locate those documents considered very relevant by centralized search. The single large collection is the subset of the WT10g used to define providers' contents in the P2P network ("*WT10g-subset*"). The 50 top-ranked documents retrieved from this single large collection for each query are treated as the set of "relevant" documents for this query. Although this methodology is not ideal, it is not unreasonable because distributed retrieval systems are not yet better than the "single collection" baseline. In fact, our prior experimental results show that similar conclusions regarding the performance of federated search in P2P networks can be drawn using automatically-generated queries and the "single collection" baseline compared with using TREC queries and real relevance judgments (Lu and Callan 2004b).

## 5.2 Evaluation Methodology

To shield the evaluation of full-text federated search from factors that affect search performance due to unpredictable and hard-to-control network conditions, we ignore the properties of the underlying physical layer and the interactions between logical and physical layers so that the evaluation can focus on how the search mechanism executed at the logical layer of the network impacts search performance.

Although our full-text search mechanism does not assume digital libraries to use the same document retrieval algorithm, for the convenience of experiments, each information provider in the hierarchical P2P network used the K-L divergence document retrieval algorithm to conduct full-text ranked retrieval (Ogilvie and Callan 2001).

The performance of federated search is measured by search accuracy as well as efficiency. When search results are presented without relevance-based rankings, search accuracy is measured using *set-based Precision*, *Recall*, and *F-Score* (Baeza-Yates and Ribeiro-Neto 1999):

$$\text{Recall} = \frac{|r|}{|A|} \quad (5.3) \qquad \text{Precision} = \frac{|r|}{|R|} \quad (5.4) \qquad \text{F-Score} = 2 / \left(\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}\right) \quad (5.5)$$

where $R$ is the set of documents returned by search in the P2P network, $A$ is the set of relevant documents for a TREC query, or the set of the (up to 50) top-ranked documents returned by search using the single WT10g-subset collection

("*single-collection*" *baseline*) for a WT10g query, and $r$ is the intersection of $R$ and $A$. $|\bullet|$ denotes the size of the set. Results are averaged over a set of queries.

If documents retrieved for a query are presented in a ranked list based on their estimated relevance scores, it is more appropriate to use *rank-based* accuracy measures. *11-point average precision versus recall* is a standard rank-based measure. The interpolated precisions at 11 standard recall levels are computed for each query and values from various queries are averaged at each recall level (Baeza-Yates and Ribeiro-Neto 1999).

Another standard rank-based measure commonly used to evaluate the performance of full-text federated search in distributed information retrieval is *average precision at given document cut-off values*, which computes the average precision over a set of queries when the 5, 10, 15, 20, or 30 top-ranked documents have been seen for each query (Callan 2000). Compared with 11-point average precision versus recall, average precision at given document cut-off values is more closely correlated with user satisfaction (Buckley and Voorhees 2004).

The above measures provide a good measurement of the overall performance of a method over a set of queries. It is also important to have some measure to evaluate the performance of the individual queries in order to investigate the behavior that can be easily hidden by an average precision computation. To conveniently compare the performance of different methods for the individual queries, we would like a single precision value to summarize the performance of each query. Therefore, we chose to use *average precision over a range of document cut-off values* to minimize the evaluation error rate associated with precision at a single document cut-off value (Hull 1993) (Buckley and Voorhees 2000). Specifically, for each query's result, its precisions at document cut-off values 1-30 are averaged to get the *average precision over 1-30 document cut-offs*. The average precisions for the results of various queries can be further averaged to get the *overall average precision over 1-30 document cut-offs* for a set of queries. A similar measure is used in (Stenmark 2005) to evaluate retrieval performance.

Search efficiency is measured by the efficiency of query routing, i.e., the average number of query messages routed for each query in the network.

## 5.3 Experimental Results

The main components for full-text federated search in a hierarchical P2P network include hub-provider and hub-hub query routing, document retrieval, and result merging. Query routing can be further decomposed into the components of resource representation, resource ranking, and thresholding for resource selection. In order to separate the effect of a particular component on federated search from those of the others, we evaluated our approaches to full-text federated search in a hierarchical P2P network progressively. To be specific, starting from a baseline setting of limited-horizon flooding or random selection for query routing and no result merging, we modified the setting progressively to apply our methods one at a time and compared the system performance before and after each modification until all of our methods have been applied and evaluated. We devote the following six sections to the experimental results with regard to evaluating our approaches to resource ranking of providers (Section 5.3.1), resource ranking of hubs (Section 5.3.2), result merging (Section 5.3.3), pruning for resource representation (Section 5.3.4), thresholding for resource selection of providers (Section 5.3.5), and thresholding for resource selection of hubs (Section 5.3.6). Table 5.2 lists the experimental settings used for the results in different sections. A dark gray cell in the table marks the component to be evaluated in the corresponding section, and any component already evaluated in one of the preceding sections is listed in a light gray cell. "K-L" in the table refers to the K-L divergence resource ranking algorithms described in Section 4.3 for full-text resource ranking, or the K-L divergence document retrieval algorithm (Ogilvie and Callan 2001) for full-text document retrieval.

For all the experimental results reported here, each query was issued to the network by a provider randomly selected to act as a consumer temporarily and forward the query to all of its neighboring hubs. For resource ranking, the smoothing parameter $\mu$ in Dirichlet smoothing was set to be 1000, a value which has been shown to work well for ad-hoc retrieval over various TREC test collections (Zhai and Lafferty 2001). It is also shown in (Zhai and Lafferty 2001) that retrieval using Dirichlet smoothing is quite robust when the value of $\mu$ is chosen from a wide range (500-10000). The number of the top-ranked documents returned by each provider that received a query was up to 50.

**Table 5.2 The experimental settings used for the results in different sections to evaluate different components in federated search.**

| Setting | Sec. 5.3.1 | Sec. 5.3.2 | Sec. 5.3.3 | Sec. 5.3.4 | Sec. 5.3.5 | Sec. 5.3.6 |
|---|---|---|---|---|---|---|
| Queries | 15,000 WT10g | 100 TREC/ 1,000 WT10g | 100 TREC/ 1,000 WT10g | 100 TREC/ 1,000 WT10g | 100 TREC/ 1,000 WT10g | 100 TREC/ 1,000 WT10g |
| TTL | 4 | 6 | 6 | 6 | 6 | 6 |
| # hubs | 25 | 25 | 25 | 25 | 50 | 50 |
| # providers | 2500 | 2500 | 2500 | 2500 | 2500 | 2500 |
| Document retrieval | K-L | K-L | K-L | K-L | K-L | K-L |
| Ranking of providers | K-L/ random/ flooding | K-L | K-L | K-L | K-L | K-L |
| Ranking of hubs | flooding | K-L/ random/ flooding | K-L | K-L/ flooding | K-L | K-L |
| Result merging | N/A | N/A | extended Kirsch's/ raw scores/ centralized | extended Kirsch's | extended Kirsch's | extended Kirsch's |
| Resource description | full | full | full | full/ pruned | pruned | pruned |
| Selection of providers | top 1% | top 1% | top 1% | top 1% | learned threshold/ fixed optimal | learned threshold |
| Selection of hubs | N/A | top 1 | top 1 | top 1 | top 2 | learned threshold/ fixed optimal |

Although most experimental settings could remain the same for the experiments reported in different sections, adjustments for some settings were still necessary. First, an initial TTL (time-to-live) value of 4 for a query message would be adequate when the flooding technique was used for hub-hub query routing, but full-text resource selection of a very small number of neighboring hubs might require a larger initial TTL value in order to reach a sufficient number of hubs. Therefore, the initial TTL value for a query message was set to 4 in the earlier experiments and increased to 6 in the later experiments. Second, a larger number of hubs (50 instead of 25) were used for evaluating thresholding for resource selection in order to test the effectiveness of automatic unsupervised threshold learning on more hubs. Finally, more top-ranked hubs (2 instead of 1) were selected for resource selection of hubs when the number of hubs was increased to 50 in order for each query to reach a similar proportion of the network as before.

### 5.3.1 Resource Ranking of Providers

The experiments reported in this section focused on comparing the performance of federated search using full-text resource selection with that using random selection or flooding for hub-provider query routing. A hub that received a query i) used the K-L divergence resource ranking algorithm to rank its neighboring providers (Section 4.3.1) and forwarded the query to up to the top 1.0% of these ranked providers ("*full-text resource selection*"), or ii) randomly forwarded the query to a subset of its neighboring providers to yield a similar number of query messages as i) ("*random selection*"), or iii) flooded the query to all neighboring providers ("*flooding*").

**Table 5.3 The search performance of different methods for hub-provider query routing.**

| Method | Precision | Recall | F-Score | Average Number of Query Messages |
|---|---|---|---|---|
| Flooding | 62.12% | 33.76% | 0.4375 | 540 |
| Random | 60.94% | 19.59% | 0.2965 | 122 |
| Full-text (K-L + top 1%) | 71.82% | 29.65% | 0.4197 | 116 |

Flooding each query to all neighboring providers would yield a huge volume of query traffic which was difficult to handle even for a simulator. Therefore, flooding was applied together with an additional "*match all query terms*" rule for hub-provider query routing, which required the vocabulary of a provider's content to match all the query terms in order for the provider to be eligible to receive the query. For fair comparison, the same query matching rule was also applied to full-text resource selection and random selection. For random selection and flooding, each hub needed to record the vocabulary of each neighboring provider in order to apply the query matching rule, which could be viewed as another form of resource description.

We used the hierarchical P2P network of 25 hubs and 2,500 providers. The initial TTL value for a query message was 4. Hub-hub query routing was fixed to be flooding, so mostly a query message would go through all the hubs given the above network size and TTL value. No result merging was conducted.

As our first set of experiments on federated search in the hierarchical P2P network of text digital libraries, we used a large number of queries requesting diverse contents so that peers in almost every part of the network could be involved in search activities. 15,000 WT10g queries were randomly selected from the automatically-generated queries defined in the P2P testbed and used for the experiments reported in this section.

Table 5.3 shows the experimental results using different methods of hub-provider query routing, measured by set-based search accuracy and efficiency. Because of the additional query matching rule for hub-provider query routing and the limit on the maximum number of documents (50) returned by each provider, flooding for hub-provider query routing didn't result in a 100% Recall. Compared with flooding, random selection greatly improved search efficiency, at the cost of reducing Recall. Although their difference in Precision was negligible (1.9%), random selection's performance loss in Recall relative to flooding was very large (42.0%). In contrast, full-text resource selection (K-L divergence resource ranking of providers and a fixed threshold) could significantly improve search efficiency without degrading Recall much. Its increase in Precision compared with flooding and random selection indicates that using term frequency information enabled resource ranking to effectively estimate each provider's likelihood of satisfying the user's information need so that query routing could be restricted to those providers with high likelihood. The slight drop in Recall using full-text resource selection compared with flooding was because query messages were routed to much fewer providers and so relevant documents from those providers not selected were missed. The F-score values of full-text resource selection and flooding were comparable, which demonstrates that these two methods gave comparable search accuracies. In summary, full-text resource selection gave a much better combination of search accuracy and efficiency than random selection or flooding for hub-provider query routing.

An additional finding in our experiments is that although the Precision and Recall values for the individual queries varied, the average Precision and Recall over random subset of the tested 15,000 WT10g queries had similar values as those in Table 5.3 as long as the subset included at least 1,000 queries. This suggests that 1,000 representative WT10g queries would be sufficient for evaluating federated search in the hierarchical P2P network defined using the P2P testbed. Therefore, the number of WT10g queries used in later experiments was reduced to 1,000 for efficient simulation.

### 5.3.2 Resource Ranking of Hubs

Fixing the method of hub-provider query routing to be full-text resource selection using K-L divergence resource ranking and a fixed threshold (the top 1.0% of the ranked neighboring providers), we shift our attention to hub-hub query routing. A hub that received a query i) used the K-L divergence resource ranking algorithm to rank its neighboring hubs based on their resource descriptions of neighborhoods (Section 4.3.2) and forwarded the query to the one top-ranked neighboring hub ("*full-text resource selection*"), or ii) randomly forwarded the query to one of its

**Table 5.4 The search performance of different methods for hub-hub query routing.**

| Queries | Method | Precision | Recall | F-Score | Average Number of Query Messages |
|---|---|---|---|---|---|
| TREC | Flooding | 1.41% | 29.74% | 0.0269 | 177 |
| | Random | 1.41% | 21.76% | 0.0265 | 63 |
| | Full-text (K-L + top 1) | 1.69% | 25.51% | 0.0317 | 59 |
| WT10g | Flooding | 12.88% | 69.92% | 0.2175 | 174 |
| | Random | 12.50% | 50.55% | 0.2004 | 60 |
| | Full-text (K-L + top 1) | 14.10% | 60.63% | 0.2288 | 54 |

neighboring hubs ("*random selection*"), or iii) flooded the query to all neighboring hubs ("*flooding*"), all of which excluded the sender hub from which it received the query.

In the previous section, a "match all query terms" rule was used to prevent each hub from routing the query to all of its neighboring providers in order to reduce unnecessary query traffic for efficient simulation. For the experiments in this section and in the following sections, since hub-provider query routing was based on full-text resource selection, the number of providers to receive the query was already greatly reduced with an appropriate selection threshold, so the query matching rule was not necessary anymore. Without the query matching rule, set-based Recall was expected to increase because providers whose resource descriptions matched some but not all query terms might be selected and contribute additional relevant documents. However, set-based Precision might decrease because these partially-matched providers might return more non-relevant documents than relevant ones.

Because each hub only selected one of its hub neighbors for full-text or random hub-hub query routing, the TTL of each query message was increased to 6 in order to reach a sufficient number of hubs. The resource descriptions of neighborhoods were created using the procedure described in Section 4.2.3. The same hierarchical P2P network used for the experiments described in the previous section was used.

Experiments were run on two sets of queries. The first set of queries was the 100 TREC queries (the title fields of TREC topics 451-550). The standard TREC relevance assessments supplied by NIST were used. The second set of queries was a set of 1,000 WT10g queries selected from the automatically-generated queries defined in the P2P testbed according to the query length distribution described in (Jansen et al. 2000). The "single collection" baseline (Section 5.2) was used as relevance judgments.

The same performance measures described in the previous section were used here, that is, set-based Precision and Recall were used to measure search accuracy, and the average number of query messages routed for each query in the network was used to measure search efficiency.

Table 5.4 shows the experimental results of using different methods of hub-hub query routing. As expected, the values of set-based Precision were low due to the large number of non-relevant documents returned by the providers that only partially matched a query. This implies that without relevance-based document ranking, relevant documents are more likely to be buried by the many more non-relevant documents in the search results.

Random selection had similar Precision as flooding but its average relative decrease in Recall was 27.3%. Compared with flooding, full-text resource selection using K-L divergence resource ranking of hubs based on resource descriptions of neighborhoods required around one third of the number of query messages with slightly higher Precision and on average 13.8% relative degradation in Recall. The higher F-score values of full-text resource selection demonstrate its effectiveness in selecting hubs that could reach those providers most likely to satisfy the user's information need. Full-text resource selection and random selection for hub-hub query routing gave similar search efficiency but the accuracy of the former was consistently higher than the latter. To summarize, similar as the evaluation on hub-provider query routing, full-text resource selection (K-L resource ranking of hubs and a fixed threshold) gave a better combination of search accuracy and efficiency than random selection or flooding for hub-hub query routing.

The experimental results in Table 5.4 provides additional support on the usefulness of the automatically-generated WT10g queries and the "single collection" baseline in evaluating the performance of federated search in P2P networks. When this set of queries was used to evaluate the performance of federated search in a hierarchical P2P network, it directly measured the ability of federated search in the hierarchical P2P network to match the results from search in a centralized environment, not the effectiveness of federated search from the user's point of view. Therefore, using the "single collection" baseline as the set of relevant documents relied on the assumption that search using a centralized index was effective in satisfying the user's information needs, which was not necessarily the case. Due to this reason, we were concerned with whether the automatically-generated queries would behave similarly as real queries and whether the conclusions drawn using the "single collection" baseline for evaluation would still be valid with real relevance judgments. If we compare the figures for WT10g queries with those for TREC queries in Table 5.4, we can see that although the absolute values were quite different, the relative performance difference of different methods for WT10g queries was similar to that for TREC queries. Therefore, the same conclusion regarding the relative effectiveness of various methods could be drawn using either WT10g queries with the "single collection" baseline, or TREC queries with real relevance assessments. This indicates that the automatically-generated queries and the "single collection" baseline are useful resources in studying federated search in P2P networks.

### 5.3.3 Result Merging

To adopt our approach to result merging, each provider that responded to a query augmented the result list with the summary statistics (document length and how often each query term matched) of the returned documents. Each hub collected and merged the results returned by its selected providers by using the *extended Kirsch's* algorithm to recalculate document scores using these summary statistics (Section 4.5). Hubs returned the merged results for a query to the consumer that issued the query, and the consumer directly merged the results from multiple hubs based on the document scores they provided.

The extended Kirsch's result merging algorithm was compared against two baseline methods. The upper bound baseline method took the documents returned by federated search in the hierarchical P2P network and ranked them by their corresponding scores returned from search in a centralized collection, i.e., WT10g-subset collection ("*centralized scores*"). The lower bound baseline method directly merged the documents from different providers using the initial document scores provided by these providers ("*raw score merge*").

Query routing was fixed to be full-text resource selection with a fixed threshold, i.e., the K-L divergence resource ranking algorithm was used by hubs to select neighboring providers and hubs, and each query was routed to the top 1.0% of the ranked neighboring providers and the one top-ranked neighboring hub.

The initial TTL value of each query message was 6. The hierarchical P2P network was the same one as used in previous two sections. Both the 100 TREC queries and the 1,000 WT10g queries were used.

Because result merging made it possible for search results to be presented with relevance-based rankings of documents, rank-based measures could be used to evaluate search accuracy. Figure 5.1 shows the performance of different result merging methods in the hierarchical P2P network evaluated by 11-point precision versus recall. The extended Kirsch's algorithm worked much better than the lower bound at low to medium recall levels. Its performance was similar to the lower bound at high recall levels. At all recall levels, the extended Kirsch's algorithm had near "optimal" performance compared with the upper bound. Its relative improvements in average precision over the lower bound were 39.3% for the 100 TREC queries and 30.5% for the 1,000 WT10g queries. The performance losses in average precision relative to the upper bound were 1.8% for the 100 TREC queries and 4.7% for the 1,000 WT10g queries, which few users would notice. Thus with a small amount of cooperation, satisfactory performance could be obtained for result merging in the hierarchical P2P network.

Figure 5.2 shows the performance of different result merging methods evaluated using average precision at document cut-off values 5, 10, 15, 20, and 30, which is more correlated with user satisfaction. Once more, the performance of the extended Kirsch's algorithm was very similar to the upper bound and much better than the lower bound. Compared with the values of set-based Precision in the previous section, the average precisions at small document cut-offs were much higher. This indicates that although more non-relevant documents were returned than relevant documents, by

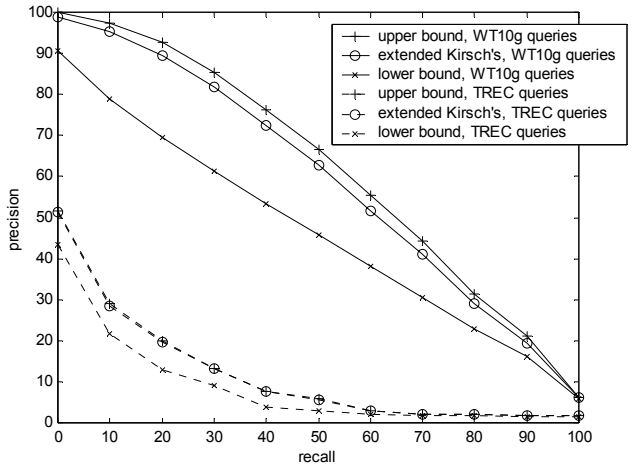


**Figure 5.1 Result merging evaluated by 11-point average precision versus recall.**

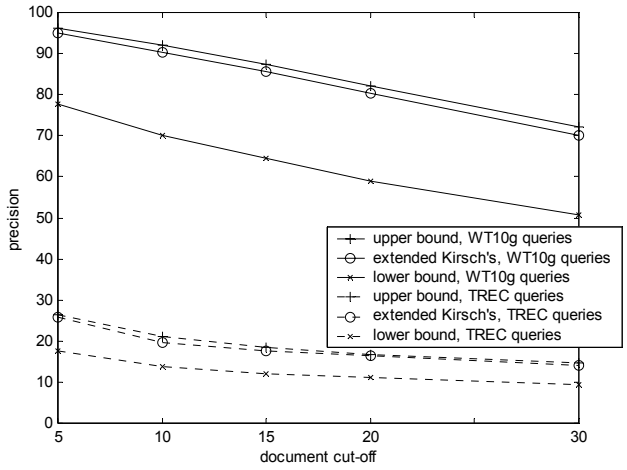


**Figure 5.2 Result merging evaluated by the average precisions at given document cut-off values.**

using result merging to generate relevance-based result presentation, a lot of relevant documents can be effectively identified.

The results in this section again demonstrate that the same conclusion regarding the relative effectiveness of various methods for federated search in the hierarchical P2P network could be drawn using either WT10g queries or TREC queries.

### 5.3.4 Pruning for Resource Representation

As described in Section 4.2.4, the size of a resource description can be reduced by pruning terms that are rare or terms that are more likely to be generated by general English than by the content of the resource. Because the former approach is simple and has been shown to be effective (Lu and Callan 2003a), while the latter approach is complex and expensive to use in practice for resource descriptions of large vocabularies, we focused on evaluating the effectiveness of reducing the size of a resource description by pruning terms of low frequencies.

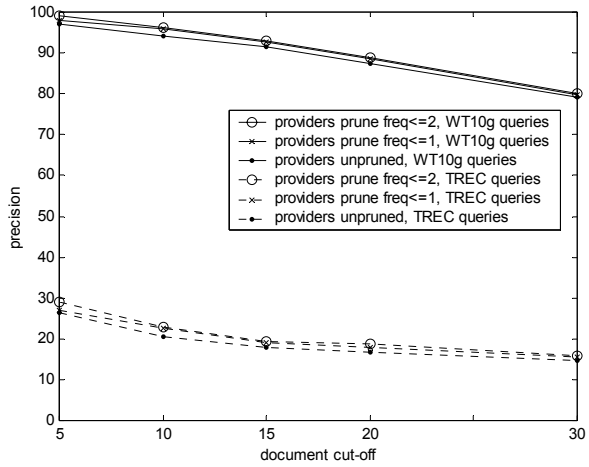


**Figure 5.3 The average precisions at given document cut-off values with flooding for hub-hub query routing.**

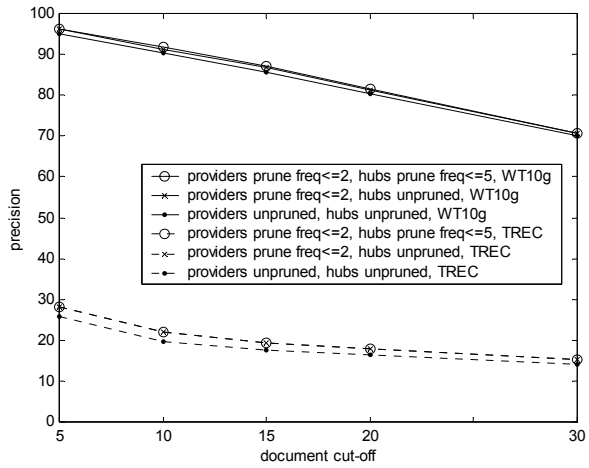


**Figure 5.4 The average precisions at given document cut-off values with full-text resource selection for hub-hub query routing.**

We tested the effects of pruning resource descriptions on hub-provider query routing as well as hub-hub query routing in two steps. The first step focused on the effectiveness of using pruned resource descriptions for full-text resource selection of providers. Rare terms in providers' resource descriptions were pruned and the flooding technique was used for hub-hub query routing. In the second step, rare terms in all types of resource descriptions were pruned and full-text resource selection of both providers and hubs were used.

**Table 5.5 Average amount of reduction in the vocabulary size of a resource description with different pruning settings.**

| Resource Type | Pruning Setting | Relative Vocabulary Size Reduction |
|---|---|---|
| Provider | prune freq ≤ 1 | 42.2% |
| Provider | prune freq ≤ 2 | 58.3% |
| Neighborhood | prune freq ≤ 5 | 86.4% |

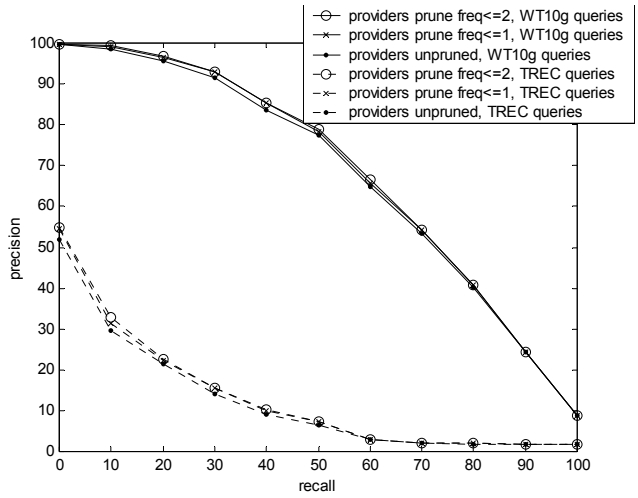


**Figure 5.5 The 11-point precision versus recall with flooding for hub-hub query routing.**

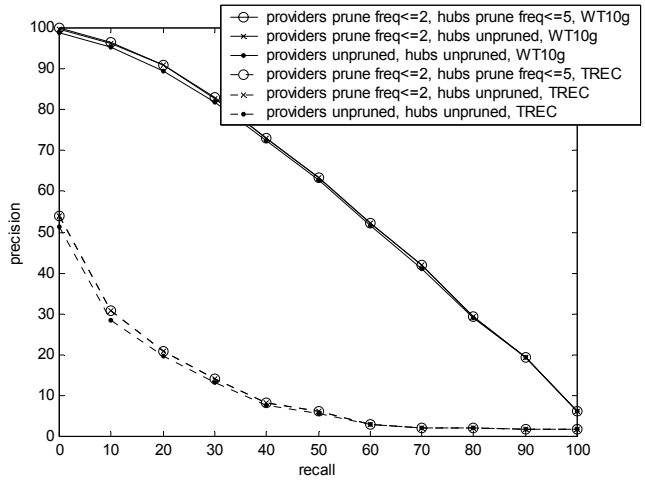


**Figure 5.6 The 11-point precision versus recall with full-text resource selection for hub-hub query routing.**

Pruning of a provider's resource description took place at the provider. Upon request, each provider supplied a pruned resource description to the network. Each hub was responsible for pruning its own resource description and the neighborhoods' resource descriptions it maintained.

Query routing used the same methods as in the previous section, which were K-L divergence resource selection of the top 1% of the ranked neighboring providers and the one top-ranked neighboring hub. The initial TTL value of each query message was 6. Given the evaluation in the previous section, we were confident of adding the component of result merging to federated search. The extended Kirsch's algorithm was adopted to merge results.

Experiments that were run using the 100 TREC queries and the 1,000 WT10g queries are shown in this section. The same hierarchical P2P network used in the previous sections was used.

Since using pruned resource descriptions does not affect search efficiency, search accuracy alone is sufficient to measure the effect of pruning on search performance. We measured search accuracy by both 11-point average

precision versus recall and average precision at document cut-off values 5, 10, 15, 20, and 30. The effect of pruning on reducing system overhead for sending and storing resource descriptions was measured by the average amount of reduction in the vocabulary size of a provider's or a neighborhood's resource description since the size of a resource description was roughly proportional to the total number of terms it contained.

Figure 5.3 compares average precisions at document cut-off values 5, 10, 15, 20, and 30 with and without the pruning of providers' resource descriptions when hub-hub query routing used the flooding technique and hub-provider query routing used full-text resource selection. Figure 5.4 compares the cases when all types of resource descriptions were pruned against those with the pruning of only providers' resource descriptions or those without any pruning, using full-text resource selection for both hub-hub query routing and hub-provider query routing. Figures 5.5 and 5.6 show the results evaluated using 11-point average precision versus recall. Table 5.5 shows the average amount of reduction in the vocabulary size of a provider's resource description or a neighborhood's resource description by applying different pruning settings. "prune freq <= *x*" in the figures and in the table means that all the terms that occurred no more than *x* times in a resource description were discarded. The figures show that pruning with small threshold values didn't decrease search accuracy at any recall levels. Table 5.5 demonstrates that even a small pruning threshold could result in a great reduction in the cost associated with resource representation. Overall, the results indicate that reducing the sizes of resource descriptions by pruning low-frequency terms enabled dramatic savings in communication and storage costs without degradation in search accuracy.

### 5.3.5 Thresholding for Resource Selection of Providers

The effectiveness of different threshold learning methods (Section 4.4) for resource selection of providers can be evaluated by comparing the performance of federated search using resource selection of providers based on the learned thresholds with that based on the fixed threshold empirically tuned for optimal performance. To obtain the "optimal" threshold value for resource selection of providers based on a fixed threshold, a series of experiments was conducted with a set of continuously varied threshold values. Resource selection of providers based on a learned threshold used i) *set-based threshold learning with original ranking scores* (*method I*), or ii) *set-based threshold learning with normalized ranking scores* (*method II*), or iii) *individual-based threshold learning with normalized ranking scores* (*method III*), or iv) *the combination of methods I and II*, or v) *the combination of methods I, II, and III*. The number of the top-ranked merged documents regarded as "relevant" documents for each training query (*r* value) was 50 (Sections 4.4.1 and 4.4.2). The parameter *b* in individual-based threshold learning was empirically chosen to be 3 to value precision more than recall (Section 4.4.1). The parameter *n* in set-based threshold learning which determined the criterion for a provider to be considered relevant with respect to a query was set to 5 (Section 4.4.2). These parameter values worked effectively for hubs that had different numbers of neighbors and covered different content areas.

We fixed the method for resource selection of hubs to be full-text resource selection based on a fixed threshold (selecting the two top-ranked neighboring hubs). Experiments were run using the extended Kirsch's algorithm for result merging and an initial TTL value of 6. Resource descriptions were pruned to discard terms that occurred no more than twice in a provider's resource description and terms that occurred no more than five times in a hub's or a neighborhood's resource description.

The hierarchical P2P network used for evaluation was a network of 50 hubs and 2,500 providers in order to test the effectiveness of threshold learning on more hubs (Section 5.1.2). The queries were the 100 TREC queries with the standard relevance assessments supplied by NIST, and the 1,000 WT10g queries with the "single collection" baseline. The 100 TREC queries were used for both threshold learning and evaluating the effectiveness of the learned threshold. Each experiment using the learned threshold had several runs in a way similar as leave-one-out cross validation, i.e., 99 queries were used as training queries to learn the threshold for testing on the 1 query that was left out, and the results from different runs were averaged to get the final result. When the 1,000 WT10g queries were used for testing, the 100 TREC queries were used as training queries. Because all threshold learning methods were unsupervised, only queries and retrieved documents were used for training. The relevance judgments provided by NIST for the 100 TREC queries were not used to learn the thresholds for resource selection of providers.

To conveniently compare the performance of different threshold learning methods with that using continuously varied threshold values, we used average precision over 1-30 document cut-offs to summarize the performance of each query and averaged the values for all queries (overall average precision over 1-30 document cut-offs). Figure 5.7 compares

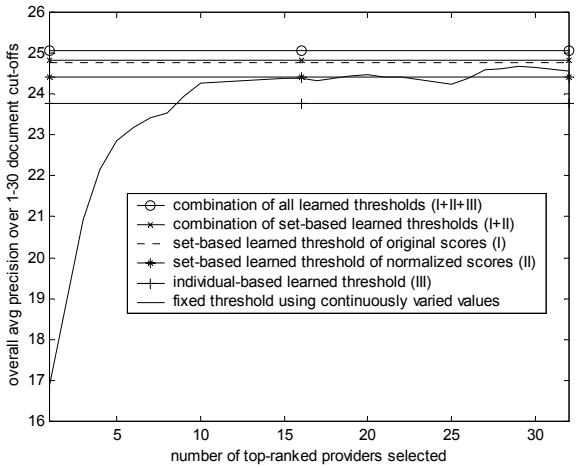


**Figure 5.7 The overall average precisions over 1-30 document cut-offs using different methods to get thresholds for resource selection of providers, 100 TREC queries.**

**Table 5.6 The search efficiency of different thresholding methods for full-text resource selection of providers, 100 TREC queries.**

| Method | Average Number of Hub-Provider Query Messages | Method | Average Number of Hub-Provider Query Messages |
|---|---|---|---|
| "optimal" fixed | 221 | III | 85 |
| I | 205 | I+II | 169 |
| II | 123 | I+II+III | 143 |

**Table 5.7 The performance of different methods to learn thresholds compared against the "optimal" fixed threshold for full-text resource selection of providers, 100 TREC queries.**

| Method | % "equal" queries | % "better" queries | Relative change over "better" queries | % "worse" queries | Relative change over "worse" queries | Overall relative change |
|---|---|---|---|---|---|---|
| I | 40.0% | 27.0% | +52.63% | 33.0% | -15.45% | +9.11% |
| II | 51.0% | 23.0% | +22.63% | 26.0% | -19.05% | +0.25% |
| III | 32.0% | 36.0% | +13.83% | 32.0% | -22.51% | -2.22% |
| I+II | 50.0% | 22.0% | +58.95% | 28.0% | -12.40% | +9.50% |
| I+II+III | 53.0% | 27.0% | +52.57% | 20.0% | -12.70% | +11.65% |

the overall average precisions over 1-30 document cut-offs for the 100 TREC queries when different thresholding methods were used for full-text resource selection of providers. The value used for the fixed threshold varied from 1 to 32 (roughly 20% of the average number of neighboring providers a hub had and 5% of the number of providers that belonged to the largest content-based cluster). Table 5.6 shows the search efficiency of different thresholding methods in terms of the average number of query messages routed from hubs to providers for the 100 TREC queries. From the figure and the table we can see that using full-text resource selection of providers based on a fixed threshold, the search accuracy remained relatively stable when the number of the top-ranked neighboring providers selected was larger than 10 and the "optimal" threshold value that enabled the highest average precision within the tested range was 29, which required an average of 221 messages to be routed from hubs to providers. Compared with using this "optimal" fixed threshold for resource selection of providers, method I, the combination of methods I and II , and the combination of methods I, II, and III enabled slightly superior search accuracies with moderately improved search efficiency. Method II and method III provided search accuracies comparable to that of the "optimal" fixed threshold with much higher

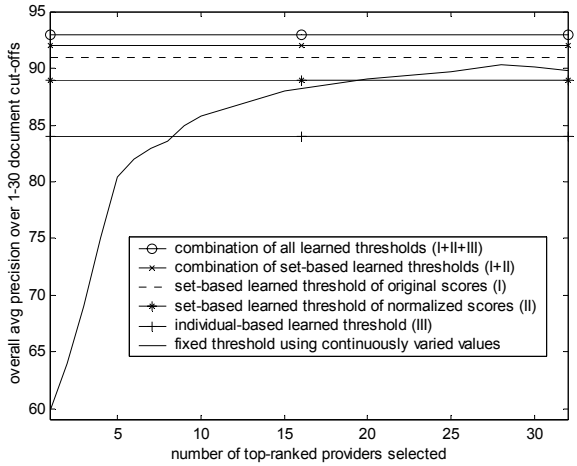


**Figure 5.8 The overall average precisions over 1-30 document cut-offs using different methods to get thresholds for resource selection of providers, 1,000 WT10g queries.**

**Table 5.8 The search efficiency of different thresholding methods for full-text resource selection of providers, 1,000 WT10g queries.**

| Method | Average Number of Hub-Provider Query Messages | Method | Average Number of Hub-Provider Query Messages |
|---|---|---|---|
| "optimal" fixed | 218 | III | 83 |
| I | 203 | I+II | 165 |
| II | 120 | I+II+III | 139 |

**Table 5.9 The performance of different methods to learn thresholds compared against the "optimal" fixed threshold for full-text resource selection of providers, 1,000 WT10g queries.**

| Method | % "equal" queries | % "better" queries | Relative change over "better" queries | % "worse" queries | Relative change over "worse" queries | Overall relative change |
|---|---|---|---|---|---|---|
| I | 44.5% | 26.8% | +55.26% | 28.7% | -11.58% | +11.49% |
| II | 53.7% | 21.6% | +24.72% | 24.7% | -18.75% | +0.71% |
| III | 35.0% | 38.5% | +11.38% | 26.5% | -19.65% | -0.83% |
| I+II | 55.1% | 23.4% | +62.13% | 21.5% | -13.19% | +11.70% |
| I+II+III | 57.3% | 26.2% | +54.97% | 16.5% | -13.28% | +12.21% |

search efficiency. This illustrates that our threshold learning methods were able to determine thresholds autonomously and adaptively for close-to-optimal search performance, and the hybrid approaches were slightly more effective than individual-based or set-based approaches alone by taking advantage of their complementary strengths.

Above we compared the performance of different methods for thresholding over a set of queries. We also want to compare their performance for the individual queries because we are interested in investigating how the relative performance gain or loss changed for different queries, but this information is hidden by averaging precisions over many queries. Using average precision for 1-30 document cut-offs as a single precision value for each query, for each threshold learning method we compared its performance against the performance of using the "optimal" fixed threshold by measuring how many queries had better/worse/equal performance, and the average relative performance change for the group of better/worse/equal queries. Table 5.7 shows the results of such comparisons for the 100 TREC queries. It demonstrates that although none of the threshold learning methods consistently outperformed the method of using the

"optimal" fixed threshold for every query, their average relative improvements for those queries with better performance were more significant than their average relative degradations for those queries with worse performance (except method III), earning our confidence in the stability of using the learned resource selection thresholds for optimal search performance. Among the five threshold learning methods, set-based threshold learning with original ranking scores (method I), the combination of two set-based methods (method I + method II), and the combination of individual-based and set-based threshold learning (method I + method II + method III) had a much higher degree of gain than loss. Particularly, the combination of methods I, II, and III had the highest overall relative performance improvement. It was also the most stable method. Its percentage of queries having the same performance as that of using the "optimal" fixed threshold (53.0%) was the largest among all threshold learning methods, and its percentage of queries having worse performance than that of using the "optimal" fixed threshold (20.0%) was the smallest.

Figure 5.8, Table 5.8 and Table 5.9 show the corresponding results for the 1,000 WT10g queries, which could lead to the same conclusion as the results for the 100 TREC queries with respect to the effectiveness and stability of automatic threshold learning.

In summary, compared with using the "optimal" fixed threshold with relatively expensive empirical tuning, automatic threshold learning for resource selection of providers, particularly the hybrid approach, enabled less query traffic and slightly higher average accuracy with a relatively stable performance over the individual queries.

### 5.3.6 Thresholding for Resource Selection of Hubs

In Section 4.4.2 we discussed the difficulty of applying set-based threshold learning to learn the threshold for resource selection of hubs due to the nontrivial task of determining the criterion for a hub to be considered relevant with respect to a query. Even if we make extra effort to determine the criterion, for the hierarchical P2P network defined in our testbed, the number of hub neighbors each hub has is too small to make reliable estimation for the distributions of the ranking scores of relevant and non-relevant hubs. Therefore, we only evaluated resource selection of hubs with individual-based threshold learning (Section 4.4.1).

With individual-based threshold learning for resource selection of hubs, because the number of non-relevant documents returned by each hub neighbor almost always greatly surpassed the number of relevant documents it returned, precisions at all hub ranks were almost certain to be inconsiderably low, making recall a much more sensible measure than precision. For this reason, instead of using the *E* evaluation measure to combine precision and recall and deciding the threshold by searching for the optimal *E* value, we returned to the initially proposed simple method for individual-based threshold learning. A hub decided its threshold of ranking scores for a query by going down the list of neighboring hubs sorted by their normalized ranking scores until a sufficiently large percentage of "relevant" documents had been returned, i.e., a sufficiently high recall (50.0%) was obtained, and using the last ranking score before stopping as the threshold. The number of the top-ranked merged documents for each training query that were regarded as "relevant" documents was 50.

Resource selection of hubs based on the learned thresholds was compared with that based on the "optimal" fixed threshold. In the hierarchical P2P network of 50 hubs and 2,500 providers, hub-provider query routing used full-text resource selection of providers with the hybrid approach that combined individual-based and set-based threshold learning. Result merging used the extended Kirsch's algorithm. Each query message was initialized to have a TTL of 6. Results are shown for the 100 TREC queries and the 1,000 WT10g queries.

In order to find the "optimal" fixed threshold for resource selection of hubs, we varied the number of the top-ranked hub neighbors each hub selected from one to three and compared their performance. Using overall average precision for 1-30 document cut-offs to measure search accuracy, for the 100 TREC queries, selecting the two top-ranked hub neighbors for hub-hub query routing had an average precision of 25.06%, slightly outperforming selecting the one top-ranked hub neighbor (24.87%) or the three top-ranked hub neighbors (24.88%). Compared with selecting the one top-ranked hub neighbor, selecting the two top-ranked hub neighbors had a 35.87% overall relative performance improvement calculated on a query-by-query basis. Selecting the two top-ranked hub neighbors and selecting the three top-ranked hub neighbors had almost the same search accuracy. Selecting the two top-ranked hub neighbors also gave

**Table 5.10 The performance of using the learned thresholds compared with the "optimal" fixed threshold for resource selection of hubs, 100 TREC queries.**

| % "equal" queries | % "better" queries | Relative change over "better" queries | % "worse" queries | Relative change over "worse" queries | Overall relative change |
|---|---|---|---|---|---|
| 54.0% | 21.0% | +13.64% | 25.0% | -12.11% | -0.16% |

**Table 5.11 The performance of using the learned thresholds compared with the "optimal" fixed threshold for resource selection of hubs, 1,000 WT10g queries.**

| % "equal" queries | % "better" queries | Relative change over "better" queries | % "worse" queries | Relative change over "worse" queries | Overall relative change |
|---|---|---|---|---|---|
| 65.1% | 17.5% | +15.22% | 17.4% | -13.96% | +0.23% |

the best combination of search accuracy and efficiency for the 1,000 WT10g queries. Therefore, for our network settings, selecting the two top-ranked hub neighbors was "optimal".

We compared resource selection of hubs based on the learned thresholds with selecting the two top-ranked hub neighbors. The results in Tables 5.10 and 5.11 show that these two thresholding methods for resource selection of hubs had very similar search accuracies on the individual queries. They also had almost the same search efficiency. Selecting the two top-ranked hub neighbors had an average number of 154 query messages, and the average number of query messages for resource selection of hubs based on the learned thresholds was 156. Although a fixed threshold of value 2 seemed an easy empirical choice that yielded the best performance for resource selection of hubs in the tested hierarchical P2P network, this "optimal" threshold value highly depends on the configuration of the network. In a different P2P network, especially a P2P network of larger scale with more hubs and hub connections, it is unlikely that selecting the two top-ranked hub neighbors is still the optimal choice. Therefore, the advantage of unsupervised threshold learning lies in its ability to learn close-to-optimal thresholds for resource selection of hubs automatically for different network configurations.

## 5.4 Summary

Although real applications of P2P file-sharing systems have reached network sizes of hundreds of thousands of peers sharing millions of documents, most research into federated search in P2P networks either have far smaller scales (e.g., in the scale of hundreds), or rely on symbolic data items without any real content. Evaluation of federated search performance in P2P networks with more realistic settings requires a testbed of larger scale containing real documents. We make our contribution by creating a new P2P testbed containing thousands of text digital libraries from the TREC WT10g dataset, which is one of the largest to be used so far for research on P2P systems. The size of our P2P network is comparable to what might be encountered in medium-sized corporate environments ("*enterprise search*"). In addition to providing content, our P2P testbed also includes tens of thousands of automatically-generated queries, which have been proved by our experimental results to be useful in evaluating federated search performance of P2P networks. This large set of queries also provides a convenient and useful resource in studying how a network can learn from past queries and evolve in order to improve the search performance over time.

Based on our P2P testbed, we evaluate various components of our network search model against existing common alternatives and conclude that the network search model provides more sophisticated search techniques and offers a better combination of accuracy, time efficiency and cost efficiency for full-text federated search in P2P networks.

# *Chapter 6*

# 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 facilitates effective and efficient full-text federated search.

As already discussed in Section 3.2, the topology of a hierarchical P2P network has the components of hub-provider topology, hub-consumer topology, and hub-hub 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 in order to improve the efficiency and effectiveness of query routing. The evolution of hub-consumer topology aims at reducing the effective search radius (TTL) by directly connecting consumers with focused interests to hubs that cover content areas most similar to the individual interests (interest-based locality), and connecting consumers with diverse information requests to hubs that can reach diverse content areas in a few hops (topic diversity). 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.

In the following three sections, we describe in detail our topology evolution algorithms that enable the three components of a hierarchical P2P network topology to evolve into ones that achieve the respective objectives. Section 6.4 discusses our experiments on topology evolution, followed by a summary in Section 6.5.

## 6.1 Evolution of Hub-Provider Topology

Because all the providers that connect to a hub naturally form a local cluster, a hub-provider topology with content-based locality can be constructed by requiring each hub's neighboring providers to form a cohesive content-based cluster. One way to do that dynamically as providers join the network is to connect each provider to those hubs that have highest similarities between the content areas they cover and the content the provider provides. Topology evolution algorithms proposed in (Crespo and García-Molina 2002a) (Schlosser et al. 2002) (Löser et al. 2003) describe a content area using schema-based representations based on a global classification hierarchy or ontology. This approach to deciding and representing the content area covered at each hub requires the content space to be partitioned exhaustively into a number of content areas, which may be difficult to satisfy for the environments containing text digital libraries of heterogeneous and open-domain contents. Another approach to describing a hub's content area is to use a full-text resource description obtained by aggregating the resource descriptions of its neighboring providers, which is proposed in our network search model. The similarity between a provider's content and a hub's content area can be measured by the similarity between their resource descriptions. This approach has the advantages that it not only supports full-text federated search, but also enables convenient representations of heterogeneous and open-domain contents. Therefore, we adopt it in our development of hub-provider topology evolution.

In consideration of the unique characteristics of a network with hierarchical P2P network architecture and heterogeneous, open-domain contents, our design of an evolution algorithm for hub-provider topology is guided by the following principles.

1. *Decentralization*. When a provider requests to join a hierarchical P2P network, since there is no centralized server to decide which clusters it should join, decisions must be made in a decentralized manner on which hubs the provider can connect to so as to maintain content-based locality in hub-provider topology.

2. *Role differentiation*. Because in a hierarchical P2P network, hubs may have more processing power and connection bandwidth, they should play a more active role in the evolution of hub-provider topology so that providers with limited resources can simply wait for hubs' decisions passively.

3. *Adaptive clustering*. 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. Therefore, in contrast to an explicit fixed clustering policy, each hub should use an implicit adaptive clustering criterion, which is more autonomous and self-adjusting.

4. *Efficiency*. Because constructing and adjusting hub-provider topology requires additional communication costs and computations, it is cost-efficient to distribute the resource description of a provider only to those hubs whose content areas are likely to match the provider's content and shield hubs with dissimilar content areas from unnecessary overhead. Content-based distribution of providers' resource descriptions is also desired for scalability, because if each provider's resource description is broadcast to all hubs, each hub receives the resource description of each provider even if their contents are completely different, which will be a problem when the number of providers joining the network becomes large.

To decide the hub-provider connections for a provider, a joint effort of the provider and the hubs that receive this provider's resource description is required. Each provider is responsible for providing its resource description to the network and making final decision on which hubs to connect to. Each hub that receives the provider's resource description is responsible for using a matching function to calculate the degree of match between the provider's resource description and the hub's resource description, providing this information to the provider to facilitate its decision making, and distributing the provider's resource description to other hubs. Content-based distribution of the provider's resource description at the hub level is conducted in a way similar to query routing in full-text federated search. Hubs need to know about the contents covered in their neighborhoods so as to decide where to distribute the provider's resource description. Since hubs need to collect neighborhood content information anyway for hub-hub query routing, content-based distribution of the provider's resource description adds few additional costs.

Naturally a hub needs a similarity threshold to decide whether to accept a provider into its cluster and integrate the provider's content into its content area. Because the degree of content-based locality in a hub-provider topology depends on the cohesion of the content-based cluster at each hub, making each content-based cluster as cohesive and homogenous as possible becomes a major incentive to set a tight threshold. However, if all hubs have tight thresholds, it is quite likely that a large number of providers will have difficulty in finding hubs that are willing to accept them. If as a solution to the problem, new content-based clusters are created to accommodate these providers, the network will end up having too many fragmented content areas. Therefore, a more sophisticated solution is needed in order to create a reasonable number of cohesive content-based clusters.

Based on our experience from earlier experiments, the degree of similarity between a provider's content and a content area has three levels: very similar, marginally similar, and dissimilar. Intuitively, if a provider's content is very similar to the content area covered at a hub, integrating the provider's resource description will not dramatically change the description of the hub's content area. In contrast, if a provider's content is only marginally similar to the content area covered at a hub, integrating the provider's resource description may introduce "noise" which can destroy the homogeneity of the hub's description of its content area. However, it may still be beneficial to accept the provider into the hub's content-based cluster when the provider has no better cluster to join and the differences between its content and existing content areas in the network are not significant enough to trigger creation of a new content area. Because a hub needs to distinguish between the acceptance of a very similar provider and that of a marginally similar one, one additional threshold is required. The hub can use the lower threshold to distinguish between similar and dissimilar

contents in order to decide whether to accept a provider into its content-based cluster, and use the upper threshold to further distinguish between very similar and marginally similar contents in order to decide whether to integrate the provider's resource description into its description of the covered content area. The rationale is that by dividing members of a content-based cluster into core (very similar ones) and periphery (marginally similar ones) and only using the contents of core members to represent the content area at the hub, the centroid of the content area remains relatively stable without drifting in an uncontrollable manner. We refer to core members of a content-based cluster as *citizens* and periphery as *permanent residents* of the cluster. Citizens and permanent residents both contribute their resource descriptions to the resource description of the hub associated with the cluster for full-text federated search, but the description of the hub's content area for hub-provider topology evolution ("*core resource description*") only aggregates its citizens' resource descriptions. In addition to citizens and permanent residents, there may be a third class of providers connecting to a hub, which we refer to as *visitors*. Visitors are those providers that are temporarily attached to the hub when they join the network and before they become citizens or permanent residents of one or more hubs' clusters. Strictly speaking, a hub's visitors do not belong to the content-based cluster at the hub, but the hub may provide some services for them such as hub-provider query routing.

When a provider *P* 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, its join process proceeds as follows. Initially, *P* finds an initial set of hubs by querying host-cache servers or by pinging the network. It arbitrarily chooses a hub from the list to connect to temporarily, which is responsible for acquiring *P*'s resource description and starting its distribution among hubs within a radius specified by the TTL of the message. Each non-empty hub[10] that receives *P*'s resource description executes two operations. First, the hub compares *P*'s resource description with its own core resource description using the matching function, and issues a connection offer together with the calculated similarity value to *P* if it qualifies as a citizen or a permanent resident based on the output of the matching function. Second, the hub selects some neighboring hubs to receive *P*'s resource description based on the match between *P*'s resource description and the resource description of each of its neighborhoods. Each empty hub that receives *P*'s resource description directly distributes it to its hub neighbors without conducting a local matching operation. *P* collects the connection offers from hubs and connects to the hubs that accept it as a citizen until its maximum connection capacity is reached. The core resource descriptions of the connected hubs are updated to integrate *P*'s resource description. If *P* only receives connection offers accepting it as a permanent resident, it connects to the most similar hub it receives an offer from. The number of permanent residences for a provider is restricted to one to minimize its negative effect on the cohesion of a content-based cluster it is only marginally similar to. If *P* does not receive any connection offer after waiting for a sufficiently long time, it assumes that it has been rejected by all the non-empty hubs in the network and requests an empty hub to initiate a new content-based cluster and accept it as a citizen. Initiating a new content-based cluster only when a new content area emerges (signaled by a provider whose content is dissimilar to all existing content areas in the network) avoids generating multiple content-based clusters that locate in different parts of the network but actually belong to the same content area. If no empty hub is available, *P* connects to a hub designated for "miscellaneous" contents.

Due to the dynamic nature of a P2P network, the content area covered by each hub may change as providers join (and leave) the network or update their contents. Because the status (citizen or permanent resident) of a provider in a hub's cluster is determined by the degree of match between the provider's and the hub's resource descriptions, whenever there is an update in either one's resource description, the provider's status in the hub's cluster may change. Therefore, each hub needs to reevaluate the statuses of its providers periodically or when the difference between its old and new resource descriptions is significant. If a provider becomes a citizen, the hub integrates the provider's resource description into its resource description. If a provider changes from a citizen to some other status, the hub segregates the provider's resource description from its resource description. If a provider changes to a visitor, the hub drops the established connection with this provider. A provider with all of its connections dropped must rejoin the network.

[10] A hub is "non-empty" if it has a non-empty content-based cluster, i.e., if it connects to at least one provider. Otherwise, it is empty.

## 6.2 Evolution of Hub-Consumer Topology

A consumer *C* may perform only *ad-hoc search* for which successive information requests are not closely related conceptually, only *focused search* for which information requests during a period of time are closely related to one another to indicate its persistent interests in specific topics, or both focused search and ad-hoc search. For the benefit of ad-hoc search, *C* should connect to the hubs that can reach diverse content areas in a small number of hops. To optimize the performance of focused search, *C* should connect to the 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 with the topics. We propose our approach to hub-consumer topology evolution in Chapter 7 as part of the dissertation research.

## 6.3 Evolution of Hub-Hub Topology

As defined in Section 3.2.3, 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 (Watts and Strogatz 1998). Our evolution model of hub-hub topology with content-based small-world properties has a number of specific features that distinguish it from other topology evolution models developed either for the World Wide Web (Barabási et al. 1999) (Menczer 2002) (Manna and Kabakcioglu 2003) (Clauset and Christopher 2004) or for P2P networks with restrictive lattice or hierarchical network models (Kleinberg 2001) (Kleinberg 2003), which are shown below.

1. *Dynamic adaptation.* A P2P network is dynamic in nature; hubs may join and leave the network, the content area of each hub may change over time as providers connect and disconnect to it, and new content areas may emerge in the network. Hence each hub needs to adjust its connections to other hubs periodically to accommodate these changes and maintain content-based small-world properties.

2. *Utilization of limited local content information*. 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.

3. *Degree balancing.* Because connections at the hub level are major channels for query routing and distribution of resource descriptions, hubs that are highly connected (i.e., with a high degree) may become potential bottlenecks which will restrict the information flow in the network. In addition, an attack resulting in the removal of these highly connected hubs could cripple the network. In our approach, special effort is made to balance degrees without undermining content-based small-world properties.

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 chooses its hub neighbors at the moment. *H*'s resource description is initialized when it responds to a provider's request to start a new content-based cluster. Then *H* connects to the provider and the provider's resource description becomes *H*'s resource description.

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 hubs within two hops from it;

2. $H$ connects to $M_c$ most similar hubs whose incoming hub connection capacities have not reached their maximally allowed values, where $M_c$ 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, $H$ 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 $M_r$ dissimilar hubs that have not reached their maximum incoming hub connection capacities with probability:

$$\mathrm{P}(H_i \in \{H_j : GD(H, H_j) \le 2)\} \mid CD(H, H_i) \ge \rho^*) = cCD(H, H_i)^{-\beta} \qquad (6.1)$$

   where $M_r$ is the maximum number of outgoing long-range 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 the 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 periodically adapting each hub's outgoing[11] connections at the hub level, hub-hub topology effectively maintains content-based small-world properties in a dynamic environment. Since each hub adjusts its connections only based on its local knowledge of 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. 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 2003).

## 6.4 Experiments

The effectiveness of our topology evolution algorithms to build a hierarchical P2P network topology with desired search-enhancing properties was evaluated by simulation. We refer to the topologies constructed using our topology evolution algorithms as *content-based topologies*. The constructed hierarchical P2P network was studied in order to evaluate i) whether the content-based hub-provider topology exhibited content-based locality and if it was effective in enhancing search performance compared with a random hub-provider topology, ii) the effectiveness and efficiency of content-based distribution of providers' resource descriptions compared with broadcasting in the evolution of the content-based hub-provider topology, and iii) whether the content-based hub-hub topology exhibited content-based small-world properties and if it was effective in enhancing search performance compared with a random hub-hub topology. We describe our experimental settings in the next section, followed by experimental results and analysis in Sections 6.4.2 and 6.4.3.

### 6.4.1 Experimental Settings

We used the 2,500 digital libraries defined in our P2P testbed (Section 5.1.1) as providers. The number of hubs in the hierarchical P2P network was chosen to be 50 so that we could have a sufficient number of hubs to study the evolution

[11] 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.

of the hub-hub topology as well as a reasonable size range for content-based clusters of the hub-provider topology. The last non-empty hub during the simulation was designated for a cluster of "miscellaneous" contents.

The hierarchical P2P network was initialized to be a network of 50 empty hubs randomly connecting with one another without any providers. Because we focused on the join process for the moment, the experiments assumed that all the hubs remain active in the course of simulation and no providers depart the network after joining it. The simulation stopped after all of the 2,500 providers had joined the network.

Because large-scale P2P networks are typically sparse, and it is quite expensive for each hub to acquire and maintain neighborhoods' resource descriptions for a large number of hub neighbors, we chose small values for various maximum connection capacities to simulate the topology evolution of a sparse network. The maximum number of hubs a provider could connect to was restricted to 5. A hub's maximum number of outgoing local hub connections $M_c$ was 2 and its maximum number of outgoing long-range hub connections $M_r$ was 3 minus its actual number of outgoing local hub connections. Additional experiments to study the sensitivity of topology evolution to various settings of maximum connection capacities are scheduled for the dissertation research.

Simple cycle detection was used to avoid cycles of length 3 in hub-hub connections to improve the accuracies of neighborhoods' resource descriptions and the efficiency of query routing.

The lower and upper thresholds for hub-provider topology evolution were empirically set to $-2.0$ and $-1.0$ respectively. Each hub distributed a provider's resource description to a neighboring hub when the K-L divergence between the neighbor's and the provider's resource descriptions was less than 1.5. The threshold $\rho^*$ to distinguish between similar and dissimilar hubs for hub-hub topology evolution was $-1.0$. The exponent $\beta$ for the power-law distribution was 2.0.

To avoid distributing a provider's resource description in the network indefinitely, each message carrying a provider's resource description had an initial time-to-live (TTL) value of 6.

### 6.4.2 Evaluating Hub-Provider Topology

Because the providers that connect to a hub form a cluster, the quality of a content-based hub-provider topology can be measured by the quality of the corresponding clusters. A hub-provider topology generated by randomly connecting each of the 2,500 providers to the hubs in the network with the constraints on connection capacity described in the previous section was used as the baseline. To construct the baseline topology, each provider was assumed to be aware of all the 50 hubs in order to randomly select a subset of them for connection according to the uniform distribution, which required global knowledge on the identities of hubs.

In addition to the random hub-provider topology, the content-based hub-provider topology constructed using content-based distribution of providers' resource descriptions was compared against a topology built by broadcasting. Broadcasting providers' resource descriptions to all the hubs would be expected to construct the best hub-provider topology because it essentially conducts an exhaustive search for each provider to find the best cluster it can join. However, this method becomes very expensive as the network becomes large. Content-based distribution of providers' resource descriptions avoids distributing providers' resource descriptions to hubs covering dissimilar content areas and therefore should be more efficient and scalable. The experiments investigated whether the two methods of distributing providers' resource descriptions yielded hub-provider topologies with comparable qualities.

Table 6.1 shows the characteristics of the clusters generated using different methods. The three numbers in each cell indicate the minimum, mean, and maximum values of the corresponding measure. Although the average number of clusters a provider belonged to ("# hub connections per provider") didn't vary much for different topologies, how providers were distributed was dramatically different comparing the content-based hub-provider topologies with the random hub-provider topology. One indication of this significant change is the distribution of cluster sizes ("# provider connections per hub"). The random hub-provider topology gave a more-or-less uniform distribution of cluster sizes, while the distributions of cluster sizes for the content-based hub-provider topologies were much more skewed. A skewed distribution of cluster sizes would support a better content-cased clustering than a uniform one because the

**Table 6.1 The characteristics of the clusters of different hub-provider topologies.**

| Method | # hub connections per provider | # provider connections per hub | hub-provider divergence per hub |
|---|---|---|---|
| Content-based hub-provider topology + hub broadcasting providers' descriptions | 1/2.71/5 | 1/135/712 | 0/0.54/0.90 |
| Content-based hub-provider topology + content-based distribution of providers' descriptions | 1/2.73/5 | 1/136/634 | 0/0.61/1.12 |
| Random hub-provider topology | 1/2.68/5 | 108/134/161 | 1.08/1.19/1.29 |

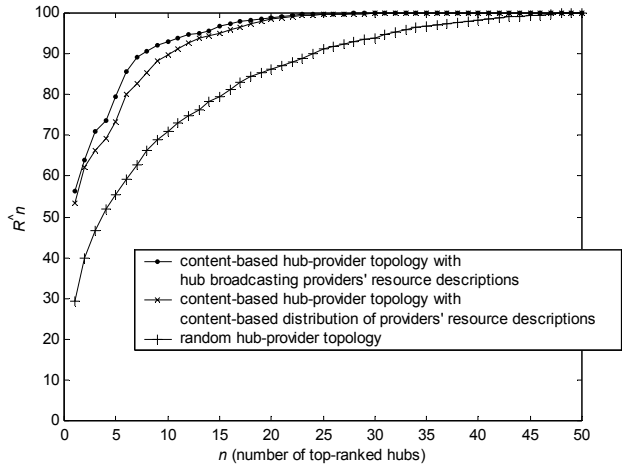


**Figure 6.1 The cumulative distributions of the relevant documents for the 100 TREC queries among hubs with different hub-provider topologies.**

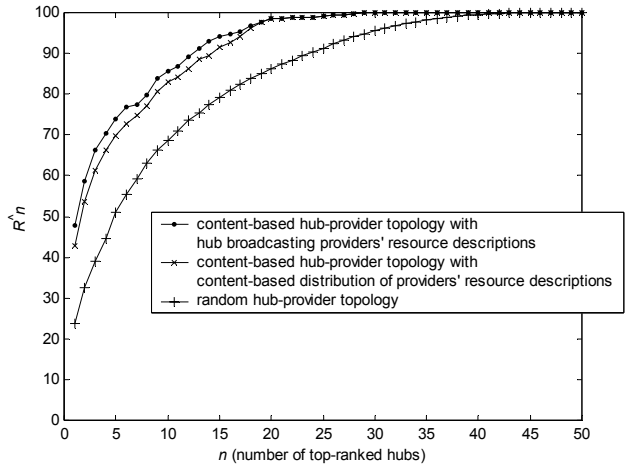


**Figure 6.2 The cumulative distributions of the relevant documents for the 1,000 WT10g queries among hubs with different hub-provider topologies.**

contents of the 2,500 digital libraries are known to be skewed. The average K-L divergence between a hub's resource description and the resource description of each of its providers measured the cohesion of the cluster at this hub. The big difference between the values of "hub-provider divergence per hub" shows that the clusters generated by the

content-based hub-provider topologies were much more cohesive than the clusters generated by the random hub-provider topology.

Another way to measure the quality of a hub-provider topology is to measure the quality of content-based locality in the network. If the network exhibits good content-based locality, then most contents relevant to an information request are expected to be concentrated in a small part of the network so that only a small number of hubs need to be contacted in order to obtain sufficient relevant documents. We used the metric of $\hat{R}_n$ to measure the concentration of relevant documents among different hubs. Hubs were ranked by the number of relevant documents contained in their clusters, and $\hat{R}_n$ was the percentage of the total number of relevant documents that had been accumulated via the $n$ top-ranked hubs. This metric has been used to evaluate the effectiveness of resource ranking in traditional distributed information retrieval (French et al. 1998). We used the 100 TREC queries with the relevance assessments and the 1,000 WT10g queries with the "single collection" baseline (Section 5.1.3). Figures 6.1 and 6.2 show the results for different hub-provider topologies, averaged over the 100 TREC queries and the 1,000 WT10g queries respectively. From the figures we can see that the degree of content-based locality in the network with a content-based hub-provider topology was consistently greater than that in the network with a random hub-provider topology. For example, for TREC queries, only around 5-6 top-ranked hubs were needed in order to cover an average of 80% of the relevant documents for the network with a content-based hub-provider topology. In contrast, the network with a random hub-provider topology required at least 15 top-ranked hubs in order to cover the same percentage of the relevant documents. This indicates that the content-based hub-provider topologies were indeed able to concentrate most relevant contents in a few clusters so that a small number of good hubs would be sufficient to guarantee a high recall.

Table 6.1 and Figures 6.1 and 6.2 also show that the quality of the hub-provider topology constructed using content-based distribution of providers' resource descriptions was comparable to that using broadcasting to distribute providers' resource descriptions. However, with content-based distribution of providers' resource descriptions, on average of about 41% of the total number of hubs were actually contacted to consider the join request of a provider, making the topology evolution more efficient and scalable.

In summary, our topology evolution algorithm for hub-provider topology was effective in constructing a hub-provider topology with a high degree of content-based locality in an efficient and scalable manner.

### 6.4.3 Evaluating Hub-Hub Topology

The first measure to evaluate a small-world topology would be characteristic path length. The average shortest path length between any two hubs (characteristic path length) was 3.11 for the content-based hub-hub topology compared with 3.06 for a random hub-hub topology with the same connection density. Thus the content-based hub-hub topology had a short characteristic path length. However, it did not have a high clustering coefficient value because cycles of length 3 were deliberately avoided in hub-hub connections for efficient query routing. Therefore, a different approach is needed to evaluate whether the content-based hub-hub topology exhibited content-based small-world properties.

If a hub-hub topology exhibits content-based small-world properties, when an information request is initiated from a "relevant" neighborhood, most relevant documents should be covered by nearby hubs due to local clustering of hubs with similar content areas. When an information request is initiated from a "non-relevant" neighborhood, the request should only need to travel along a short path to reach a "relevant" neighborhood thanks to the short characteristic path length between hubs. Therefore, whether a hub-hub topology has content-based small-world properties can be measured by the effectiveness and efficiency of hub-hub query routing with different search radiuses when queries start from different neighborhoods.

In order to evaluate the hub-hub topology, we conducted resource selection of hubs for federated search in the hierarchical P2P network for a set of queries and evaluated how fast queries could be routed to the hubs covering most relevant documents. Upon receiving a query message, a hub in the network used content-based resource selection of hubs to select the two top-ranked hub neighbors to further forward the query message. How fast a query could be routed to a "relevant" neighborhood was measured by the percentages of the relevant documents covered by those hubs that were reached when the query message was initialized with different time-to-live (TTL) values. The content-based

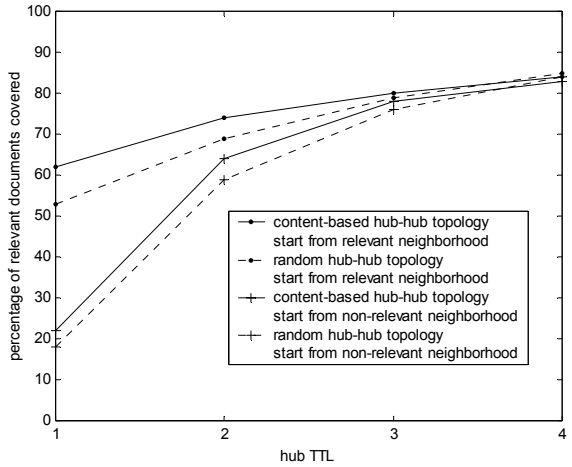


**Figure 6.3 The percentages of the relevant documents covered by those hubs that received the TREC query with different hub-hub topologies and search radiuses.**

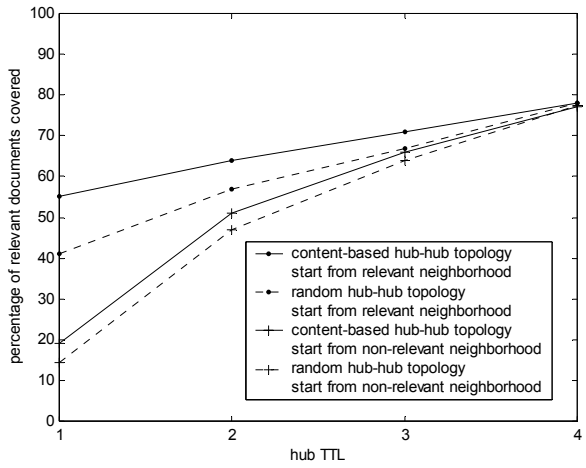


**Figure 6.4 The percentages of the relevant documents covered by those hubs that received the WT10g query with different hub-hub topologies and search radiuses.**

hub-hub topology was compared against the random hub-hub topology with the same average number of hub neighbors per hub.

Figure 6.3 compares the average results of the 100 TREC queries using the content-based and random hub-hub topologies for the cases when each query was issued through a hub ranked among the top five hubs in the number of relevant documents covered for this query ("relevant neighborhood") and the cases when each query was issued through a hub that didn't cover any relevant documents for this query ("non-relevant neighborhood"). Figure 6.4 shows the results for the 1,000 WT10g queries. The figures indicate that if a query message only traveled one hop to reach at most three hubs (one initial hub and two of its direct hub neighbors), which neighborhood to start from made a big difference. The content-based hub-hub topology had consistently better performance than the random hub-hub topology when queries were initiated from relevant neighborhoods because similar contents (and therefore more relevant contents) were near to one another in the content-based hub-hub topology but most certainly not so in the random hub-hub topology. When queries were initiated from non-relevant neighborhoods, both topologies could not locate a substantial amount of relevant documents for any query that started from a non-relevant content area and

traveled only one hop. However, the content-based hub-hub topology still slightly outperformed the random hub-hub topology.

As query messages traveled farther, the performance difference between different cases quickly became smaller. After three hops, similar percentages of the relevant documents were covered. This indicates that in all cases the query message was able to reach a "relevant" neighborhood within three hops at the hub level. However, the random hub-hub topology required around 12% more query messages than the content-based hub-hub topology.

Overall, the content-based hub-hub topology enabled better query routing performance than the random hub-hub topology with the same connection density when the TTL was small ($\leq 2$). Their performance was similar when the TTL became large ($\geq 3$), but search in the content-based hub-hub topology was more efficient. This demonstrates the effectiveness of our hub-hub topology evolution algorithm in constructing a hub-hub topology with content-based small-world properties to support efficient and effective query routing in a hierarchical P2P network. Because the size of the hub-hub topology was relatively small with 50 hubs, the difference in the coverage of relevant documents between the content-based hub-hub topology and the random hub-hub topology was not as big as expected. However, as the number of hubs increases and the contents as well as the queries become more diverse, the advantage of the content-based hub-hub topology over the random hub-hub topology will become more salient.

## 6.5 Summary

In this chapter, we describe a network evolution model to construct a hierarchical P2P network topology with search-enhancing properties such as content-based locality, interest-based locality, and content-based small-world properties (described in our network overlay model) dynamically and autonomously so that full-text federated search can be carried out efficiently and effectively using our network search model.

Previous approaches to constructing a network topology with content-based locality either uses predetermined policies to cluster peers into content-based clusters and establishes connections based on cluster membership (Crespo and García-Molina 2002a) (Schlosser et al. 2002) (Löser et al. 2003), or starts from a random topology and forms content-based locality by every peer seeking to rewire its connections to other peers with similar contents (Khambatti et al. 2002) (Asvanund 2004). The former approach is only applicable to limited-domain content; the latter ignores the differences in peers' connection bandwidth and processing power. Both approaches would not work well with open-domain full-text representations of content that require nontrivial content propagation and similarity measurement for topology evolution. In contrast, our network evolution algorithm for hub-provider topology works effectively in this case by using implicit, adaptive clustering policies instead of explicit, static ones and assigning most work to hubs to fully utilize their high connection bandwidth and processing power. Our algorithm is also shown to be more efficient and scalable with its use of selective propagation of providers' content information at the hub level.

Although some previous research recognizes the existence of focused search (i.e., a consumer's subsequent information requests are on the same topics as its earlier requests) in P2P networks and designs topology evolution and/or search algorithms to take advantage of it (Sripanidkulchai et al. 2003) (Shao and Wang 2004), the distinction between ad-hoc search and focused search has not been studied for federated search in P2P networks, and there has not been any effort on constructing a network topology to optimize the performance for both types of search simultaneously. As part of the dissertation research, we will provide the first attempt to solve this problem.

To provide an adaptive, cost-efficient solution to constructing a hub-hub topology with content-based small-world properties and good navigability, the network evolution algorithm must satisfy several requirements: i) peer distance is defined based on content similarity, ii) each hub must establish its connections based on a limited local view of the network, iii) hubs should be able to adjust their connections dynamically, and iv) the distribution of connection lengths should be power-law instead of uniform. The topology evolution algorithms previously developed for P2P networks or for the World Wide Web either rely on simplified network models with unrealistic assumptions on the content and the amount of information available for topology construction (Watts and Strogatz 1998) (Kleinberg 2003) (Manna and Kabakcioglu 2003) (Clauset and Christopher 2004), or ignore the necessary conditions for a small-world topology to be navigable (Merugu et al. 2004) (Sakaryan and Unger 2003). To the best of our knowledge, there has not been a single topology evolution algorithm capable of satisfying all the above requirements simultaneously. Taking inspirations

from earlier work, our network evolution algorithm for hub-hub topology not only fulfills all the requirements, but also takes extra steps to avoid potential bottlenecks of information flow and reduce the network's susceptibility to malicious attacks on highly connected hubs by balancing hub degrees. Experimental results show that our algorithm is indeed effective in constructing a hub-hub topology with content-based small-world properties and good network navigability.

Although the experimental results presented in Section 6.4 demonstrate the effectiveness of our algorithms for hub-provider and hub-hub topology evolution, further study is required to analyze the stability and scalability of the constructed network. We briefly describe this piece of future work in Chapter 7.

# *Chapter 7*

# DISSERTATION RESEARCH

Previous chapters introduce new research on full-text federated search in hierarchical P2P networks with a network overlay model, a network search model and a network evolution model in an integrated framework. In this chapter, we discuss future work scheduled for the dissertation research on providing more practical solutions to full-text federated search in real operational P2P environments. Section 7.1 presents our plan on enhancing the network evolution model. Section 7.2 proposes a network administration model to address issues such as load balancing and fast fault handling. Section 7.3 discusses future work on developing a larger-scale P2P testbed and improving a prototype P2P application for evaluation in more varied P2P environments. Expected contributions of the dissertation are summarized in Section 7.4.

## 7.1 Enhancement to the Network Evolution Model

The network evolution model described in Chapter 6 provides algorithms for hub-provider and hub-hub topology evolution. Preliminary experimental results show that the algorithms are effective in constructing a hierarchical P2P network topology with desired properties to support effective and efficient full-text federated search. To further facilitate high-performance full-text federated search in large-scale, dynamic P2P networks, we will enhance the network evolution model by i) developing an algorithm for hub-consumer topology evolution, and ii) investigating the stability and scalability of topology evolution.

### 7.1.1 Evolution of Hub-Consumer Topology

As mentioned in Section 6.2, a consumer *C* may issue queries to perform two different types of search, namely *ad-hoc search* and *focused search*, depending on whether queries are related to its persistent interests in specific topics. The best set of hubs *C* should connect to for efficient and effective federated search may vary with queries of different types and topics.

For constructing a hub-consumer topology to support efficient and effective ad-hoc search, there exists an easy but effective solution. Because content-based hub-hub topology with small-world properties (Section 6.3) generally enables each hub to reach any other hub by following a short path, when combined with a small search radius, random hub-consumer connections would be sufficient for simultaneously achieving the efficiency and effectiveness of locating relevant contents in ad-hoc search.

Focused search can be made much more efficient than ad-hoc search by taking advantage of interest-based locality. If *C* directly connects to the hubs covering content areas most similar to its interests, then due to interest-based locality, most relevant contents can be covered by these hubs so that hub-hub query routing can be dropped. An approach to discovering such hubs with minimum additional cost is for *C* to learn from the responses it received for the focused search it performed in the past. This approach has been shown to be effective in (Sripanidkulchai et al. 2003) and (Shao and Wang 2005), but previous research didn't consider the case that if the topics *C* is interested in belong to quite different content areas, a separate "optimal" set of hub-consumer connections may be required for each topic.

For hub-consumer topology evolution, we will progressively develop solutions to the following scenarios.

1. *Focused search of a single topic*. If all the queries of a consumer *C* are related to one topic, then the single "optimal" set of hubs it should connect to is a set of hubs that are most likely to provide quality results for future queries. In previous research, the likelihood of providing quality results is estimated by the number of documents returned for past queries (Sripanidkulchai et al. 2003) (Shao and Wang 2005). This approach may work well for known-item search which typically either returns relevant documents or returns no document at all. However,

full-text federated search may return a lot of non-relevant documents, so the total number of documents returned would not be a good indication of retrieval quality. Because it has been shown that the result merging algorithm proposed for full-text federated search has satisfactory performance in identifying relevant documents and giving them top ranks in the merged results (Section 5.3.3), a better estimate of a hub's likelihood of providing quality results would be the number of documents that were returned by the hub and appeared in the top of the merged results for past queries. Therefore, we propose the following procedure for the evolution of a consumer *C*'s connections if it performs focused search of a single topic. Initially *C* randomly connects to several hubs to issue some queries with a search radius of multiple hops. Then it collects merged results, and counts for each responding hub the accumulated number of documents that are returned by that hub and appear in the top of the merged results. The learned statistics can be used to rank hubs and *C* adjusts its connections to connect to the top-ranked hubs. We will evaluate the effectiveness of using the number of the top-ranked documents compared with using the total number of documents returned in learning hub-consumer connections.

2. *Focused search of multiple topics.* When the queries issued by a consumer *C* belong to multiple topics, a separate "optimal" set of hub-consumer connections may be required for queries of each topic. So queries need to be distinguished by topics both for learning *C*'s "optimal" connections for each topic, and for consumer-hub query routing to selective hubs according to topics. One way that groups queries by topics explicitly is to conduct content-based query clustering. *C* uses training queries together with their top-ranked responses as representations to learn each cluster's "optimal" set of hub-consumer connections. Any new query classified to a cluster is routed to the associated "optimal" set of hub-consumer connections. Another approach without explicit query clustering or classification is to use the nearest neighbor method. The best hubs *C* should connect to for training queries are decided on a query-by-query basis. The "optimal" set of hub-consumer connections for a new query is determined by weighing the best hubs for past queries using the similarities between past queries and the new query, and selecting hubs with highest weights. We will compare the effectiveness of these two approaches with experiments.

3. *Focused search of multiple topics mixed with ad-hoc search*. Queries for focused search and those for ad-hoc search need to be distinguished because ad-hoc search requires a larger search radius (TTL) in order to reach more hubs covering diverse content areas. If content-based query clustering is used, queries that belong to big clusters can be treated as queries for focused search and those that don't as queries for ad-hoc search. If the nearest neighbor approach is adopted, then a threshold may be required to determine whether a query is sufficiently dissimilar from other queries and can be considered a query for ad-hoc search. We will conduct experiments to study whether queries for focused search and those for ad-hoc search can be distinguished with high accuracy, and how the accuracy of federated search will be affected.

Although an initial period is required for all the cases above during which *C* can only randomly connect to some hubs, issue queries with a multiple-hop search radius, and collect responses before it accumulates enough information and starts adjusting its connections, this period is expected to be longer as the scenario becomes more complex. We will investigate the minimum length of the initial period required for each scenario to learn effective hub-consumer connections.

### 7.1.2 Stability and Scalability of Topology Evolution

The topology of a P2P network may be under constant change due to its dynamic nature. It is desirable that certain properties which have great impact on the performance of federated search should remain stable during topology evolution. For example, maintaining content-based small-world properties in the hub-hub topology is necessary to guarantee effective and efficient hub-hub query routing. Our previous experiments only evaluated full-text federated search performance in the constructed network at a single point in time, which told us little about what changes the network underwent at different time slices. We will conduct more experiments to investigate issues such as when the contents of each cluster formed by the hub-provider topology become stabilized, and how the distribution of connection lengths changes during the periodic adaptation of the hub-hub topology. Seeking answers to these questions may lead to new discoveries which can help us to further refine our topology evolution algorithms.

To provide practical solutions to the evolution of large-scale P2P networks, it is important to study the behavior of the network when it grows to a large size. Because hub-hub query routing plays a more important role in efficient and

effective query routing in a large-scale P2P network containing peers with bounded connection capacity, we are particularly interested in investigating how different graph metrics of the hub-hub topology change as the number of hubs under various constraints on connection capacity increases in the network. Such graph metrics may include diameter, characteristic path length (the average shortest path length between any two hubs), degree distribution, and distribution of connection lengths.

## 7.2 Network Administration Model

A network administration model is responsible for monitoring the conditions of the network and fixing problems such as traffic congestion and peer failures. Without the network administration model, approaches in the network search model and the network evolution model can still be applied in real operational P2P environments for full-text federated search, but their performance will be discounted by the often uncoordinated and unpredictable behaviors of peers and their connections. The duty of the network administration model is to assure a fast and smooth execution of federated search and network evolution. We plan to focus on two problems in developing the network administration model: i) load balancing to avoid traffic congestion, and ii) fast fault handling. Solutions to these problems must not rely on a central authority due to the decentralized nature of our network overlay model. Compared with centralized network administration that is easy to implement, decentralized administration requires more careful planning and coordination.

### 7.2.1 Load Balancing

A search/evolution hotspot can become a bottleneck of search/evolution if the peer in the hotspot cannot efficiently handle the workload with its connection bandwidth and/or processing power. A provider may be overloaded if the contents it provides are popular by demand. A hub may form a search/evolution hotspot if the content area it covers is popular by demand/supply, or if it is the pivot of busy routing paths among hubs. The network administration model should provide methods to dynamically monitor the load in the network and detect hotspots. It should also provide solutions to alleviate the burdens of peers in the hotspots. For example, a provider's index of popular contents may be cached at its neighboring hubs so hubs can handle some of the query load on behalf of the provider. Multiple hubs may coordinately share the responsibility of serving providers that belong to a popular content area. Additional regulations may be required for routing queries or the resource descriptions of newly joined providers in order to balance traffic along different paths. We will study previous demand-based, supply-based, and replication-based approaches to load balancing in P2P networks and develop efficient distributed solutions.

### 7.2.2 Fast Fault Handling

Although both hubs and leaves could fail in the network, we focus on hub failures and do not consider failures of individual leaves (providers and consumers) because a hub's failure may affect the network's navigability and the accessibility of other peers, while a leaf's failure does not have such influence on other peers.

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. An isolated provider due to the failure of its connecting hub must rejoin the network by running the join process all over again if it does not have any knowledge of the similarities between its content and the content areas covered other hubs. When a hub loses its local or long-range hub connections due to the failure of its neighboring hubs, it must seek new connections through later topology adaptations if it does not keep a record of the information about other hubs it acquired earlier. 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.

Fast fault handling restores network connectivity and peer accessibility from hub failures by establishing new connections, which may cause changes on the distribution of contents in the network. Because dramatic changes in content distribution result in costly updates in the resource descriptions required by full-text resource selection, any long-distance migration of peer locations in the network topology should be avoided to minimize such changes. To achieve this objective, a more complex redundancy scheme with extra communication and coordination among hubs is

required for each hub to 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.

Experiments will be designed to study the effectiveness of fault handling based on each redundancy scheme and their combinations in terms of the stability of topological structure and search performance.

## 7.3 Evaluation in More Varied P2P Environments

In Chapters 5 and 6 we present the experimental results using the P2P testbed created from the TREC WT10g Web test collection consisting of 2,500 providers and at most 50 hubs, which is comparable to what might be encountered in medium-sized corporate environments. Although the evaluation results using this P2P network of medium scale can provide valuable insights regarding the effectiveness of different approaches to full-text federated search, some aspects of system performance are difficult to measure due to its limited number of peers. For example, we would like to know how different graph metrics such as diameter and degree distribution change in hub-hub topology as the number of hubs scales up, but 50 hubs are far from sufficient to provide any meaningful results with respect to the scalability of hub-hub topology. Therefore, a P2P testbed that is comparable to larger organizations is desired to evaluate the effectiveness of full-text federated search in all aspects. One possible solution is to partition the larger Web research collections used for TREC Web & Terabyte Tracks such as .GOV, .GOV2, and WT100g (VLC2) to create a larger number of text collections (e.g., in the order of $10^5$). Properties of the new testbed such as the distribution of documents in the deduced collections, the distribution of relevant documents with respect to TREC queries or the automatically-generated queries, and the distribution of contents in different topics need to be carefully studied and controlled in order to reflect the real-world application scenario to the greatest extend.

One limitation of our evaluation on full-text federated search in P2P networks is that it only focuses on the logical protocol layer of the network so that the properties of the underlying physical layer and the interactions between logical and physical layers are largely ignored. It simplifies network settings and peer behaviors to a great extend, which might disguise the problems that could emerge in the real world. Ideally we would like to evaluate our models for full-text federated search using a real-world P2P application. A prototype system implementing the basic components of our models has been developed at the Information Networking Institute of Carnegie Mellon University. As part of the dissertation research, we will further improve the system to make a working application with reasonable scalability and use it as a platform to test full-text federated search of text digital libraries in a real P2P environment.

## 7.4 Expected Contributions

Federated search in P2P networks has become a hot 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 development that has been established in 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 full-text search of document contents with relevance-based document ranking. In contrast, full-text ranked retrieval has already 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. Our objective 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. Our development offers one of the first set of practical solutions to enable full-text federated search in P2P networks, which 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 which 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 and

conduct full-text federated search across them offers a single interface to access the "hidden" Web contents that cannot be reached using 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.

The work discussed in this proposal and scheduled for the dissertation research provides the first integrated framework for full-text federated search of text digital libraries using hierarchical P2P networks as a federated search layer. 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 and efficient 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 fully utilizes the functionalities and the 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. A *network administration model* will also be included in the framework for full-text federated search, which will provide decentralized solutions to problems such as load balancing and fast fault handling in order to make full-text federated search more practical in real operational P2P environments.

The dissertation also aims to provide valuable resources for evaluating federated search in a large-scale P2P network with realistic settings, which will hopefully benefit future research in this field. A P2P testbed with a large number of text collections and queries has been developed and used for evaluating existing and new approaches to full-text federated search and providing useful insights to guide future research. Efforts will be made to develop a P2P testbed of an even larger scale and a working P2P application to build a useful evaluation platform with a flavor of the large-scale P2P environments in the real world.

# *Chapter 8*

# SCHEDULE

**September 2005 – October 2005**

Evolution of hub-consumer topology.

**November 2005 – December 2005**

Stability and scalability of topology evolution.

**January 2006 – February 2006**

Load balancing.

**February 2006 – March 2006**

Fault handling.

**April 2006 – June 2006**

Developing a larger-scale P2P testbed to evaluate full-text federated search and network evolution.

**July 2006 – December 2006**

Developing and testing a reasonable-scale P2P system for full-text federated search.

# BIBLIOGRAPHY


Adamic L, Lukose R, Puniyani A and Huberman B (2001) Search in power-law networks. *Physical Review E*, 64(4): 46135-46143.

Asvanund A, Krishnan R, Smith M, Telang R, Bagla S and Kapadia M (2003) Intelligent club management in peer-to-peer networks. In *Workshop on Economics of Peer-to-Peer Systems*.

Asvanund A (2004) Peer-to-peer networks: user behaviors, network effects and protocol extensions. Ph.D. thesis, the Heinz School of Public Policy and Management, Carnegie Mellon University.

Atkeson C, Moore A and Schaal S (1997) Locally Weighted Learning. *Artificial Intelligence Review*, 11(1-5): 11-73.

Barabási A, Albert R and Jeong H (1999) Emergence of scaling in random networks. *Science*, 286: 509-512.

BearShare, http://www.bearshare.com.

Baeza-Yates R and Ribeiro-Neto B (1999) Modern Information Retrieval. ACM Press/Addison Wesley, NewYork, NY.

Bloom B (1970) Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13(7): 422-426.

Borgman C (1999) What are digital libraries? Competing visions. *Information Processing & Management*, 35(3): 227-243.

Buckley C and Voorhees E (2004) Retrieval evaluation with incomplete information. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Callan J (2000) Distributed information retrieval. In Croft W B ed. Advances in Information Retrieval, chapter 5, pp. 127-150. Kluwer Academic Publishers.

Callan J and Connell M (2001) Query-based sampling of text databases. *Transactions on Information Systems*, 19(2): 97-130.

Clauset A and Christopher M (2004) How do networks become navigable? oai:arXiv.org:cond-mat /0304563.

Craswell N, Hawking D and Thistlewaite P (1999) Merging results from isolated search engines. In *Proceedings of the 10th Australasian Database Conference*.

Crespo A and García-Molina H (2002a) Semantic overlay networks for P2P systems. Technical report, Computer Science Department, Stanford University.

Crespo, A. and García-Molina H (2002b) Routing indices for peer-to-peer systems. In *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS)*.

Cuenca-Acuna F and Nguyen T (2002) Text-based content search and retrieval in ad hoc p2p communities. Technical Report DCS-TR-483, Rutgers University.

Dabek F, Kaashoek M, Karger D, Morris R and Stoica I (2001) Wide-area cooperative storage with CFS. In *Proceedings of the 18th ACM Symposium on Operating Systems Principles (SOSP '01)*.

Daswani S and Fisk A Gnutella UDP Extension for Scalable Searches (GUESS) v0.1.

Dury A (2004) Balancing access to highly accessed keys in peer-to-peer systems. In *Proceedings of IEEE International Conference on Services Computing (SCC'04)*.

Edutella, http://edutella.jxta.org.

eDonkey, http://www.edonkey2000.com.

eMule, http://www.emule-project.net.

French J, Powell A, Viles C, Emmitt T and Prey K (1998) Evaluating database selection techniques: A testbed and experiment. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Gnucleus, http://www.gnucleus.com.

Gnutella v0.4, 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, García-Molina H and Tomasic A (1994) The effectiveness of GlOSS for the text database discovery problem. In *Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data*.

Gravano L and García-Molina H (1995) Generalizing GlOSS to vector-space databases and broker hierarchies. In *Proceedings of 21th International Conference on Very Large Data Bases (VLDB'95)*.

Gravano L, Chang C, García-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*.

Hawking D (2000) Overview of the TREC-9 Web track. In *Proceedings of the 9th Text Retrieval Conference (TREC-9)*.

Hull D (1993) Using statistical testing in the evaluation of retrieval experiments. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Intel (2003) Peer-to-peer content distribution: Using client PC resources to store and distribute content in the enterprise. White paper, Intel Information Technology.

IRIS, http://www.project-iris.net/.

Jansen M, Spink A and Saracevic T (2000) Real Life, real users, and real needs: A study and analysis of user queries on the Web. *Information Processing and Management*, 36(2).

JXTA, http://www.jxta.org.

Javasim, http://javasim.ncl.ac.uk.

Kalogeraki V, Gunopulos D and Zeinalipour-Yazti D (2002) A local search mechanism for peer-to-peer networks. In *Proceedings of the 11th International Conference on Information Knowledge Management (CIKM 2002)*.

Karger D and Ruhl M (2004) Simple efficient load balancing algorithms for peer-to-peer systems. In *Proceedings of the 16th Annual ACM Symposium on Parallelism in Algorithms and Architectures*.

KaZaA, http://www.kazaa.com.

Khambatti M, Ryu K and Dasgupta P (2002) Efficient discovery of implicitly formed P2P communities. *Int'l Journal of Parallel and Distributed Systems and Networks*.

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 (2001) Small-world phenomena and the dynamics of information. *Advances in Neural Information Processing Systems (NIPS)*.

Kleinberg J (2003) The small-world phenomenon: an algorithmic perspective. In *Proceedings of 32nd ACM Symposium on Theory of Computing*.

Krishnamurthy B and Wang J (2000) On Network-Aware Clustering of Web Clients. AT&T Labs--Research Technical Memorandum *HA1630000-000101-01TM*.

Krovetz R (1993) Viewing morphology as an inference process. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Le Calv A and Savoy J (2000) Database merging strategy based on logistic regression. *Information Processing and Management*, 36(3): 341-359.

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.

Limewire, http://www.limewire.com.

Lin K and Kondadadi R (2001) A similarity-based soft clustering algorithm for documents. In *Proceedings of the 7th International Conference on Database Systems for Advanced Applications*.

Liu K, Yu C, Meng W, Santos A and Zhang C (2001) Discovering the representative of a search engine. In *Proceedings of the 10th International Conference on Information Knowledge Management (CIKM 2001)*.

Löser A, Naumann F, Siberski W, Nejdl W and Thaden U (2003) Semantic overlay clusters within super-peer networks. In *Proceedings of Information Systems and P2P Computing in Conjunction with the VLDB 2003*.

Lu J and Callan J (2002) Pruning long documents for distributed information retrieval. In *Proceedings of the 11th International Conference on Information Knowledge Management (CIKM 2002)*.

Lu J and Callan J (2003a) Content-based retrieval in hierarchical peer-to-peer networks. In *Proceedings of the 12nd International Conference on Information Knowledge Management (CIKM 2003)*.

Lu J and Callan J (2003b) Peer-to-peer testbed definitions: trecwt10g-2500-bysource-v1 and trecwt10g-query-bydoc-v1. http://www.cs.cmu.edu/~callan/Data.

Lu J and Callan J (2004a) Merging retrieval results in hierarchical peer-to-peer networks (poster description). In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Lu J and Callan J (2004b) Federated search of text digital libraries in hierarchical peer-to-peer networks. In *Peer-to-Peer IR Workshop of the 27th International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Lu J and Callan J (2005) Federated search of text digital libraries in hierarchical peer-to-peer networks. In *Proceedings of the 27th European Conference on Information Retrieval Research (ECIR 2005)*.

Lv C, Cao P, Cohen E, Li K and Shenker S (2002) Search and replication in unstructured peer-to-peer networks. In *Proceedings of ACM SIGMETRICS'02*.

Manku G, Bawa M and Raghavan P (2003) Symphony: Distributed hashing in a small world. In *Proceedings of the 4th USENIX Symposium on Internet Technologies and Systems (USITS)*.

Manna S and Kabakcioglu (2003) A Scale-free network on Euclidean space optimized by rewiring of links. oai:arXiv.org:cond-mat/0302224.

Maymounkov P and Mazières D (2002) Kademlia: A peer-to-peer information system based on the XOR metric. In *Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS 2002)*.

Menczer F (2002) Growing and navigating the small world Web by local content. *National Academy of Sciences,* 99(22): 14014-14019.

Merugu S, Srinivasan S and Zegura E (2004) Adding structure to unstructured P2P networks: the use of small-world graph. *Journal of Parallel and Distributed Computing on Theoretical and Algorithmic Aspects of Sensor*, *Ad Hoc Wireless and P2P Networks*.

Morpheus, http://www.morpheus.com.

MusicNet, http://www.musicnet.com.

Nottelmann H and Fuhr N (2003) Evaluation different methods of estimating retrieval quality for resource selection. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Ogilvie P and Callan J (2001) Experiments using the Lemur toolkit. In *Proceedings of the 10th Text Retrieval Conference (TREC-10)*.

Ratnasamy S, Francis P, Handley M, Karp R and Shenker S (2001) A scalable content-addressable network. In *Proceedings of the ACM SIGCOMM'01 Conference*.

Ratnasamy S, Shenker S and Stoica I (2002) Routing algorithms for DHTs: Some open questions. In *Proceedings of the 1st International P2P Workshop (IPTPS'02).*

Renda M E and Callan J (2004) The robustness of content-based search in hierarchical peer to peer networks. In *Proceedings of the 13th International Conference on Information and Knowledge Management (CIKM'04)*.

RevConnect, http://www.revconnect.com.

van Rijsbergen C (1979) Information Retrieval.

Rohrs C (2001) Query routing for the Gnutella network. http://rfc-gnutella.sourceforge.net.

Rowstron A and Druschel P (2001) Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *IFIP/ACM International Conference on Distributed Systems Platforms*, pages 329-350.

Sakaryan G and Unger H (2003) Topology evolution in distributed P2P networks. In *Proceedings of Applied Informatics (AI 2003).*

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)*.

Schlosser M, Sintek M, Decker S and Nejdl W (2002) A scalable and ontology-based P2P infrastructure for semantic Web services. In *Proceedings of the 2nd IEEE International Conference on P2P Computing (P2P2002)*.

Shareaza, http://www.shareaza.com.

Shao Y and Wang R (2005) BuddyNet: history-based P2P search. In *Proceedings of the 27th European Conference on Information Retrieval Research (ECIR 2005)*.

Si L and Callan J (2003a) Relevant document distribution estimation method for resource selection. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Si L and Callan J (2003b) A semi-supervised learning method to merge search engine results. *ACM Transactions on Information Systems*, 21(4): 457-491.

Si L and Callan J (2004) The effect of database size distribution on resource selection algorithms. Distributed Multimedia Information Retrieval. LNCS 2924, Springer.

Sripanidkulchai K, Maggs B and Zhang H (2003) Efficient content location using interest-based locality in peer-to-peer systems. In *Proceedings of Infocom 2003*.

Stenmark D (2005) Query expansion on a corporate intranet: Using LSI to increase relative precision in explorative search. In *Proceedings of HICSS 2005.*

Stoica I, Morris R, Karger D, Kaashoek M and Balakrishnan H (2001) Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the ACM SIGCOMM'01 Conference*.

Stutzbach D and Rejaie R (2005) Characterizing the two-tier Gnutella topology. In *Proceedings of the ACM SIGMETRICS'05 Conference.*

Swapper.NET, http://www.revolutionarystuff.com/swapper/.

Tang C, Xu Z and Dwarkadas S (2003) Peer-to-peer information retrieval using self-organizing semantic overlay networks. In *Proceedings of the ACM SIGCOMM'03 Conference*.

Tsoumakos D and Roussopoulos N (2003a) Adaptive probabilistic search for peer-to-peer networks. In *Proceedings of the 3rd International Conference on Peer-to-Peer Computing (P2P'03)*.

Tsoumakos D and Roussopoulos N (2003b) A comparison of peer-to-peer search methods. In *Proceedings of the 6th International Workshop on the Web and Databases*.

Wang X, Zhang Y, Li X and Loguinov D (2004) On zone-balancing of peer-to-peer networks: analysis of random node join. In *Proceedings of the ACM SIGMETRICS'04 Conference*.

Watts D and Strogatz S (1998) Collective dynamics of small-world networks. *Nature*, 393.

Xu J and Croft W B (1999) Cluster-based language models for distributed retrieval. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Yaga, http://www.yaga.com.

Yang B and García-Molina H (2002) Improving search in peer-to-peer systems. In *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS)*.

Zhai C, Jansen P, Stoica E, Grot N and Evans D (1998) Threshold Calibration in CLARIT adaptive filtering. In *Proceedings of the 7th Text Retrieval Conference (TREC-7)*.

Zhai C, Jansen P and Evans D (2000) Exploration of a heuristic approach to threshold learning in adaptive filtering. In *Proceedings of 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

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.

Zhang H, Goel A and Govindan R (2002) Using the small-world model to improve Freenet performance. In *Proceedings of Infocom 2002*.

Zhang Y and Callan J (2001) Maximum likelihood estimation for filtering thresholds. In *Proceedings of 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Zhang Y, Xu W and Callan J (2002) Exact maximum likelihood estimation for word mixtures. In *Workshop on Text Learning of the 9th International Conference on Machine Learning (TextML' 2002)*.

Zhao B, Huang L, Stribling J, Rhea S, Joseph A and Kubiatowicz J (2004) Tapestry: A resilient global-scale overlay for service deployment. *IEEE Journal on Selected Areas in Communications*, 22(1): 41–53.

Zhu Y and Hu Y (2003) Efficient proximity-aware load balancing for structured peer-to-peer systems. In *Proceedings of the 3rd IEEE International Conference on Peer-to-Peer Computing (P2P2003)*.