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

Jie Lu

CMU-LTI-07-003

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
www.lti.cs.cmu.edu

Thesis Committee:

Jamie Callan (chair)
Jaime Carbonell
Christos Faloutsos
Norbert Fuhr, University of Duisburg-Essen, Germany

*Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
In Language and Information Technologies*

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

Jie Lu

Language Technologies Institute
School of Computer Science
Carnegie Mellon University

ABSTRACT

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

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

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

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOTATION

P	An information provider
C	An information consumer
H, h	A hub
N	A neighborhood
Q	A query
q	A query term
c	A query cluster
d	A document retrieved for a query
R	A set of documents retrieved for a query
A	The set of relevant documents for a query
r	The set of relevant documents retrieved for a query
TTL, ttl	Time-To-Live field of a query message that determines the maximum number of times the message can be relayed in the network
HD	Hub description
ND	Neighborhood description
F	Exponential decay factor for neighborhood description
G	Background language model used for smoothing
$tf(t, P)$	Aggregate term frequency of a term t in provider P's description
$tf(t, H)$	Aggregate term frequency of a term t in hub H's description
$tf(t, G)$	Aggregate term frequency of a term t in background language model G
$\mathrm{P}(Q)$	Probability of query Q
$\mathrm{P}(P \mid Q)$	Probability of predicting provider P given query Q
$\mathrm{P}(P)$	Prior probability of provider P
$\mathrm{P}(Q \mid P)$	Probability of provider P generating query Q
$\mathrm{P}(N \mid Q)$	Probability of predicting neighborhood N given query Q
$\mathrm{P}(N)$	Prior probability of neighborhood N
$\mathrm{P}(Q \mid N)$	Probability of neighborhood N generating query Q
μ	Smoothing parameter for Dirichlet smoothing
$E(j)$	E evaluation measure that combines precision and recall at rank j
$R(j)$	Recall (the percentage of relevant documents retrieved) at rank j
$P(j)$	Precision (the percentage of retrieved documents being relevant) at rank j
b	Parameter in the calculation of E measure to adjust the relative importance of recall and precision
S, s	Resource ranking score to estimate the likelihood of relevance of a resource
θ	Threshold of resource ranking score to determine how many top-ranked resources to select for further query routing
$U(\theta)$	A linear utility function of θ used in set-based threshold learning for resource selection of providers by each hub
$N_{rel}(\theta)$	Number of providers with relevant content whose resource ranking scores are above θ

$N_{nonrel}(\theta)$	Number of providers with no relevant content whose resource ranking scores are above threshold θ
$\mathrm{P}(s \wedge rel)$	Probability of an information provider having resource ranking score s and containing relevant content
$\mathrm{P}(s \wedge nonrel)$	Probability of an information provider having resource ranking score s and not containing relevant content
$\mathrm{P}(s \mid rel)$	Probability of an information provider containing relevant content to have resource ranking score s
$\mathrm{P}(s \mid nonrel)$	Probability of an information provider not containing relevant content to have resource ranking score s
$\mathrm{P}(rel)$	Prior probability of an information provider to have relevant content
KL	Kullback-Leibler divergence between resource descriptions
$T_{cluster}$	Threshold for query clustering
$T_{classification}$	Threshold for classifying a new query into existing query clusters
S_{min}	Minimum size of a query cluster to represent a topic of interest
D_{top}	Number of top-ranked documents used to represent a query
N_{max}	Maximum number of query clusters for a consumer
$\mathrm{P}_{\mathrm{emp}}(t \mid d)$	Empirically estimated probability that term t occurs in document d
$\mathrm{P}_{\mathrm{core}}(t \mid d)$	Underlying probability that term t is generated by the language model of document d
$\mathrm{P}(t \mid background)$	Probability that term t is generated by background (general English) model
λ	Smoothing parameter in linear interpolation smoothing
$\mathrm{P}(t_1, t_2 \mid d)$	Probability that terms t_1 and t_2 are generated by the language model of document d
$c(t \mid d)$	Number of times that term t occurs in document d
$c(t_1, t_2 \mid d)$	Number of times that terms t_1 and t_2 occur together in document d
ρ^*	Threshold to distinguish between similar and dissimilar hubs
M_{og}	Maximum number of outgoing long-range ("global") hub connections a hub can have
M_{ol}	Maximum number of outgoing local hub connections a hub can have
M_i	Maximum number of incoming hub connections a hub can have
GD	Graph distance (number of hops) between hubs
CD	Content distance (the inverse of content similarity) between hubs
β	Exponent to control the "distance scale" of long-range hub connections
$\hat{R}_n$	Percentage of the providers with relevant content accumulated via the n top-ranked hubs for a query

DEFINITIONS

Digital library A set of resources and associated technical capabilities for creating, searching and using information.

Text digital library A digital library consisting of a collection of documents primarily in text form.

Federated search Search using a single interface to support finding items that are scattered among a distributed set 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.

Known-item search Search aimed to find a single instance of a known object (e.g., a particular song by a particular artist).

Full-text search, full-text ranked retrieval Search that evaluates the relevance of documents to a query based on the full body of their text contents and presents the list of retrieved documents in relevance-based ranking.

Hub An abstract functional unit to represent a resource with more processing power and connection bandwidth to provide regional directory services in the network.

Information provider An abstract functional unit to represent a digital library that shares (text) documents in the network.

Information consumer An abstract functional unit to represent a user with information requests in the form of queries.

Leaf An information provider or information consumer typically with limited computing and network resources.

Peer An abstract notion of a participating entity in the network. It can be a single functional unit, or a combination of multiple functional units (e.g., both a provider and a consumer).

Logical connection A direct communication channel between a pair of peers at a protocol layer. Information is exchanged in the form of messages.

Neighbor The peer directly connected to a peer by a logical connection.

Degree The number of neighbors a peer has.

Hop A trip a message takes from one peer to another using the logical connection between them. The number of hops a message needs to travel between two peers is equal to the path length between them in the graph, with nodes representing peers and edges representing connections.

Neighborhood The set of peers that a message can reach by following the path(s) from one peer to its neighboring peer(s) and further traveling a number of hops.

Overlay A set of logical connections to organize peers in the network at a protocol layer.

Network architecture The design of a network which defines the functionality and responsibility of peers and their connections.

Network topology A particular instantiation of peer organization under a specific network architecture.

Graph distance The distance between peers in the graph used to model the network topology, which can be measured by the shortest path length from one peer to another.

Content distance The degree of dissimilarity between peers' contents.

Content-based locality A property of the network topology that has short path lengths between peers with similar contents.

Interest-based locality A property of the network topology that has a short path length between each peer and those whose contents are similar to its interests.

Small-world A property of the network topology that has a high likelihood of having a short path between any two peers without requiring dense connections.

Search mechanism The description of the process of federated search with a set of protocols for resource representation, resource location, and result integration.

Resource representation The process of discovering and representing the content covered by each resource (leaf, hub, or neighborhood).

Resource description The description of the content covered by a resource.

Language model A model that assigns a probability to a sequence of words by means of a probability distribution. For this dissertation, it refers to the description of a resource's content with a list of terms or phrases and their corresponding frequencies or probabilities.

Resource location The process of locating resources most appropriate for an information need based on resource representations.

Result selection The process of selecting resources most appropriate for an information need.

Result integration The process of integrating search results from multiple resources.

Result merging The process of merging multiple ranked retrieval results into a single, integrated ranked list.

Flooding The search mechanism that requires each peer to relay the query message it receives to all of its neighbors.

Characteristic search Search related to a user's persistent, long-term interests.

Uncharacteristic search Search aimed to satisfy a user's transient, ad-hoc information needs.

Precision The percentage of the retrieved documents being relevant.

Recall The percentage of the relevant documents retrieved.

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 each query. Many of them allow free search access to their contents via the Internet, but do not provide complete copies of their contents upon request. Many do not allow their contents to be crawled by Web search engines. In consequence, the contents provided by these digital libraries cannot be accessed by Web search engines such as Google and AltaVista that only conduct search on materials that can be copied to centralized repositories. How best to provide federated search[2] across such independent digital libraries is an unsolved problem often referred to as the "Hidden Web" problem.

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

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

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

[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 "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).

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 usually only has 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.

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 text digital libraries developed during the last decade to perform *full-text search*, in which the full body of each text document is searched. In addition, term frequency information is often used to rank documents by how well they satisfy each query, and the search result is presented with some form of relevance ranking ("*full-text ranked retrieval*"). We argue that most of the recent research on P2P networks offers little useful guidance for providing full-text search of current text digital libraries. 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 search techniques developed for full-text ranked retrieval assume a centralized control. Either all the documents are stored in a centralized repository, or information about all the documents is gathered at a centralized directory service. Traditional federated search ("distributed information retrieval") only requires the aggregate directory information about each collection instead of each individual document. However, a centralized directory is still assumed to store the directory information of all the collections. A central authority for search purpose may be undesired in P2P networks due to its susceptibility to become a performance bottleneck or the target of malicious attacks, or because it requires IT infrastructure and resources that are unavailable or impractical in the environment. Therefore, federated search in P2P networks requires new solutions to extend existing techniques designed for environments with a global control in order to address the problem of how multiple distributed resources work autonomously and collaboratively to accomplish the retrieval task.

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

1.3 Contributions

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

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

The network search model, which is described in Chapter 4, utilizes the network architecture and topology defined in the network overlay model in designing a full-text search mechanism that can offer a better combination of accuracy and efficiency than previous approaches to federated search of text digital libraries in P2P networks. We show in detail that the network search model is not a simple adaptation of existing solutions to full-text ranked retrieval. Its significance lies in our new development for each of the main components (resource representation, resource selection, and result merging) in consideration of the characteristics and requirements of federated search in P2P networks. Specifically, the concept of a neighborhood is defined, and exponentially decayed resource descriptions of neighborhoods are used for resource selection of hubs; unsupervised threshold learning methods are developed for resource selection of providers; user modeling with adaptive query clustering is proposed to improve resource selection performance for queries representing persistent and long-term user interests; and Kirsch's algorithm for result merging is extended to effectively merge multiple ranked lists without requiring global corpus statistics.

The network overlay model and the network search model are most useful if we can show that there are decentralized algorithms capable of evolving the topology of a P2P network into one with the search-enhancing properties described in the network overlay model and desired by the network search model. For this reason, we propose the network evolution model 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 without relying on a central control. In addition, it puts extra effort into avoiding high system overhead on topology evolution, and balancing load to make the network more scalable and robust.

This dissertation also includes extensive experimental results and analyses (Chapter 5 and Chapter 7) to provide strong empirical evidence for the effectiveness and practicality of the proposed models. The two P2P testbeds developed for evaluation are two of the largest so far consisting of real-content text digital libraries, showing our effort towards applying the newly developed approaches to real operational environments and verifying their effectiveness.

1.4 Outline

The rest of the dissertation is organized as follows. Chapter 2 provides background knowledge and discusses related work. Chapter 3 presents the network overlay model, including network architecture and network topology. Chapter 4 describes different components of the network search model, i.e., resource representation, resource selection, and result merging. Chapter 5 provides evaluation resources and experimental results to verify the effectiveness of the network search model. The description and evaluation of the network evolution model are included in Chapter 6 and Chapter 7 respectively. Chapter 8 concludes the dissertation by summarizing the research contributions, describing their significance, and discussing open problems and potential future research topics.

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. These logical connections are not necessarily associated with the underlying physical connections in the network. In this dissertation, by default "connections" refers to logical connections.

Three components of a P2P network are essential to federated search: *network architecture*, *search mechanism,* and *network topology*. A *network architecture* defines 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 integrated before being presented to the user ("*result integration*"*)*. 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. Figure 2.1 illustrates these P2P architectures.

Brokered P2P architecture

In a brokered P2P architecture, a group of peers (possibly located in the same setting) function as a single, centralized logical directory service ("*broker*"), and 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 these providers to download contents.

Perhaps the most famous brokered P2P architecture was the original Napster music file-sharing system.[3] Although brokered P2P architectures are vulnerable to a single point of failure, because they are easy to implement and control, they are still in active use (Yaga) (MusicNet) (Intel 2003).

Completely decentralized P2P architecture

All peers in a completely decentralized P2P 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.

[3] http://shumans.com/p2p-business-models.pdf

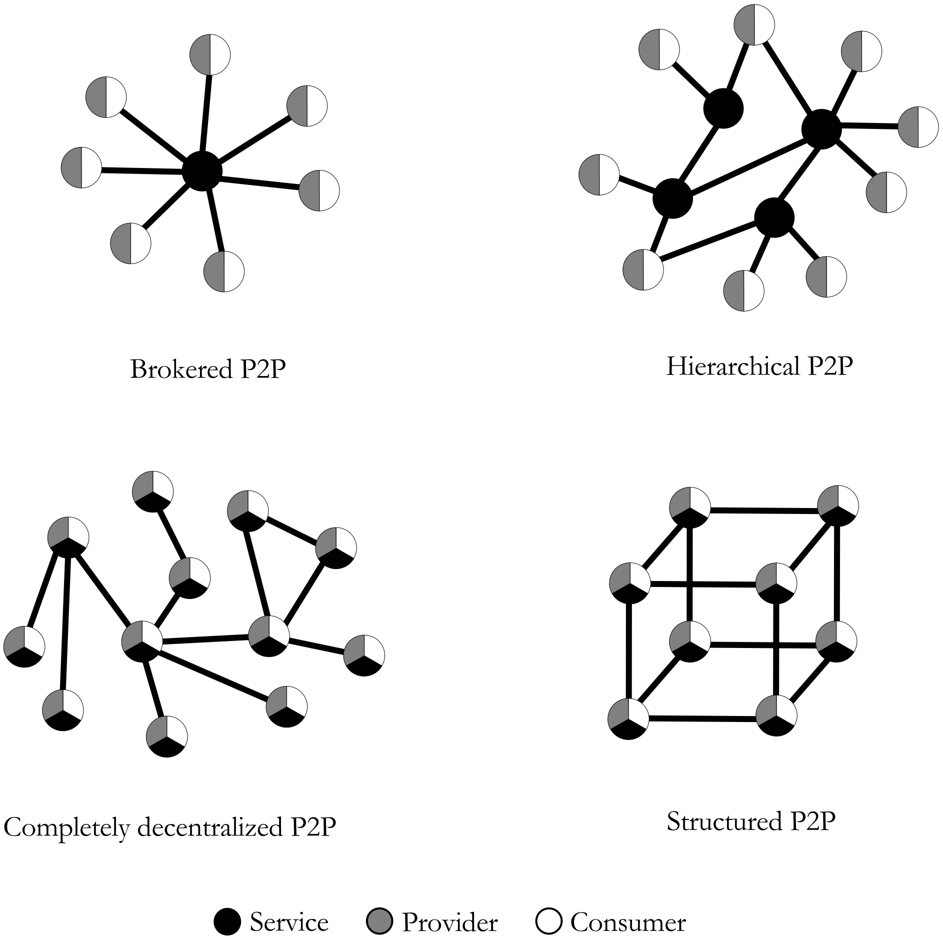


Figure 2.1 Illustration of different P2P architectures.

Completely decentralized P2P architectures are very flexible and robust, but have low scalability in terms of search performance. Gnutella v0.4 is the most representative example of a completely decentralized P2P architecture (Gnutella v0.4).

Hierarchical P2P architecture

Peers in a hierarchical P2P network 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).[4] Each

[4] 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. However, we are not aware of any hierarchical P2P systems that actually use more than two levels.

information provider independently stores its contents. Each hub provides directory services 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 or layers.

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. 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.[5]

Structured P2P architecture

In a structured P2P architecture, each peer is assigned a peer identifier drawn from certain key space. The connections between peers are determined by the locations of their peer identifiers in the key space.[6] For most P2P applications using structured P2P architectures, the contents provided in the network are partitioned based on either documents or terms. If document partition is used, each document (or document reference/pointer) is associated with a key identifier, taken from the same key space (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. If term partition is used, each term is associated with a key in the key space, and each peer stores the inverted lists of terms whose keys fall into the peer's part of the key space. Each term's inverted list records the list of documents (or peers) with contents containing the term. For certain applications that represent queries and contents using low-dimensional feature vectors, the contents can be partitioned based on the positions of their vectors in the hyperspace. Because systems having structured P2P architectures are generally realized through Distributed Hash Table abstractions, they are often referred to as DHT-based systems.

[5] BearShare, http://www.bearshare.com; Edutella, http://edutella.jxta.org; Gnucleus, http://www.gnucleus.com; Gnutella2, http://www.gnutella2.com; JXTA, http://www.jxta.org; KaZaA, http://www.kazaa.com; Limewire, http://www.limewire.com; Morpheus, http://www.morpheus.com; Shareaza, http://www.shareaza.com; Swapper.NET, http://www.revolutionarystuff.com/swapper/.

[6] The structured P2P architecture shown in Figure 2.1 is only to illustrate that the locations of peers and the relations between peers in a structured P2P network are determined by the positions of their peer identifiers in the partitioned key space. A structured P2P architecture does not necessarily have a cubical structure. For example, a ring structure with connections along chords is a popular alternative (Stoica et al. 2001).

Structured P2P architectures are scalable and efficient, but their strict content placement and peer connections may limit peer autonomy and the types of content or search services that the network can provide. For example, search in structured P2P networks is mostly restricted to known-item search or keyword-based search over a small, controlled vocabulary due to the difficulty to support full-text search, which is explained in more detail in Section 2.2.4. Widely used 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). 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).

2.1.2 Search Mechanism

For centralized search, the locations of the resources that generate search results for a query are fixed and known to users since documents are stored in a centralized repository. For federated search in a P2P network, where 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 ("*flooding*") is an admissible approach, a huge volume of network traffic will be generated for every query, making it extremely inefficient and not scalable (e.g., Gnutella v0.4). 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 (or references to contents) among peers so as to fix particular contents (or references to contents) to specific locations. Once relevant resources are located, an additional federated search problem is how to integrate search results returned by multiple resources into a single (ranked) list. In summary, three basic problems need to be addressed for federated search in a P2P network:

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

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

3. *Result integration*: Deciding how search results from multiple resources are integrated before being presented to the user that issued the information request.

We briefly describe common approaches for each of these problems.

Resource representation

For federated search in P2P networks, the basic unit of information content is typically 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 ("*free-text representation*"), iii) terms from a controlled vocabulary, possibly with a hierarchical structure based on an ontology ("*controlled-vocabulary 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 addition to individual document representations, federated search in P2P networks may also require the description of each resource's document collection as a whole. Similar representations for individual documents can be used to represent a collection of documents (e.g., the contents of an information provider) by ignoring document boundaries and treating the collection as one big document. Another approach is to associate terms extracted from document representations 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 (e.g., in music and movie P2P file-sharing applications), but they are generally not appropriate for text documents. Free-text and controlled-vocabulary 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 can be difficult or inaccurate for documents that are long and have heterogeneous contents. Therefore, they are mostly used for limited-domain contents. Full-text representations provide a much more comprehensive description for text documents than other representations, which makes them most suitable for P2P networks of text collections containing open-domain contents, but their sizes are significantly larger. Different representations are appropriate in different situations. There is no single representation that works best in all P2P environments.

In brokered P2P networks, name-based, free-text, and controlled-vocabulary representations are used more often than full-text representations due to their size advantage. Name-based representations are particularly popular because they require minimal computation, storage, and communication costs for resource location using the centralized directory service. All resource representations are widely used for resource location in completely decentralized and hierarchical P2P networks. However, because of the larger sizes and higher costs in dissemination and storage, full-text representations are more appropriate in hierarchical P2P networks, where resource location is mostly the responsibility of hubs that have more storage, processing power and connection bandwidth. Name-based, free-text, and controlled-vocabulary representations can be applied in a straightforward manner in structured P2P networks, but full-text representations usually require additional processing and manipulation (Tang et al. 2004) (Tang and Dwarkadas 2004).

Because disseminating resource representations for effective resource location usually involves non-trivial communication costs in P2P networks, reducing the sizes of resource representations is an important issue for federated search. 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). *Bloom filters* are widely used for this purpose. 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 filters provide 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. In addition, they cannot be used by common resource location algorithms that require term frequency information to estimate a resource's relevance to a query.

Resource location

Resource location based on name-based, free-text, or controlled-vocabulary representations typically uses simple keyword matching algorithms to decide which resources to select for each query. Full-text representations provide the potential for resource location to use more sophisticated algorithms based on term frequency information to estimate each resource's likelihood of providing relevant information.

In unstructured (brokered, completely decentralized, and hierarchical) P2P networks, which peer(s) is/are responsible for resource location depends on the chosen network architecture. In a network with a brokered P2P architecture, a single, logical directory service conducts 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 are conducted collectively by *query routing* among neighboring peers. The difference is that in a completely decentralized P2P network every peer is responsible for relaying the query message it receives to its neighbors (e.g., Gnutella v0.4), but in a hierarchical P2P network query routing is restricted to the directions of consumers to hubs, hubs to hubs, and hubs to providers (e.g., 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 (content-based or non content-based). Each message in the network has a Time-To-Live (TTL) field that determines the maximum number of times (*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. Sections 2.2.2 and 2.2.3 present different approaches to resource location in completely decentralized P2P networks and hierarchical P2P networks respectively.

In structured P2P networks, queries are converted to key identifiers, and the task of resource location is to locate the peers responsible for them (Ratnasamy et al. 2002). Each peer stores resource representations associated with a certain range of keys, and is responsible for directing queries of these keys to the corresponding peers. When a peer receives a queried 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 key identifier. Different DHTs explore different key spaces, which lead to different measures for distance between identifiers, and therefore affect how to choose which neighbor to relay a query during search. Section 2.2.4 provides more details on the mechanisms used by various structured P2P systems for resource location.

Result integration

Although presenting search results in some form of relevance-based ranking is common practice for search in a centralized repository (e.g., Google), the search results provided by most P2P networks do not have relevance-based rankings. Document retrieval at an information provider in most P2P networks uses Boolean keyword matching, which can only return a list of matched documents 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 provides a relevance-based ranking of documents by using term frequency information from full-text representations and sophisticated term weighting schemes to estimate how well they each satisfy the query. However, if multiple ranked search results are returned by providers, result integration becomes important and challenging because different providers use different corpus statistics (which are most likely skewed) to calculate relevance-based ranking scores, making them globally incomparable (Callan 2000). More complex result merging techniques are required to merge them into a single, integrated relevance-based ranking, which are not provided by current P2P networks.

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 relative positions of 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*. It is worth noting that these properties are not mutually exclusive, i.e., it is possible that a single network exhibits all three properties.

Interest-based locality

A network topology with interest-based locality has a short path length between each peer and those peers whose contents are similar to its interests. 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 each peer's queries expressing its interests do not need to travel far in order to locate relevant contents.

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). There has also been some work on utilizing interest-based locality in P2P networks to improve search efficiency (Sripanidkulchai et al. 2003) (Shao and Wang 2005).

Content-based locality

A network topology is said to exhibit content-based locality if peers with similar contents are located near to one another (i.e., connected by short path lengths). 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.

Content-based locality has been used in completely decentralized and hierarchical P2P networks to improve search efficiency by locating relevant content-based clusters and reducing the search load on peers with unrelated contents (Crespo and García-Molina 2002a) (Schlosser et al. 2002) (Löser et al. 2003). Content-based locality can also improve the robustness of federated search because resource representations generated from network regions with content-based locality are more resilient to changes in content due to peer arrivals and departures.

Small-world

The small-world phenomenon refers to the property that any two individuals in the network are likely to be connected through a short sequence of intermediaries (Kleinberg 2000). 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 v0.6 (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 is desirable because it not only guarantees the existence of short paths (graph distance) 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 2000). 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, from a theoretical point of view, 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 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 was a representative brokered P2P network that used a name-based representation and Boolean keyword matching for music files.[7] 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 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 representations 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 integrations.

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 downloads files by directly contacting providers that responded.

A completely decentralized P2P network is simple to build and maintain. It easily reacts to the high dynamism 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 of 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 that motivate distributed P2P systems.

Recent research provides a variety of solutions to increase search performance in completely decentralized P2P networks by avoiding flooding. 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

[7] The original Napster refers to the Napster peer-to-peer music-sharing service that was started in 1999 and shut down in 2001 (http://shumans.com/p2p-business-models.pdf).

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). Some approaches explore query-dependent criteria for resource location. In (Kalogeraki et al. 2002), each peer uses the past query responses from its neighbors to build run-time profiles for them (essentially a free-text 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 query-dependent approach is for each peer to store content information about other peers, based on which it selects peers most likely to have contents relevant to the query. Query Routing Protocol uses a Bloom filter to summarize document keywords and disseminates it in the network (Rohrs 2001). The *routing index* of a peer records the numbers of documents for a set of topics that may be found along paths that begin at each neighbor (Crespo and García-Molina 2002b). In PlanetP, each peer collects compact summaries about other peers' inverted indices (Cuenca-Acuna and Nguyen 2002). Similarly, *local indices* require each peer to maintain a local index of the contents of other peers that exist within a predetermined range (Yang and García-Molina 2002).

Approaches that belong to the third category use certain properties in network topologies for more efficient and effective resource location. In Crespo and García-Molina's work on Semantic Overlay Network (SON), each query is routed to the appropriate SON formed by peers with similar contents, 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 improves search performance in a completely decentralized P2P network by utilizing concept clusters formed by peers based on a global ontology (Schlosser et al. 2002). (Sripanidkulchai et al. 2003) propose 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. To find relevant content, a peer first queries peers that answered its previous queries and turns to a default search mechanism such as flooding only if these peers cannot answer the query. There is also research that explores small-world properties under completely decentralized P2P architectures. For instance, (Merugu et al. 2004) show that search in the network with 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) also construct a small-world topology in a completely decentralized P2P network to improve search performance.

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. For instance, Query Routing Protocol uses a free-text representation with a Bloom filter (Rohrs 2001). Routing indices (Crespo and García-Molina 2002) and the system described in (Kalogeraki et al. 2002) with "intelligent search" mechanism use free-text representations. 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), Hypercube P2P (Schlosser et al. 2002), and the system of (Sakaryan and Unger 2003) adopt controlled-vocabulary representations.

Although previous work demonstrates that search efficiency in completely decentralized P2P networks can be greatly improved by using more sophisticated search mechanisms to avoid flooding, the improvement is achieved at the cost of complicating the functionality and responsibility of each peer without considering the differences between different peers' available resources. Peers with limited processing power and connection bandwidth easily become bottlenecks and may cripple the whole network, especially when large-sized resource representations such as full-text representations are used, making completely decentralized P2P networks inappropriate for full-text ranked retrieval.

2.2.3 Search Mechanisms in Hierarchical P2P Networks

Hierarchical P2P architectures provide another approach to alleviate search overhead caused by 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 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.

A hierarchical P2P network can also take advantage of the search-enhancing properties (described in Section 2.1.3) of its network topology to further improve federated search performance. For example, (Löser et al. 2003) suggest that in a schema-based hierarchical P2P network, search efficiency can be enhanced and flooding the network with messages can be reduced if the network topology exhibits content-based locality by forming semantic overlay clusters. 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. A history-based search and topology adaptation mechanism is proposed in (Shao and Wang 2005) to improve the efficiency of search in a hierarchical P2P network by utilizing interest-based locality. To the best of our knowledge, no work has been done to explicitly use small-world properties to improve search performance in hierarchical P2P networks.

Hierarchical P2P architectures easily support sophisticated search techniques that are not constrained to representations using controlled or small vocabularies. 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. However, compared with structured P2P networks, more peer autonomy and less restrictive content placement in hierarchical P2P networks cause higher system overhead on dynamic self-organization, making hierarchical P2P networks more flexible but less cost-efficient in terms of computation and communication.

2.2.4 Search Mechanisms in Structured P2P Networks

Various structured P2P systems (DHTs) can be distinguished by the mechanisms they use to generate key identifiers and peer identifiers, to determine peer connections, and to perform distributed hash table lookup for resource location. The most well-known structured P2P systems include CAN, Chord, Pastry, and Tapestry. CAN (Ratnasamy et al. 2001) embeds its key space in a torus with d dimensions. Each peer is responsible for a hypercubical region of this key space, and its $O(d)$ neighbors are peers responsible for the contiguous hypercubes. $O(dn^{1/d})$ hops are required for query routing. Chord (Stoica et al. 2001) places peers on a one-dimensional circle. Each peer is responsible for the keys whose numerical values are most closely followed by its peer identifier. The $O(log\ n)$ neighbors of each peer include its immediate successors along the circle, and peers spaced exponentially around the key space. The routing path lengths are $O(log\ n)$ hops. Similar to Chord, Pastry (Rowstron and Druschel 2001) uses a one-dimensional circular key space, and peers are responsible for keys that are closest numerically. Each peer has $O(log\ n)$ neighbors (half larger, half smaller in peer identifiers), and routes a queried key to its neighbor with the longest matching prefix, requiring $O(log\ n)$ hops. Tapestry (Zhao et al. 2004) uses SHA-1 to produce a 160-bit key space represented by a 40-digit hex key. Peer identifiers are roughly evenly distributed in the key space. Each peer's neighbors has multiple levels where each level contains peers whose identifiers match up to a certain digit position in the key space. A queried key is progressively routed by incremental suffix routing. Similar to Chord and Pastry, each peer has $O(log\ n)$ neighbors and the path lengths for routing are $O(log\ n)$ hops.

A structured P2P network with document partition can be used for known-item search. In this case, the queried key is the key identifier of the document to be located, and the targeted peer is the one that contains the document or a pointer to the document. A structured P2P network with term partition can support keyword-based search. For a query with multiple terms, the key associated with each query term is used to locate the peer that stores the term's inverted list which lists documents or peers containing the term. Typically simple intersection is used to combine multiple inverted lists, ignoring any correlation between the terms in the query. Because the best document or peers for the entire query may not be among the top candidates for any of the individual query terms, this approach often leads to unsatisfactory search accuracy. In addition, the communication cost for an intersection grows proportionally with the number of query terms and the length of the inverted lists. Several approaches have been developed to remedy the problems caused by "factorization" by requiring each peer to store more information in addition to the inverted lists of the terms it is responsible for. For example, the hybrid global-local indexing approach proposed by (Tang and Dwarkadas 2004) requires each peer to store the term list of each document that occurs in the inverted list of a term it is responsible for, so that a multi-term query can be processed locally without using intersection to combine the inverted lists in the global index distributed in the network. The method described in (Bender et al. 2006) disseminates periodically mined term correlation statistics and requires each peer to store additional information for terms that are strongly correlated with the terms it is originally responsible for.

For some P2P applications such as music file-sharing, queries and contents can be described by a small number of attribute-value pairs using low-dimensional feature vectors for representations. There has been some work on using a KD-tree to partition such low-dimensional content spaces, and to distribute the tree structure in the P2P network with a DHT architecture (Gao 2004). Range and simple similarity queries can be supported by locating the right "cells" in the partitioned hyperspace for the requested feature vectors. A small number of peers serve as rendezvous points to avoid message flooding. Load balancing is used to avoid overloading these rendezvous points.

Known-item search in document-partitioned structured P2P networks uses name-based resource representations. Keyword-based search in item-partitioned structured P2P networks can be based on free-text, controlled-vocabulary, or full-text representations. However, because it adopts global indexing in distributed environments, the high communication cost of index updating, especially for full-text representations, may limit its use. In addition, the skewed corpus statistics each peer has as a result of term partition lead to globally incomparable ranking scores, making result integration of relevance-based rankings ineffective if not impossible. Structured P2P networks with tree-based partitioning (Gao 2004) work well for search over low-dimensional feature representations. However, the costs of searching and dynamically organizing network structures in these networks are too high for full-text search because full-text search requires a much larger number of dimensions for resource representation.

2.3 Related Work on Network Topologies in P2P Networks

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

2.3.1 Network Topologies with Interest-Based Locality

Work by (Ramanathan et al. 2002) moves a peer closer to those that more frequently responded successfully to its past information requests in order to improve resource location performance for future queries. 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 none of these approaches explicitly distinguish between the different interests of a user (e.g., sports versus music), their effectiveness will be negatively affected when the user has multiple distinct interests or when the user has different short-term and long-term interests.

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) propose to use a global classification hierarchy to cluster peers into one or more Semantic Overlay Networks (SONs), and require peers to 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 are usually not the case with text digital libraries containing heterogeneous, open-domain contents. In addition, this approach assumes that the classification hierarchy or ontology will distribute items fairly evenly, which means that it must be well-matched to the contents available in the network, a condition difficult to satisfy for open-domain contents in dynamic networks. 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 requires greater computation and communication, which may become a burden for peers with limited resources. 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 2000). In his d-dimensional lattice model, the probability of establishing 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[8] 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.

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

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, 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 a large clustering coefficient, and the distribution of connection lengths has a stretched exponential tail. Inspired by surfers' behavior on the Web to update the outgoing links from their home pages, (Clauset and Christopher 2004) describe 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.

There are also network evolution algorithms that construct a small-world topology without relying on global knowledge or rigorous network models. A network evolution algorithm that constructs a small-world topology adaptively using only local information at each peer is proposed 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. In the topology evolution algorithm described in (Merugu et al. 2004), each peer adaptively moves to the appropriate location in the topology by selecting "better" neighbors (several closest peers and a few random peers) repeatedly according to its local view of the network. Peers obtain their local knowledge of the network by measuring their latency distances to peers that are located within two hops.

Among the algorithms described above, (Watts and Strogatz 1998) and (Kleinberg 2000) require global knowledge of the content distance from each peer to any other peer, which may be difficult to acquire in real P2P environments. Although the rewiring processes proposed in (Manna and Kabakcioglu 2003) and (Clauset and Christopher 2004) can converge to a small-world topology, the convergence may be quite slow for large-scale P2P networks. (Sakaryan and Unger 2003) and (Merugu et al. 2004) 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. Topology evolution algorithms based on local information such as (Sakaryan and Unger 2003) and (Merugu et al. 2004) work well in the face of dynamic peer arrivals and departures. Direct applications of the other algorithms mentioned in this section to dynamic environments are unlikely to be successful due to their requirements of global information and/or assumptions about rigorous network models.

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 (Callan 2000):

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 three problems have been developed in distributed information retrieval. We briefly review some of them below.

2.4.1 Resource Representation

For full-text search, 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. Resource representation deals with the problems of acquiring information about terms, frequencies, and collection sizes from digital libraries.

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 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 (fixed) small number of queries and documents in distributed information retrieval environments (Callan and Connell 2001). Recently several variations to the basic query-based sampling algorithm have been proposed to improve the quality of sampling by adaptively adjusting the numbers of sampling queries and documents from digital libraries based on the estimated quality of existing samples (Caverlee et al. 2006).

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). Two new methods of estimating collection sizes in uncooperative distributed environments, namely "Multiple Capture-Recapture" and "Capture-History", have been proposed recently (Shokouhi et al. 2006). Evaluation results across several collections demonstrate that they provide a closer estimate of collection sizes than previous methods, and require less information than the Sample-Resample technique.

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 ranks resources by their likelihood of returning relevant documents, and selects the top-ranked resources to process the information request.

Resource selection algorithms such as CORI (Callan et al. 1995) (Callan 2000), CVV (Yuwono and Lee 1997), and Kullback-Leibler (K-L) divergence-based (Xu and Croft 1999) algorithms use techniques adapted from document retrieval for resource ranking. They treat resource representations as big documents without explicitly considering individual documents within each resource. CORI uses a Bayesian inference network model with an adapted Okapi term frequency normalization formula to rank available resources. CVV assigns higher weights to terms that better distinguish different resources and ranks resources by the sum of the weighted document frequencies of query terms. The K-L divergence-based resource selection algorithm ranks resources by the K-L divergence between query language model and the unigram language model of each resource.

The hierarchical database sampling and selection algorithm (Ipeirotis and Gravano 2002) and the shrinkage-based resource selection algorithm (Ipeirotis and Gravano 2004) use base algorithms such as CORI to conduct resource selection but provide a better way to smooth the word distribution in resource representations.

Resource selection algorithms that consider individual documents within each resource include vGlOSS (Gravano and García-Molina 1995) (Gravano et al. 1999), DTF (the decision-theoretic framework for resource selection) (Nottelmann and Fuhr 2003), ReDDE (Si and Callan 2003a), and the unified utility maximization framework for resource selection (Si and Callan 2004b). These algorithms rank resources by directly estimating the amount of relevant documents from each resource for a given query. Additional statistics or training data are required for such estimation. For example, vGlOSS needs from each resource information about the sum of each term's weights in its documents. One variant of DTF, named DTF-sample, uses sample documents to estimate how relevant documents are distributed among the available resources. Another variant, DTF-normal, models the distribution of document scores from a resource with normal distribution and maps document scores to probability of relevance using a function learned with user relevance feedback. ReDDE and the unified utility maximization framework for resource selection rely on sample documents in the centralized sample database obtained using query-based sampling to estimate the relevance of each resource.

Resource selection algorithms such as query clustering/RDD (Voorhees et al. 1995) and lightweight probes (Hawking and Thistlewaite 1999) use training queries to obtain information from resources for ranking. Given a query, the query clustering/RDD algorithm ranks resources based on the distribution of human-judged relevant documents for similar training queries. The lightweight probes method broadcasts two-word subsets of user queries to resources to obtain query term statistics, which are used for ranking these resources.

Applying the above resource selection algorithms in brokered P2P networks is straightforward. But for other P2P networks that rely on multiple, regional directory services to conduct resource selection, each directory service not only is responsible for selecting among information providers in its own region, but also needs to participate in the work of locating the right directory services across the network. Most existing resource selection algorithms are developed for using a single, centralized directory service to select among multiple databases, which are not directly applicable to selection of directory services. The resource selection algorithm of hGlOSS (Gravano and García-Molina 1995) uses a higher-level server to select multiple lower-level directory services based on their summaries. However, global knowledge is required to organize directory services into a tree-style hierarchy, which may not be practical for full-text federated search in large-scale, dynamic P2P networks. Therefore, new development is needed for resource selection according to the unique characteristics of P2P networks.

Which resource selection algorithms to choose as the basis of new development largely depends on the trade-off between performance and cost. Compared with resource selection algorithms that either rely on human relevance judgments (Voorhees et al. 1995) (Hawking and Thistlewaite 1999) or require much more information from resources in order to obtain better relevance estimates (Ipeirotis and Gravano 2002) (Ipeirotis and Gravano 2004) (Nottelmann and Fuhr 2003) (Si and Callan 2003a) (Si and Callan 2004b), simple resource selection algorithms such as the CORI (Callan et al. 1995) (Callan 2000) and K-L divergence-based (Xu and Croft 1999) algorithms have an advantage for federated search in P2P networks because they require simpler resource representations and less communication cost. In addition, previous studies show that these two algorithms are quite robust and effective in different experiment environments (French et al. 1999) (Xu and Croft 1999) (Craswell et al. 2000). However, if the higher communication cost incurred by acquiring more information from resources can be tolerated in exchange for higher search accuracy, then resource selection algorithms such as ReDDE (Si and Callan 2003a) can be extended to federated search in P2P networks.

2.4.3 Result Merging

Several result merging algorithms have been proposed in distributed information retrieval (Kirsch 1997) (Callan 2000) (Le Calv and Savoy 2000) (Si and Callan 2003b). One approach is based on normalizing resource-specific document scores into resource-independent document scores. The CORI merging algorithm uses a linear combination of digital library scores and document scores to normalize the scores of the documents from different digital libraries (Callan 2000). 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. The relative weights in the linear combination are set empirically based on query-independent heuristics. The work of (Le Calv and Savoy 2000) uses logistic regression to learn resource-specific query-independent merging models to normalize document scores, but relevance judgments are required for training. 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).

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. Kirsch's algorithm (Kirsch 1997) allows very accurate normalized document scores to be determined without the high communication cost of downloading by requiring each resource to provide summary statistics for each of the retrieved documents, but global corpus statistics are required in score recalculation. (Viles and French 1995) show in their work that partial dissemination of global corpus statistics can still enable effective result merging. (Craswell et al. 1999) use a reference statistics database containing all the relevant statistics for some set of documents to substitute corpus statistics and demonstrate its effectiveness.

In P2P environments where digital libraries often vary widely in their sizes and contents, the CORI merging algorithm (Callan 2000) is not likely to work well due to its resource-independent and query-independent linear weights for score normalization. The human-judged training data required by learning logistic merging models (Le Calv and Savoy 2000) may not be easily available, and the high communication cost associated with constructing the centralized sample database to generate the required training data makes the Semi-Supervised Learning result merging algorithm (Si and Callan 2003b) undesirable in P2P networks that are cautious about bandwidth usage. Because disseminating global corpus statistics also involves high communication cost in distributed environments, the benefit of using Kirsch's algorithm for result merging (Kirsch 1997) may not offset its cost. Result merging in P2P networks desires an algorithm that can work effectively with minimum additional training data and communication cost, for which none of existing result merging algorithms directly qualify.

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. Basic network architectures include brokered, completely decentralized, hierarchical, and structured P2P architectures. A brokered P2P architecture uses a single, centralized directory service, which is simple, efficient, and easy to control, but less robust and not appropriate for distributed environments without the support of a central IT infrastructure. A completely decentralized P2P architecture requires each peer to provide directory services, which is robust and easy to deploy in distributed environments, but less efficient. A hierarchical P2P architecture relies on multiple collaborative regional directory services, which combines the strengths of brokered and completely decentralized P2P architectures, but requires extra system overhead for collaboration among directory services. A structured P2P architecture uses a distributed hash table for directory services, which is efficient, but restrictive and less flexible.

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

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

Following the overview of the basic components of federated search, previously developed approaches to search mechanisms and network topologies are reviewed, and their strengths and weaknesses for search in various P2P environments are pointed out. Despite the development of several main ingredients, no existing work has provided a complete recipe for full-text ranked retrieval in P2P networks. The study of these approaches inspired our development of network architecture (network overlay model), search mechanism (network search model), and network topology (network overlay model and network evolution model) for full-text federated search.

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

Chapter 3

NETWORK OVERLAY MODEL

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

In this chapter, we present the network overlay model, based on which our network search model and network evolution model are developed. Within the network overlay model, the network architecture is described first, followed by the network topology.

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. It is worth noting that the general framework described in this dissertation 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) (Lu and Callan 2006a).

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

A *provider* is a digital library that shares text documents in the network. It provides a full-text search service by running a document retrieval algorithm over a local document collection and returning a list of matched documents in response to a 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. Each provider can choose its own document retrieval algorithm, which means that it is not necessary for all providers to use the same algorithm for document retrieval. In addition to responding to incoming queries, a provider also provides an accurate description of its content to its neighboring hubs upon request.

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

3.1.2 Connections

As described in Chapter 2, connections in an information-sharing P2P network are data channels established at the logical protocol layer through which information is exchanged in the form of messages. In previous P2P architectures, each connection is a data channel between a pair of peers. For a peer with multiple functional units (e.g., both as a consumer and as a provider), each of its connections to other peers is shared by these units. Because different functional units use the same connection for different purposes, it is difficult to optimize the utilities of all functional units of the same peer using a single set of connections since different functional units may desire different sets of connections. For example, it is quite likely that a peer's utility as a provider is optimized when it connects to one set of peers, but its utility as a consumer is optimized by connecting to a different

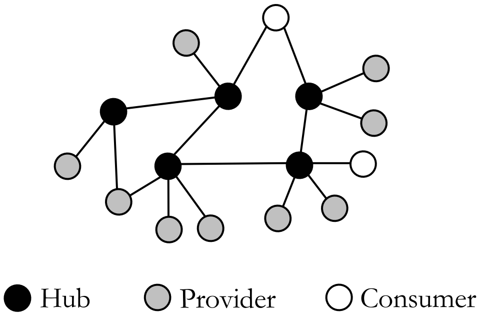


Figure 3.1 Illustration of the network architecture.

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.[9] By doing so, connections to a peer with multiple functional units for different purposes can be established and adjusted independently so that it is easier to achieve an optimal setting for the utilities of all functional units simultaneously.

In our hierarchical P2P network architecture, leaves (providers and consumers) only connect to hubs. Hubs connect with leaves and other hubs. 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 the defined network architecture. Note that a leaf (more precisely, each functional unit of a leaf such as a provider or a consumer) can connect to multiple hubs, as illustrated in Figure 3.1.

3.2 Network Topology

For a network that adopts the hierarchical P2P architecture described in Section 3.1, its topology has the components of hub-provider topology, hub-hub topology, and hub-consumer topology, each of which serves different purposes and desires different search-enhancing properties. 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. How the topology of a network can be evolved to one with these properties is the topic of our network evolution model (Chapter 6).

[9] 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

The hub-provider topology exhibits content-based locality by connecting providers with similar contents to the same hub to form a content-based cluster. Each content-based cluster defines a *content area* in the network. With content-based locality, 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. In contrast, a randomly generated hub-provider topology produces an arbitrary content distribution in the network. In this case, queries must be routed to hubs all over the network in order to locate enough relevant contents when the objective of search is not locating a single relevant document, but retrieving a sufficient number of documents relevant to the information request.

Having content-based locality in the hub-provider topology can also increase the robustness of full-text federated search. By covering a cohesive content area, the representation of the contents each hub serves remains relatively stable even if the members of its connecting information providers may change over time due to the dynamic nature of P2P networks. By comparison, in a randomly generated hub-provider topology, the representation of the content area covered by a hub may change dramatically as a result of the arrivals and departures of information providers. Because the performance of full-text federated search largely depends on the effectiveness of resource location while resource location relies on the quality of resource representation, content-based locality reduces the susceptibility of full-text federated search to dynamic content change in P2P networks.[10]

Existing work on constructing content-based locality in P2P networks either uses a global classification hierarchy or ontology to partition the content space into explicitly defined content areas (Crespo and García-Molina 2002a) (Schlosser et al. 2002) (Löser et al. 2003), or relies on individual peers to discover implicitly formed content areas without distinguishing between peers' differences in their available resources for such discovery (Khambatti et al. 2002) (Asvanund 2004). The use of the former approach is mostly limited to structured and limited-domain contents, while the latter approach puts a high burden on peers with limited resources. By using dynamically constructed content-based clusters to define content areas implicitly and requiring hubs with more resources to manage these clusters, content-based locality in our network overlay model can be applied more efficiently and effectively to P2P networks with heterogeneous, open-domain contents.

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

3.2.2 Hub-Hub Topology with Content-Based Small-World Properties

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

As mentioned in Section 2.1.2, federated search in a hierarchical P2P network relies on message-passing to first locate hubs that cover relevant contents before these hubs further direct messages to the connecting providers. By keeping hubs with similar content areas near to one another, relatively homogeneous *content regions* (a collection of similar content areas) can be formed at the hub level so that query routing can be more effective once a query arrives at the right content region. The existence of hub-hub connections that link dissimilar content regions of the network assures that a query can be routed to the targeted content region efficiently irrespective of where it starts. Therefore, efficient and effective full-text federated search in a hierarchical P2P network requires the properties of both locational proximity of similar content areas to form content regions and short global separation of dissimilar content regions, which are exactly small-world properties with a definition of peer distance based on content similarity.

Most efforts on studying and constructing networks with small-world properties use global knowledge or assume rigorous topology models (Watts and Strogatz 1998) (Kleinberg 2000) (Manna and Kabakcioglu 2003) (Clauset and Christopher 2004), which are not suitable for distributed, dynamic environments. The few approaches that work in dynamic P2P networks with local information randomly choose long-range connections without considering the distribution of similarities for navigability (Sakaryan and Unger 2003) (Merugu et al. 2004). To the best of our knowledge, no previous work has focused on content-based small-world properties with good navigability in dynamic P2P networks, making our attempt one of the first.

3.2.3 Hub-Consumer Topology with Interest-Based Locality

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

Several methods that are similar in nature have been proposed to use interest-based locality in improving search performance in P2P networks (Ramanathan et al. 2002) (Sripanidkulchai et al. 2003) (Shao and Wang 2005). However, they ignore the fact that a consumer can perform two different types of search activities: search related to the user's persistent, long-term interests ("*characteristic search*") and search aimed to satisfy transient, ad-hoc information needs ("*uncharacteristic search*"). Interest-based locality can improve search performance for queries of characteristic search ("*characteristic queries*") which are conceptually related to each other, since it is likely that hubs covering content areas relevant to past characteristic queries also cover relevant contents for future queries expressing similar interests, especially when the network topology exhibits content-based locality. But interest-based locality does not provide an effective solution to queries of uncharacteristic search ("*uncharacteristic queries*") since their relevant contents are unlikely to be covered by hubs with content areas related to characteristic queries. As a result, for uncharacteristic queries, the consumer must resort to a more extensive search using a larger search radius and rely on certain properties of the hub-provider and hub-hub topologies for effective and efficient query routing, trading efficiency for accuracy. Recognizing when interest-based locality enables effective federated search and when it doesn't distinguishes our work from previous research. It also helps us establish hub-consumer connections with better interest-based locality by filtering the noise introduced by uncharacteristic search.

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 document ranking. 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., characteristic search and uncharacteristic search), and their importance in achieving an optimal setting for the utilities of all functional units simultaneously.

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

Although the network overlay model provides a general framework which allows various approaches for different components of federated search to be plugged in to suit the conditions and requirements of particular P2P networks (e.g., cooperative vs. uncooperative), in the later chapters we primarily focus on federated search in cooperative environments and seek solutions that can achieve desired effectiveness without high computational complexity and communication cost. Specifically, the next chapter describes in detail how the functionalities of various functional units are implemented, and how the task of full-text federated search is accomplished collectively. Chapter 6 presents our newly developed algorithms to dynamically evolve the topology of a P2P network into one with the defined search-enhancing properties to further facilitate high performance federated search.

Chapter 4

NETWORK SEARCH MODEL

Search is essential to information-sharing peer-to-peer networks. Federated search in hierarchical P2P networks provides robustness and scalability by relying on local coordination, but it is typically less efficient than using a central authority to provide global coordination among peers. As a result, compared with search using a centralized index or 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 architecture 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 P2P networks. In the next chapter we discuss experimental evaluation.

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 neighboring peers 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) (Cuenca-Acuna and Nguyen 2002) (Crespo and García-Molina 2002b), but few previous methods of using directed breadth-first search in P2P networks have been targeted at full-text ranked retrieval, which motivated us to develop a type of directed breadth-first search mechanism for this task.

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 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 is further improved by extending 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, we also extend existing result merging techniques 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 required 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 other 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 different neighborhoods of the network is developed (Section 4.2.3). In consideration of peer autonomy and lack of central control in peer-to-peer networks, resource selection (Section 4.3) and result merging (Section 4.5) are designed to work without relying on global corpus statistics or centralized coordination.

One problem that cannot be solved by extending traditional distributed information retrieval techniques is initial resource selection by consumers. Consumers conduct initial resource selection to choose hubs that serve as entry points to the network. These hubs use their directory services to further propagate the query in the network. Because most current P2P networks provide very limited information to consumers about the available contents and their placement in the network, the resource selection conducted by each consumer to initiate search is typically no more than a random selection from a list of known hubs. Since there is no guarantee that the (arbitrarily) selected hub(s) can directly locate relevant providers, a relatively large search radius is usually required to reach the hubs that cover relevant contents. To eliminate the randomness and improve the quality of initial hub selection at each information consumer so that the amount of hub-hub query routing can be reduced, we propose to model the user's persistent, long-term interests at each information consumer based on past queries, and use the model to conduct initial hub selection for new queries according to the hubs' resource location effectiveness for old queries with similar interests ("*interest-based hub selection*"). Interest-based hub selection can be applied to characteristic search (Section 3.2.3) of persistent information needs to effectively reduce the amount of query routing and improve search efficiency without sacrificing accuracy. However, uncharacteristic search of transient, ad-hoc information needs requires a different search strategy because interest-based hub selection is unlikely to be effective for information requests not related to search history. Therefore, our approach has been developed to enable each consumer to distinguish between different types of queries in order to apply different search strategies to optimize the overall search performance. Details are given in Section 4.4.

Our search mechanism does not adopt the decentralized search algorithm proposed by Kleinberg for efficient navigation in a P2P network (Kleinberg 2000) because the assumptions and requirements of the algorithm cannot be satisfied in full-text federated search. Specifically, 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 to quickly find a path that leads from source to target. However,

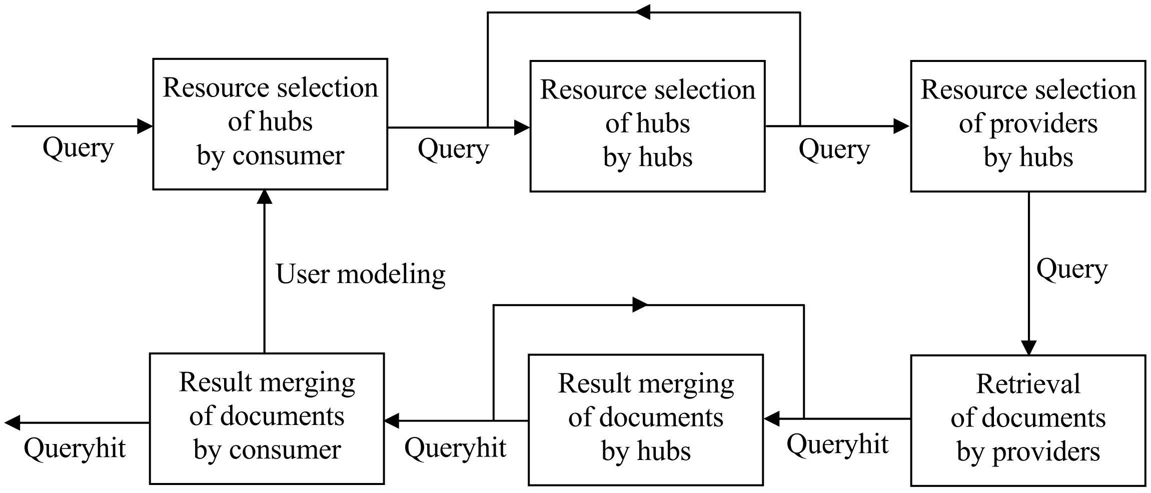

Query
Resource selection of hubs by consumer
Query
Resource selection of hubs by hubs
Query
Resource selection of providers by hubs
User modeling
Query
Queryhit
Result merging of documents by consumer
Queryhit
Result merging of documents by hubs
Queryhit
Retrieval of documents by providers

Figure 4.1 Illustration of the network search model.

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. As a result, the graph distance from the source or any other peer to the target, which is required by the algorithm for navigation, cannot be determined in advance. In contrast, 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, 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 (Time-To-Live) value to one or more hubs selected using interest-based selection (for characteristic search) or random selection (for uncharacteristic search). A hub that receives the query message uses its resource selection algorithm to rank and select one or more neighboring providers as well as hubs and routes the query to them with a decreased TTL until the message's TTL reaches zero. A provider that receives the query message uses its full-text document retrieval algorithm to generate a relevance-based ranking of its documents and responds with a queryhit message that contains a list of the top-ranked documents. A hub is responsible for collecting the queryhit messages generated by multiple neighboring providers, using its result merging algorithm to merge multiple ranked lists of documents from these providers into a single, integrated ranked list, and returning it to the consumer. Finally, a consumer needs to merge results returned by multiple hubs. The search results for past queries are used by each consumer to construct a user model to improve the performance of interest-based selection for characteristic search. Figure 4.1 illustrates the interactions between the various components of the network search model.

Table 4.1 An example of a provider’s full-text resource description.

<table>
<tr><td colspan="4">Provider URL = www.gov.state.ak.us/ltgov/elections</td></tr>
<tr><td colspan="2">Total number of terms = 1,397,045</td><td colspan="2">Total number of documents = 1,504</td></tr>
<tr><th>term</th><th>frequency</th><th>term</th><th>frequency</th></tr>
<tr><td>vote</td><td>18,790</td><td>ballot</td><td>3,659</td></tr>
<tr><td>district</td><td>12,431</td><td>official</td><td>2,766</td></tr>
<tr><td>alaska</td><td>9,157</td><td>statement</td><td>1,635</td></tr>
<tr><td>elect</td><td>8,867</td><td>seat</td><td>1,154</td></tr>
<tr><td>candidate</td><td>4,828</td><td>mission</td><td>261</td></tr>
<tr><td>.</td><td>.</td><td>.</td><td>.</td></tr>
<tr><td>.</td><td>.</td><td>.</td><td>.</td></tr>
<tr><td>.</td><td>.</td><td>.</td><td>.</td></tr>
</table>

4.2 Resource Representation

Resource descriptions required for resource selection by hubs use full-text representation with term frequency information because it provides the most comprehensive description for text content among common representations (Section 2.1.2) and its large size is not an issue for hubs equipped with high connection bandwidth and processing power. We adopt the format used by previous resource selection algorithms (Gravano et al. 1994) (Gravano and García-Molina 1995) (Callan et al. 1995) (Xu and Croft 1999) (Callan 2000) in distributed information retrieval for a resource description, 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. Table 4.1 provides an example of a provider’s resource description.

Resource selection by consumers cannot rely on full-text resource descriptions due to the limited network and computing resources consumers usually have. In addition, because the purpose of resource selection by consumers is to improve search efficiency without scarifying accuracy for characteristic queries representing persistent interests but not for all queries, consumers do not need to maintain comprehensive information about hubs as hubs do. Therefore, the format and acquisition of resource descriptions used for resource selection by consumers are different from those used by hubs. Full-text resource descriptions used for resource selection by hubs are introduced below. Resource descriptions used for resource selection by consumers is discussed in Section 4.4 where interest-based hub selection (as opposed to full-text hub selection) is presented.

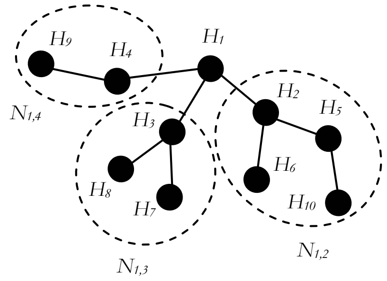


Figure 4.2 Three neighborhoods that can be reached from H_1.

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 by the hub. Since hubs work collaboratively in hierarchical P2P networks, neighboring hubs can exchange with each other their aggregate resource descriptions. However, because hubs' resource descriptions only have information for peers within one hop (providers directly connecting to them), if they are used by a hub to decide how to route query messages, the routing would not be effective when peers with relevant documents sit beyond this "horizon". Thus for effective hub selection, a hub must have information about what contents can be reached if the query travels several hops beyond each hub neighbor. This kind of information is represented by the resource description of a *neighborhood*, which is introduced in the following section.

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 4.2 illustrates the concept of neighborhood. Hub H_1 has three neighboring hubs H_2, H_3 and H_4. Thus it has three adjacent neighborhoods, labeled $N_{1,2}$, $N_{1,3}$ and $N_{1,4}$. A neighborhood's resource description provides information about the contents covered by all the hubs in this neighborhood. A hub uses resource descriptions of neighborhoods to route queries to its neighboring hubs.

Resource descriptions of neighborhoods are similar in functionality to 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 with their frequencies), while routing indices represent them with a small 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 a small controlled vocabulary.

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.

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

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.[11] 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) (Lu and Callan 2006a).

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).

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

This method is effective, but quite complex and expensive to use in practice for resource descriptions of large vocabularies. A simpler approach is to discard terms that occur in a static, predetermined list of terms that are common in general English ("*stopwords*").

In this dissertation, full-text resource descriptions are pruned by removing stopwords and the terms whose frequencies are below a certain threshold.

4.3 Resource Selection by Hubs

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

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

4.3.1 Resource Selection of Providers

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

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

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

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

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

$$\mathrm{P}(q \mid G) = \frac{tf(q, G) + 1}{numterms(G) + vocabularysize(G)} \tag{4.6}$$

4.3.2 Thresholding for Resource Selection of Providers

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

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 its resource selection threshold without supervision based on the information and functionality it already has. Because each hub has the ability to merge multiple retrieval results into a single, integrated ranked list (more details in Section 4.5), as long as result merging has reasonably good performance, we could assume that the top-ranked merged documents are "relevant", or at least appropriate for the user to consider. Thus the distribution of the top-ranked merged documents over neighboring providers should provide useful hints about the number of relevant documents each provider is likely to return. If we further assume that a hub is permitted to flood its neighboring providers with a *small* number of queries, it can use the results of these queries as training data. 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 to improve the overall search performance for future queries (Lu and Callan 2006a).

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

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

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

We introduce below individual-based threshold learning using normalized ranking scores, set-based threshold learning using original (unnormalized) or normalized ranking scores, and our attempt to take advantage of the complementary strengths of the three methods by using a hybrid approach.

Individual-based threshold learning with normalized ranking scores

For a training query, after a hub merges the documents returned by its neighboring providers, it can calculate how many "relevant" documents are returned by each provider 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 providers, the learned threshold will not work well when the top-ranked providers 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:[12]

$$E(j) = 1 - \frac{1 + b^2}{\frac{b^2}{R(j)} + \frac{1}{P(j)}} \tag{4.7}$$

$$R(j) = \frac{\sum_{i=1}^{j} n_{rel}(i)}{N_{rel}} \tag{4.8}$$

$$P(j) = \frac{\sum_{i=1}^{j} n_{rel}(i)}{\sum_{i=1}^{j} n(i)} \tag{4.9}$$

[12] For *b* = 1, the value of the *E* measure is equal to 1 minus the value of the *F* measure, which is the harmonic mean of recall and precision.

Table 4.2 An example of calculating the *E* measure for individual-based threshold learning.

Rank	Provider ID	# "Rel" Docs	# "Non-Rel" Docs	*R*	*P*	*E* (*b* = 3)
1	P105	6	44	0.1200	0.1200	0.8800
2	P050	9	41	0.3000	0.1500	0.7273
3	P032	8	42	0.4600	0.1533	0.6167
4	P156	6	44	0.5800	0.1450	0.5538
5	P088	7	43	0.7200	0.1440	0.4857
6	P015	4	46	0.8000	0.1333	0.4667
7	P350	4	46	0.8800	0.1257	**0.4500**
8	P212	2	48	0.9200	0.1150	0.4588
9	P137	2	48	0.9600	0.1067	0.4667
10	P076	2	48	1.0000	0.1000	0.4737

where *R*(*j*) is the recall calculated based on the set of documents returned by providers ranked 1st to *j*th, *P*(*j*) is the precision of the set of documents returned by providers ranked 1st to *j*th, *E*(*j*) is the *E* evaluation measure corresponding to *R*(*j*) and *P*(*j*), *b* is a parameter which reflects the relative importance of recall and precision, $n_{rel}(i)$ is the number of "relevant" documents returned by the *i*th ranked provider, *n*(*i*) is the total number of documents returned by the *i*th ranked provider, and N_{rel} is the total number of "relevant" documents for the query.

The *E* measure has values between 0 and 1; small values are preferred. 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. Because full-text federated search typically values high precision more than high recall due to efficiency concern, values of *b* greater than 1 are preferred.

To decide the threshold of ranking scores for the query, the hub goes down the list of providers sorted by their normalized ranking scores, stops at the provider that has the minimum *E* value and uses its ranking score as the threshold.

Table 4.2 provides an example to illustrate how to calculate the values of *R*(*j*), *P*(*j*) and *E*(*j*) for the 10 top-ranked providers at a hub with respective to a query. Providers are ranked by their ranking scores calculated using the hub's resource selection algorithm (Section 4.3.1). Each provider returns 50 documents. All the documents returned by the hub's neighboring providers are merged, and the 50 top-ranked documents in the merged result are treated as "relevant" documents. The columns of *# "Rel" Docs* and *# "Non-Rel" Docs* show respectively for each top-ranked provider how many of its returned documents occur in the set of "relevant" documents and how many don't. For instance, the 3 top-ranked providers return 6, 9, and 8 "relevant" documents respectively. Therefore, the value of *R*(3) is (6+9+8)/50 = 0.4600, and the value of *P*(3) is (6+9+8)/(50*3) =

0.1533. Since the minimum E value occurs at $j = 7$, the ranking score of the 7th ranked provider is used as the threshold with respect to the query.

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 with respect to a query:

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

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

$$S' = \frac{S - S_{\min}}{S_{\max} - S_{\min}} \tag{4.10}$$

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

3. The hub forwards the query to its neighboring providers and merges the lists of documents returned by these providers (Section 4.5);

4. The hub uses up to the r top-ranked documents in the merged result as the set of "relevant" documents to calculate $E(j)$ for each provider rank j, where r is a parameter of threshold learning (50 in our experiments); and

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

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

Set-based threshold learning with original or normalized ranking scores

Set-based threshold learning takes an approach similar to 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 neighboring providers. To use this approach, we need to solve three problems: i) define a utility function; ii) determine the criterion for a provider to be considered relevant with respect to a query; and iii) decide how to estimate the distributions of the ranking scores of relevant and non-relevant providers.

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

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

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

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

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

where $N_{rel}(\theta)$ and $N_{nonrel}(\theta)$ are the numbers of relevant and non-relevant providers respectively whose ranking scores are above threshold θ, P($s \wedge rel$) is the probability of a provider having score s and being relevant, P($s \wedge nonrel$) is the probability of a provider having score s and being non-relevant, P($s \mid rel$) is the probability of a relevant provider having score s, P($s \mid nonrel$) is the probability of a non-relevant provider having score s, P(rel) is the probability of a provider being relevant, α is the total number of provider neighbors, and max(s) is the maximum relevance-based ranking score of a hub's neighboring providers for the training queries. The integrals in Equations 4.13 and 4.14 are used when the corresponding probability distributions are represented with continuous probability density functions (e.g., Gaussian and uniform distributions). When discrete probability distributions are used, the integrals are replaced by sums.

For a given query, a provider 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 provider 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 providers, P($s \mid rel$) and P($s \mid nonrel$), at a hub, which we describe in the following paragraphs. Briefly, using original ranking scores requires maximum likelihood fittings of continuous Gaussian models for different groups of training queries, while using normalized ranking scores allows estimating an empirical discrete distribution for a single group containing all training queries.

When *original* ranking scores are used, because the score range is query-dependent, which may indicate a hub's provider neighbors' overall degree of relevance (total amount 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 P(*s* | *rel*) and P(*s* | *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 hub's resource description is a rough measure of the 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 hub 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 hub description. For each group of training queries, the empirical distributions of P(*s* | *rel*) and P(*s* | *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 P(*s* | *rel*) and P(*s* | *nonrel*) can be estimated from the aggregate results of all training queries. However, unlike the case with original ranking scores, score distributions of P(*s* | *rel*) and P(*s* | *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.

Table 4.3 gives an example of estimating P(*s* | *rel*) and P(*s* | *nonrel*) at a hub based on the results of a set of training queries. Queries are grouped by the ranges of the average probability of each query's terms in the hub's resource description. The relevance of each provider with respect to each query is determined based on whether it returned at least *n* "relevant" documents (using the top-ranked merged documents at the hub as the set of "relevant" documents). If original ranking scores of providers are used, then P(*s* | *rel*) is estimated for each query group based on the original scores of those providers considered "relevant" to the queries in the group. For instance, P(*s* | *rel*) for query group G1 is represented by a Gaussian distribution, whose mean and variance are computed using maximum likelihood estimation based on the scores in those rows with a "Y" in the last column for Q1 and Q3 (e.g., −5.8231, −6.1198, and −6.0031). If normalized ranking scores are used, then P(*s* | *rel*) is estimated for all the training queries as one group based on the normalized scores of those providers considered "relevant" to one or more training queries. Using the empirical discrete score distribution, P(*s* | *rel*) is the relative frequency that a normalized ranking score occurs among all the scores in the rows with a "Y" in the last column.

Usually P(*rel*) is estimated by maximum likelihood estimation using training data. However, because using a small number of 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 neighboring providers at the hub (very few relevant neighbors but a lot of non-relevant neighbors),

Table 4.3 An example of estimating P(*s* | *rel*) and P(*s* | *nonrel*) for set-based threshold learning.

Query ID	Query Group	Original Provider Score (log)	Normalized Provider Score	# "Rel" Docs Returned	Is A "Rel" Provider (*n* = 5)
Q1	G1	−5.8231	1.0000	6	Y
		−6.1198	0.9544	9	Y
		.	.	.	.
		.	.	.	.
		−12.3356	0.0000	0	N
Q2	G2	−7.2945	1.0000	7	Y
		−7.9821	0.9178	4	N
		.	.	.	.
		.	.	.	.
		−15.6632	0.0000	1	N
Q3	G1	−5.4768	1.0000	4	N
		−6.0031	0.9145	8	Y
		.	.	.	.
		.	.	.	.
		−11.6348	0.0000	1	N

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

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

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

2. If original ranking scores are used, go to the next step; otherwise, the hub normalizes ranking scores using Equation 4.10;

3. The hub forwards the query to its neighboring providers and merges the lists of documents returned by these neighbors;

4. The hub uses up to the *r* top-ranked documents in the merged result as the set of "relevant" documents to calculate for each provider neighbor how many of its returned documents are "relevant" in order to decide its relevance with respect to the query by comparing the number with *n*, where *r* is a parameter of threshold learning (50 in our experiments), and *n* is the minimum number of relevant documents a provider should provide in order to be considered relevant for a query;

5. After the hub finishes conducting the above steps for each training query, if original ranking scores are used, it defines non-overlapping groups spanning from 0 to the maximum term probability in the hub description so that all groups have roughly the same number of queries for training (at least 5 per group) and classifies each query into one of these groups based on the average probability of its terms in the hub description; 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 neighboring providers P(*s* | *rel*) and P(*s* | *nonrel*) for each query group using maximum likelihood estimation of Gaussian parameters if original ranking scores are used, or for a single query group consisting of all training queries using maximum likelihood estimation of empirical discrete distributions if normalized ranking scores are used; and

7. The hub uses Equations 4.11−4.14 to calculate θ^* that gives the maximum $U(\theta)$ value and uses it as its threshold for selection of its provider neighbors for queries that belong to the same query group.

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 hub's resource description and uses the statistics to classify 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, big differences between the original ranking scores of neighboring providers are exaggerated and thus some providers 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 providers. 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 selection is not reliable. If resource selection fails to rank most relevant provider neighbors above non-relevant providers, the top-ranked non-relevant providers will quickly increase *E* value, which can hardly be recovered even when relevant providers are included later on. Therefore, the optimal, low-valued *E* will occur early in the ranking and the method tends to select fewer providers than what is required to obtain a sufficient amount of relevant documents.

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

4.3.3 Resource Selection of Hubs

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

$$\mathrm{P}(N_i \mid Q) = \frac{\mathrm{P}(Q \mid N_i) \times \mathrm{P}(N_i)}{P(Q)} \propto \mathrm{P}(Q \mid N_i) \times \frac{numdocs(N_i)}{\sum_j numdocs(N_j)} \tag{4.15}$$

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

$$\mathrm{P}(Q \mid N_i) = \prod_{q \in Q} \frac{tf(q, N_i) + \mu \times \mathrm{P}(q \mid G)}{numterms(N_i) + \mu} \tag{4.16}$$

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

To avoid looking nearer or farther than what a query message can reach, the radius of each neighborhood that a hub looks ahead for hub-hub query routing should depend on the remaining value of the query message's TTL. In other words, TTL-dependent resource selection enables each hub to focus its selection on the neighboring hubs that can reach relevant contents within the predetermined search radius to avoid choosing a path leading to relevant contents located at a distance farther than the query message can travel. TTL-dependent hub selection can be applied when the network topology remains relatively static and hubs update their neighborhood descriptions in a synchronous manner. In this case, the descriptions of neighborhoods acquired in different iterations correspond to the contents covered in the neighborhoods of different radiuses (Section 4.2.3), so each hub can maintain neighborhood descriptions of different iterations in order to conduct TTL-dependent hub selection. However, when the network topology is highly dynamic (e.g., before topology evolution stabilizes) and neighborhood descriptions are updated asynchronously, each hub may have to rely on the most up-to-date neighborhood descriptions for TTL-independent hub selection.

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

4.4 Resource Selection by Consumers

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

The description of resource selection by consumers is divided into two sections. Section 4.4.1 describes the source, representation, and algorithm a consumer uses to construct a user model and measure hubs' performance on resource location for various interests, and Section 4.4.2 presents how to use the generated user model for initial hub selection.

4.4.1 User Modeling

User modeling for full-text federated search in a peer-to-peer network takes place at each individual information consumer due to the lack of a centralized server to monitor search activities in the network. Existing methods of user modeling for resource location in peer-to-peer networks (Ramanathan et al. 2002) (Sripanidkulchai et al. 2003) (Shao and Wang 2005) neither explicitly separate transient information needs from persistent interests nor distinguish between different interests (e.g., sports versus music). As a result, their performance measures for resource location are interest-independent, resulting in less effective resource location when a resource relevant to one interest is selected to answer queries for other interests or unrelated ad-hoc information requests. To remedy the problems, similar to the approach taken in (Voorhees et al. 1995), query clustering is used to group past queries in modeling a user's different interests, and each query cluster represents a *topic of interest*. New queries similar to any of the existing query clusters are considered characteristic queries representing persistent interests; otherwise, they are regarded as uncharacteristic queries for transient information needs. The interest-dependent performance is measured for each hub that provided search results to this consumer in the past, which is dynamically updated whenever new results are available. For a hub that covers contents related to multiple topics of interest, its performance for each topic is measured independently of the other topics. Figure 4.3 provides an algorithmic description of how a consumer updates its user model using query clustering and measures the hubs' resource location performance based on the search results for a query. More details are provided below.

Source and representation for query clustering

Query clustering requires a representation for each query/cluster, and a similarity measure between queries and clusters. Because the small number of query terms does not provide a reliable basis for clustering queries effectively, a commonly used method to measure query similarity in Web retrieval is to count the number of commonly retrieved documents for the queries (Glance 2001) (Wen et al. 2002). This method may work well if the task is to group queries that are very similar. However, to group queries by interest, it is quite likely that two queries that express similar interests in a general topic (e.g., music) may not have any retrieved document in common even though the vocabularies of their retrieved documents may have significant overlap. Therefore, it is more appropriate to measure query similarity based on the *contents* of the documents returned for each query. Which retrieved documents to choose in generating a representation for the corresponding query depends on whether and what type of feedback is available. With explicit relevance feedback from the user, documents relevant to the query are selected. When feedback is implicit in the form of mouse clicks, the clicked documents are treated as relevant documents. The top-ranked merged documents are chosen in the last resort when neither explicit nor implicit feedback is available.

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

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

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

After stopwords are removed and stemming is conducted, the contents of the chosen documents are used to generate a maximum likelihood unigram language model to represent the corresponding query. The representation of a query cluster is the aggregation of its members' language models. The similarity between a query and a cluster is measured by the Kullback-Leibler divergence between their representations.

Algorithm for query clustering

The choice of the clustering algorithm is guided by several characteristics of query clustering in peer-to-peer networks. First, because the sets of queries used for clustering are highly dynamic, the clustering algorithm should be incremental. Second, since the size of the query log at each individual information consumer is much smaller compared with the query logs of Web search engines, the clustering algorithm should be able to work well with limited data. Third, the algorithm should not require the number of clusters or the maximum size of each cluster to be set manually as it is unreasonable to assume that these parameters can be determined in advance. Based on the above considerations, a single-pass non-hierarchical clustering algorithm is chosen to incrementally update existing clusters to include new queries when their representations are similar to the old ones, or create new clusters when they are sufficiently different in order to capture the user's new interests. Neither the number of clusters nor the size of each cluster is predetermined. Specifically, a clustering threshold $T_{cluster}$ is used to determine whether to include a new query into existing clusters or to create a new cluster. Among all the clusters whose K-L divergence-based distance measures to a query's representation are smaller than $T_{cluster}$, the query chooses to join the largest cluster in order to minimize the "noise" introduced by small clusters of uncharacteristic queries.

The total number of query clusters can be limited in order to control the amount of resources dedicated by an information consumer to process and store the language models used to represent the clusters. Although in most cases a consumer may not find it necessary to limit the number of query clusters (the average size of the representation for a query cluster is 69KB in our experiments), associating each cluster with a time stamp and removing infrequently used clusters can reduce clusters of uncharacteristic queries and effectively model the user's interest shift. However, the constraint on N_{max} cannot be too tight because if N_{max} is too small, existing query clusters need to be constantly removed in order to make room for new clusters, resulting in a high cluster turnover rate which would prohibit useful clusters representing the user's persistent interests from being formed and stabilized. When the number of query clusters exceeds N_{max}, clusters among the r least recently used clusters are removed in an ascending order of cluster size until the number of query clusters drops to N_{max}. Small old clusters are removed before big old clusters because they are more likely to be clusters of uncharacteristic queries formed by chance.

Interest-dependent measure for hubs' resource location performance

In previous research on using search history to improve federated search performance in P2P networks (Ramanathan et al. 2002) (Sripanidkulchai et al. 2003) (Shao and Wang 2005), search performance is measured by the number of documents returned for each query. For the known-item search that is common in P2P networks sharing music, videos, and software, this appears to be an appropriate measure since typically the search either returns relevant documents or returns no document at all. In contrast, full-text federated search is very likely to return non-relevant documents, so the number of documents returned is no longer a good measure of search performance. Because the top-ranked documents are more likely to be relevant than most lower-ranked documents, when no feedback is available, the information about how many documents returned by a hub appear among the overall top-ranked merged documents at a consumer is a more reliable indicator of the hub's performance for a query.[13] Therefore, our approach uses this information as a surrogate for relevance feedback to measure each hub's performance on resource location for interest-based hub selection. A hub's resource location performance for a query cluster is its average performance for the queries in the cluster.

4.4.2 Resource Selection of Hubs

When a query is issued, its query terms are used as its representation in determining which existing query clusters it is most similar to, measured by the Kullback-Leibler divergence between the query and existing query clusters that exceed a certain size S_{min} (to avoid classifying queries to clusters of uncharacteristic queries formed by chance and to make the description of the topic represented by each cluster more reliable). A classification threshold $T_{classify}$ is used to distinguish characteristic queries representing long-term interests from uncharacteristic queries representing transient information needs.

Different search strategies are applied to characteristic versus uncharacteristic queries. A characteristic query is issued to the hubs selected using *interest-based hub selection* with a small search radius, which selects hubs based on their measured resource location performance for the query clusters the query is most similar to. An uncharacteristic query is issued to randomly selected hub(s) with a default, larger search radius.

A weighted *k*-nearest neighbor approach is used to increase the robustness of interest-based hub selection, where the value of *k* is determined by $T_{classify}$ and the weights are related to the similarities between the query and the clusters. Each hub's weighted performance values for different clusters are accumulated and hubs are ranked and selected according to the accumulated performance. Figure 4.4 describes in detail the initial hub selection conducted by a consumer for a query.

The effectiveness of interest-based hub selection depends on whether the hubs capable of locating relevant contents efficiently and effectively for past queries perform well for future queries that express similar interests. A hierarchical P2P network in which the information providers with similar contents connect to the same hubs (content-based locality) can best support effective interest-based hub selection since contents relevant to similar interests tend to be similar to each other. The network can provide this property by using dynamic topology evolution (Chapter 6) to regulate its content placement.

[13] Note that this heuristic is essentially the same in nature as the one used to learn thresholds on the number of providers a hub selects for a query (Section 4.3.2).

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

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

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 also required in recalculating document scores. To avoid the cost of acquiring and maintaining global corpus statistics at each hub which require aggregating information from *all* the hubs in the network, we propose for each hub to use the aggregation of the resource descriptions of its neighboring providers and neighborhoods to substitute for the corpus statistics. We refer to the algorithm that uses summary statistics of the returned documents (Kirsch's algorithm) and the aggregation of resource descriptions as corpus statistics to recalculate document scores as the *extended Kirsch's algorithm*.

A consumer may also need to merge results returned by multiple hubs. Because consumers don't maintain comprehensive information about the contents of other peers and corpus statistics as do hubs, they cannot use advanced result-merging algorithms. Thus only simple, but probably less effective, merging methods can be applied at consumers. For example, results can be merged directly based on the document scores returned by hubs ("*raw score merge*") or in a round robin fashion.

4.6 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 architecture 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 main new features of our network search model are:

1. We define the concept of a *neighborhood* and propose to use exponentially decayed resource descriptions of neighborhoods for resource selection of hubs in order to avoid "shortsighted" query routing that cannot see beyond the horizon of direct neighbors at the hub level;

2. We propose several *unsupervised threshold learning* methods for each hub to learn its query-specific provider selection threshold autonomously and adaptively so that extensive heuristic threshold tunings are no longer needed for resource selection of providers in decentralized and dynamic environments;

3. We develop an approach based on dynamic, adaptive query clustering for each information consumer to *learn a user model* representing a person's various short-term and long-term interests based on past search results, that can be used to improve resource selection performance for future queries; and

4. We modify 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.

Among the above new development, automatic threshold learning for resource selection of providers and user modeling that distinguishes between persistent and transient interests are particularly significant because they tackle difficult problems that existing approaches either have avoided or have not solved well. They are not only useful to federated search in P2P networks, but may also benefit other applications that require thresholding or user modeling.

Although our network search model is expected to be more effective, efficient and robust when the network evolution model (Chapter 6) is used to dynamically evolve the network topology into one with the desired search-enhancing properties (Section 3.2), without the support from network evolution, the network search model is still capable of providing a better combination of accuracy and efficiency for full-text federated search than existing common alternatives, as demonstrated in next chapter.

Chapter 5

EVALUATION OF NETWORK SEARCH MODEL

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 Datasets

There has been no standard data for evaluating the performance of full-text federated search in P2P networks, so we developed two *P2P testbeds* based on the TREC WT10g and .GOV2 test collections, two standard research collections associated with IR research at the TREC conference[14]. We briefly describe below how we generated the contents, queries, and topologies for simulating federated search in a medium-sized hierarchical P2P network of 2,500 text digital libraries and a large-sized hierarchical P2P network of 25,000 text digital libraries.

5.1.1 Contents

WT10g is a 10 gigabyte, 1.69 million English Web document collection used for TREC Web Tracks in 2000 and 2001 (Hawking 2000) (Bailey et al. 2001). 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.[15] The maximum, minimum, and average numbers of documents in a selected collection are 26,505, 8, and 568 respectively. Each of the 2,500 collections defined a provider (text digital library) in a medium-sized hierarchical P2P network (Lu and Callan 2003). This testbed has also been used by other researchers to study and evaluate federated search in P2P networks (Renda and Callan 2004) (Klampanos et al. 2005) (Castiglion and Melucci 2007).

.GOV2 consists of 25 million documents crawled from .gov sites in early 2004, including HTML and text, along with the extracted contents of PDF, Word, and postscript files (Clarke et al. 2004). This collection was used for TREC Terabyte Tracks in 2004 and 2005. Compared with the websites contained in WT10g, .gov websites are generally more homogeneous and have higher

[14] http://trec.nist.gov/

[15] http://boston.lti.cs.cmu.edu/callan/Data/P2P/trecwt10g-2500-bysource.v1.txt.gz

Table 5.1 Examples of collections used to define providers in P2P networks.

Source	Provider URL	# Docs
WT10g	aqui.ibm.com	96
WT10g	www.solutions.net	590
WT10g	www.cityscape.co.uk	11,946
.GOV2	smmc.ca.gov	204
.GOV2	www.bls.gov/mfp	47
.GOV2	www.gov.state.ak.us/ltgov/elections	1,504

Table 5.2 Examples of TREC queries.

Number	Content
452	do beavers live in salt water
454	parkinson's disease
462	real estate and new jersey
478	baltimore
716	spammer arrest sue
733	airline overbooking
757	murals
777	hybrid alternative fuel cars

quality contents. The same URL-based partitioning approach was used to divide the data into collections, and 25,000 collections with a total number of 11,218,349 documents were selected to define the information providers in a large P2P network.[16] The maximum, minimum, and average numbers of documents in a selected collection are 694,505, 5 and 449 respectively.

Table 5.1 includes examples of the providers in the networks.

5.1.2 Queries

The queries provided by the U. S. National Institute for Standards and Technology (NIST) for the WT10g and .GOV2 test collections are TREC topics 451-550 and 701-800 respectively. Both topic sets came with standard TREC relevance assessments that indicate which documents are relevant to each query. We refer to the title fields of these queries used in our experiments as *TREC queries*. Table 5.2 shows examples of TREC queries.

[16] http://boston.lti.cs.cmu.edu/callan/Data/P2P/trecgov2-25000-bysource.v1.txt.gz

To study the performance of different methods for federated search in various P2P networks, sometimes a large number of queries is desired and clearly TREC queries are far from enough. For the medium-sized network, because there are no available queries targeted specifically at the collections selected from WT10g, to generate a large amount of queries in a controlled manner, we extracted key terms from the documents in WT10g 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 $\mathrm{P}_{\mathrm{emp}}(t \mid d)$ that a term occurs in a document as a linear interpolation of the probability $\mathrm{P}_{\mathrm{core}}(t \mid d)$ that the term is generated by the unigram document language model, and the probability $\mathrm{P}(t \mid \mathit{background})$ that the term is generated by the background (general English) model:

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

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

The bigram document language model approach uses $\mathrm{P}(t_1, t_2 \mid d)$ to measure the importance of a "phrase"[17] to the document. It is calculated as a mixture of maximum likelihood estimates:

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

where $c(\bullet)$ denotes count, $\mathrm{P}(t_1 \mid d)$ and $\mathrm{P}(t_2 \mid d)$ are maximum likelihood estimates (with Laplacian smoothing) 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 $\mathrm{c}(t_1, t_2 \mid d) / c(t_2 \mid d)$ are un-smoothed empirical estimates of $\mathrm{P}(t_2 \mid t_1, d)$ and $\mathrm{P}(t_1 \mid t_2, d)$ 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:

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

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;

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 collection using the approach described above (Lu and Callan 2003).[18] We refer to these automatically generated queries as *WT10g queries*. Table 5.3 shows the distribution of query lengths and randomly selected examples of WT10g queries for different query lengths. Experimental results not reported here indicated that although the search performance for the individual queries varied, the average search accuracy over random subsets of WT10g queries had similar values as long as each subset included at least 1,000 queries, so we use 1,000 WT10g queries to evaluate full-text federated search in the medium-sized network. The selected subset of queries has the same distribution of query lengths as shown in Table 5.3.

For the large-sized network, we selected 1,000 queries from a query set provided by AOL, which were collected from a one-month period on AOL Search in April 2006. The set consists of 2,149,827 web queries where each query was collected when a user clicked on a search result from a *.gov domain. For each query, the domain clicked and the frequency of the action are also provided. Our sampling strategy was to sample among those queries whose corresponding clicked domains are contained in the contents of the large-sized network, with different percentages for queries of different frequencies (as shown in Table 5.4). The distribution of query lengths for the selected queries are similar to that of WT10g queries. We refer to these queries as *GOV queries*.

[18] http://boston.lti.cs.cmu.edu/callan/Data/P2P/trecwt10g-query-bydoc.v1.txt.gz

Table 5.3 Distribution and randomly selected sample WT10g queries.

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

Table 5.4 Distribution and randomly selected sample GOV queries.

Frequency on AOL	Distribution	Sample Query
0-100	30.00%	foreclosure houses
100-250	10.00%	California commission on teacher credentialing
250-500	10.00%	citizenship
500-750	10.00%	john adams
750-1,000	10.00%	national institute of health
1,000-2,500	10.00%	us savings bonds
2,500-5,000	5.00%	bird flu
5,000-7,500	5.00%	nasa
7,500-10,000	5.00%	irs forms
10,000-	5.00%	social security

5.1.3 Topologies

This chapter focuses on comparing our network search model with existing common alternatives which are widely used in peer-to-peer networks without regulated content placement or carefully controlled topology evolution. Therefore, the topologies for both networks were randomly generated (i.e., random hub-hub, hub-provider, and hub-consumer topologies). The results of our network search model in different types of network topologies are presented in Chapter 7 as part of the evaluation on the effectiveness of the network evolution model.

The number of hubs was chosen to be 32 for the medium-sized network. Each hub has 4 hub neighbors. The number of hubs in the large network is 256, eight times as many as the number of hubs in the medium-sized network. Each hub has at least 4, at most 16, and on average 9 hub neighbors. Hence for both networks, each hub is directly connected to quite a small percentage of the total number of hubs. The diameter (the maximum number of hops between any two hubs in the network) of the hub-hub topology is 4 for the medium-sized network and 5 for the large-sized network. Each information provider or consumer connects to only 1 hub.

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. Furthermore, a "static" network setting (i.e., fixed topology without peer arrivals, departures or failures) is assumed in order to compare different methods of resource representation, resource selection, and result merging for federated search without the impact of dynamic content or topology change.

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. *Average precision at given document cut-off values* is standard rank-based measure commonly used to evaluate the performance of full-text ranked retrieval in distributed information retrieval, 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). Average precision at given document cut-off values produces a curve of multiple points to evaluate the performance of the top-ranked documents. However, to evaluate federated search in peer-to-peer networks, a single precision value is often preferred in order to conveniently compare different methods in various network settings. *Average precision over a range of document cut-off values* was chosen 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.

In addition to rank-based precision, *set-based recall* (Baeza-Yates and Ribeiro-Neto 1999) is also used to evaluate the overall percentage of relevant documents full-text federated search is able to retrieve when only part of the network is reached. It is calculated as:

Table 5.5 Relevant content distributions of different query sets in the networks.

Query Set	Average # Relevant Providers	Average # Relevant Hubs
TREC 451-550	25 (of 2,500)	14 (of 32)
TREC 701-800	99 (of 25,000)	76 (of 256)
WT10g	4 (of 2,500)	3 (of 32)
GOV	68 (of 25,000)	58 (of 256)

$$recall = \frac{|r|}{|A|} \tag{5.3}$$

where A is the set of relevant documents for a query, and r is the intersection of A and the set of documents returned by search in the P2P network. |•| denotes the size of the set. Results are averaged over a set of queries.

Measuring search accuracy typically requires relevance judgments. The standard relevance assessments supplied by NIST can be used for TREC queries. For WT10g and GOV queries, because it is expensive to obtain relevance judgments for a large amount of queries, we chose to use the retrieval results from a single large collection as the baseline ("*single collection*" baseline) to measure how well federated search in P2P networks could locate those documents considered very relevant by centralized search. The single large collection was constructed by aggregating all of the providers' contents in the P2P network. The top-ranked documents retrieved from this single large collection for a query are treated as the set of "relevant" documents for this query. The numbers of "relevant" documents per query are 50 for the medium-sized network and 200 for the large-sized network. These values were chosen based on the average numbers of relevant documents for TREC queries 451-550 and 701-800 respectively. Because evaluation based on the single collection baseline essentially measures the percentage of overlap between the documents returned by centralized search and those by federated search in the P2P network, we refer to the measures calculated using the "single collection" baseline as "*overlap precision*" or "*overlap recall*" to distinguish them from values obtained using real relevance judgments. Although this methodology is not ideal, it is not unreasonable because distributed retrieval systems are not yet better than the "single collection" baseline, so the ability of a P2P network to mimic a good centralized search engine is an acceptable indicator of its performance. 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).

Given the real or pseudo relevance judgments described above, Table 5.5 shows the relevant content distributions of different sets of queries in the medium-sized and large-sized networks with random topologies. These values suggest about one relevant provider per relevant hub, which is what we would expect from a random topology. WT10g queries have the most concentrated relevant contents, while TREC queries 701-800 have the most scattered relevant contents. The big differences in the relevant content distributions of different sets of queries may affect the relative effectiveness of various resource selection methods, as we shall see in Section 5.3.

Search efficiency is measured by the average number or percentage of the hubs or providers reached by the query messages for each query.

5.3 Experimental Results

As mentioned earlier, this chapter focuses on evaluating our network search model in peer-to-peer networks with randomly-generated topologies, to demonstrate that the network search model can provide a better combination of accuracy and efficiency than existing common alternatives even without the support of the network evolution model. Because the effectiveness of interest-based initial hub selection conducted by information consumers largely relies on regulated content placement and carefully controlled topology evolution, we evaluate it together with our network evolution model in Chapter 7 and focus here on the other main components of full-text federated search in a hierarchical P2P network, e.g., resource representation, resource selection of providers/hubs by hubs, and result merging. 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 flooding for query routing, 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 five sections to the experimental results with regard to evaluating our approaches to resource selection of providers (Section 5.3.1), thresholding for resource selection of providers (Section 5.3.2), resource selection of hubs (Section 5.3.3), resource representation (Section 5.3.4), and result merging (Section 5.3.5). Table 5.6 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. "Full-text" in the table refers to our approach to federated search, which uses the K-L divergence resource selection algorithms described in Section 4.3 for resource selection by hubs, or the K-L divergence document retrieval algorithm (Ogilvie and Callan 2001) for document retrieval. The other methods listed in the table will be described in the corresponding sections.

For all the experimental results reported here, each query was issued to the network by a consumer connecting to a hub located *farthest* on average from relevant content in the network. For resource representation, because reducing the sizes of resource descriptions by pruning low-frequency terms is very simple and efficient to use in practice, and previous experimental results indicate that it enables dramatic savings in communication and storage costs without degradation in search

Table 5.6 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
Document retrieval	Full-text	Full-text	Full-text	Full-text	Full-text
Description for hub-provider routing	Pruned full-text	Pruned full-text	Pruned full-text	Pruned full-text	Pruned full-text
Ranking of providers	Full-text / Random / Size-based / Flooding	Full-text	Full-text	Full-text	Full-text
Selection of providers	Top 5%−100%	Learned thresholds / Top 5%−100%	Learned thresholds	Learned thresholds	Learned thresholds
Ranking of hubs	Flooding	Flooding	Full-text / Random / Degree-based / Size-based / Flooding	Full-text	Full-text
Selection of hubs	N/A	N/A	Top 1	Top 1	Top 1
Description for hub-hub routing	N/A	N/A	Decayed neighborhood	Direct neighbor / Decayed neighborhood / Non-decayed neighborhood	Decayed neighborhood
Result merging	Extended Kirsch's	Extended Kirsch's	Extended Kirsch's	Extended Kirsch's	Centralized merge / Extended Kirsch's / Raw score merge

accuracy (Lu and Callan 2003a), the pruned resource descriptions were used instead of the full descriptions. In particular, all the terms that occurred less than twice in a provider's description and all the terms that occurred less than five times in a hub's description were discarded. Table 5.7 shows the average sizes of different resource descriptions after pruning for the two testbeds. The average size of a provider description in the large network was smaller than that in the medium-sized network due to the smaller average number of documents per provider and the more coherent

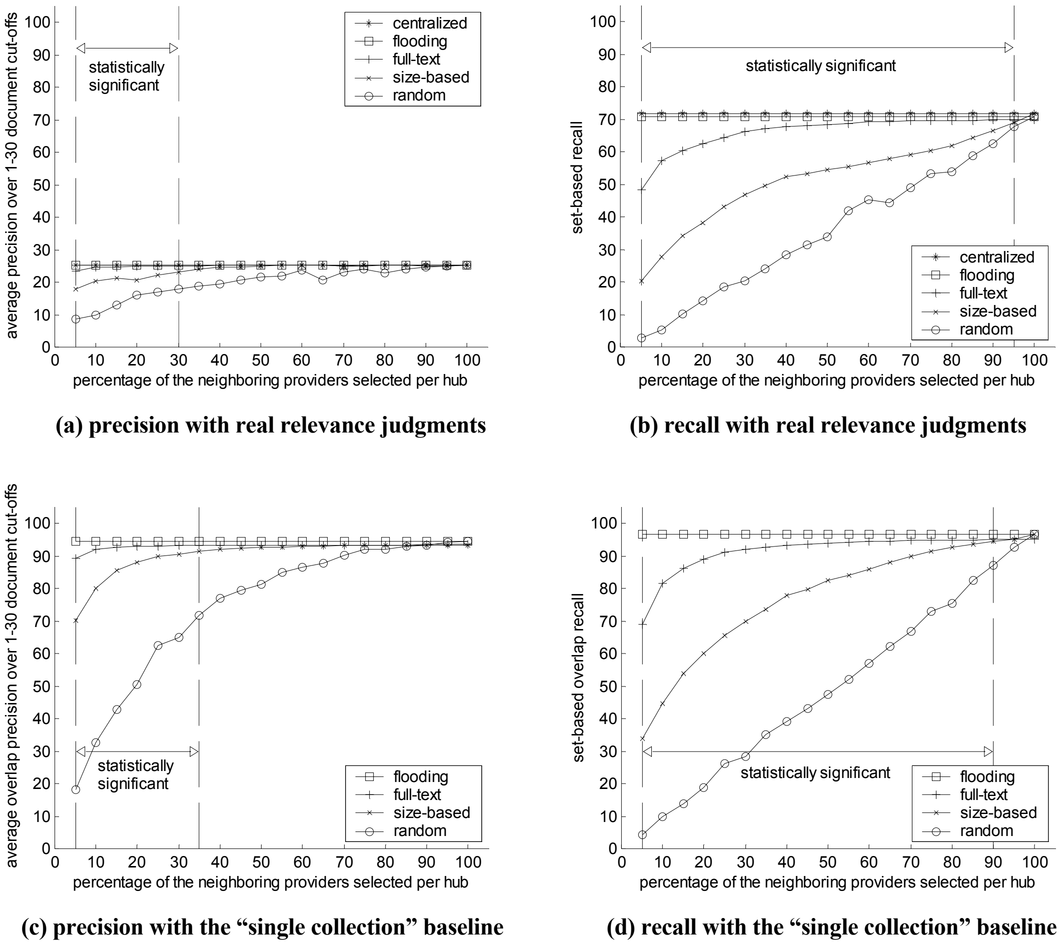
Table 5.7 Average sizes of pruned resource descriptions.

Resource	Testbed	Average Size
Provider	Medium	58KB
	Large	30KB
Hub	Medium	530KB
	Large	585KB
Neighborhood	Medium	5.8MB
	Large	15.5MB

contents in the collections of the large network. However, because the large network had on average more neighboring providers and hubs per hub, its average sizes of hub and neighborhood descriptions were larger. For resource selection, the smoothing parameter μ 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 μ 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. Each hub used its result merging algorithm to merge the ranked lists returned by its neighboring providers and returned the top-ranked documents (50 for the medium-sized network and 200 for the large-sized network) to the consumer that issued the query. The aggregation of the descriptions a hub maintained for its neighbors was used as background model for resource selection and as substitute corpus statistics for result merging conducted by the hub. Each consumer merged the ranked lists returned by multiple hubs directly using the raw score merge method (Section 4.5).

5.3.1 Resource Selection 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, size-based selection or flooding for hub-provider query routing. Each query was broadcast to all the hubs in the network. On receiving a query, each hub i) used the K-L divergence resource selection algorithm to rank its neighboring providers (Section 4.3.1) and forwarded the query to the top-ranked providers up to a certain percentage of the total number of its neighboring providers ("*full-text resource selection*"), ii) randomly forwarded the query to a subset of its neighboring providers to yield a similar number of query messages as i) ("*random selection*"), iii) ranked its neighboring providers by each one's collection size (i.e., total number of documents) and forwarded the query to the top-ranked providers up to a certain percentage of the total number of its neighboring providers ("*size-based selection*"), or iv) flooded the query to all neighboring providers ("*flooding*").



(a) precision with real relevance judgments

(b) recall with real relevance judgments

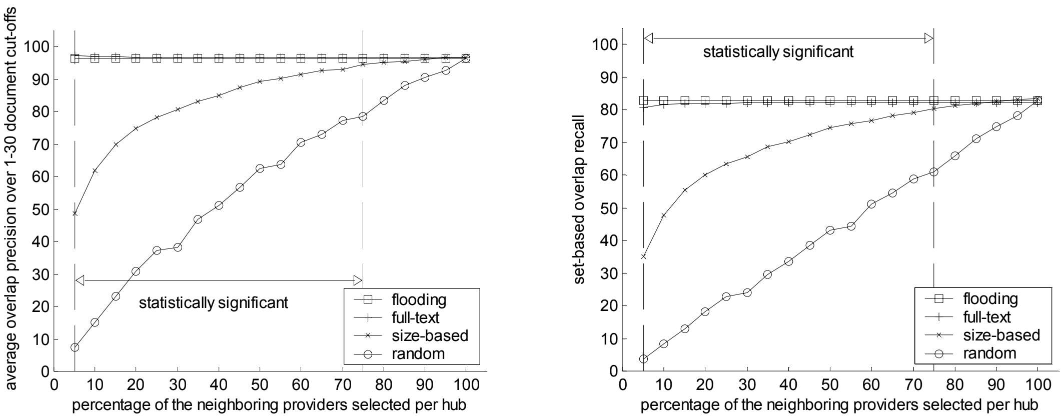
(c) precision with the "single collection" baseline

(d) recall with the "single collection" baseline

Figure 5.1 The search performance of different methods of hub-provider query routing for TREC queries 451-550 in the medium-sized network.

In order to focus the comparison on the effectiveness of different methods for hub-provider query routing, we assumed that no matter what method was used for hub-provider query routing, each hub used the extended Kirsch's algorithm based on the substitute corpus statistics generated by aggregating the resource descriptions of neighboring providers each hub acquired for result merging. In real operational environments, because flooding and random selection don't have to acquire resource descriptions from providers, and size-based selection only requires size information instead of full-text resource descriptions, they may not have comprehensive corpus statistics for result merging so that only simple, but probably less effective, merging methods (such as raw score merge) can be applied, yielding even worse results than those shown here.

Figures 5.1-5.4 show the experimental results for different sets of queries and network sizes using different methods of hub-provider query routing. The measured rank-based or set-based accuracy



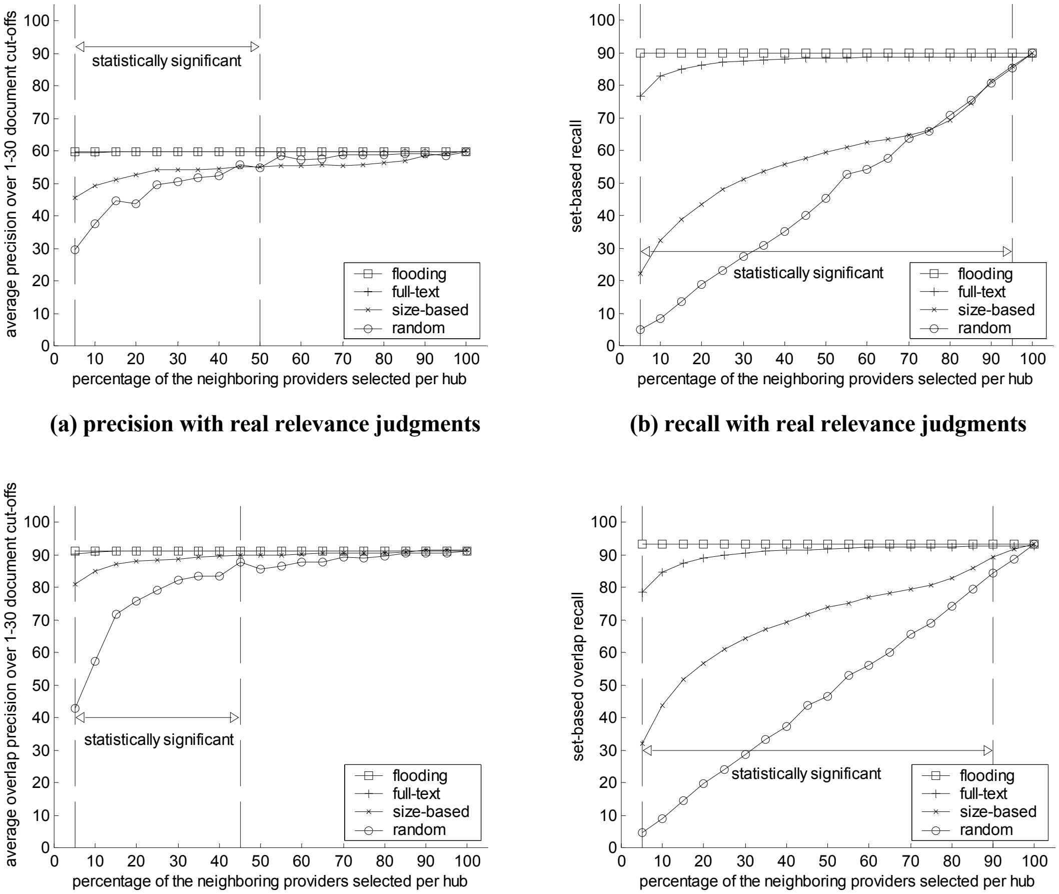
(a) precision with the "single collection" baseline **(b) recall with the "single collection" baseline**

Figure 5.2 The search performance of different methods of hub-provider query routing for WT10g queries in the medium-sized network.

values (y-axis) are plotted against the percentage of the neighboring providers each hub selected to route query messages (x-axis). Since the percentage of the neighboring providers selected per hub was linearly correlated with the number of query messages routed in the network, it can be regarded as an indirect measure of search efficiency. The higher the percentage, the lower the efficiency. Flooding always had a value of 100% for the x coordinate since all of the neighboring providers were selected by each hub. However, for greater contrast between the accuracy of flooding and those of other resource selection methods, the performance of flooding was depicted in the figures by a straight line instead of a single point.

Using the evaluation results of individual queries as samples, paired two-sided sign tests were applied to test whether the difference between full-text resource selection and random or size-based selection in search accuracy was statistically significant at a given level of search efficiency. The vertical dashed lines in the figures mark the ranges within which full-text resource selection had a statistically significant improvement at the 0.01 significance level.

Figures 5.1 (a)-(b) depict the results for TREC queries 451-550 in the medium-sized P2P network using real relevance judgments. Compared with flooding, random selection could greatly improve search efficiency at the cost of reducing accuracy. Its increase in search accuracy was nearly linear as the percentage of the neighboring providers selected per hub increased, indicating that random selection didn't have the ability to effectively identify the providers with high likelihood of providing relevant content and restrict query routing to them. In contrast, full-text resource selection could significantly improve search efficiency without degrading accuracy much. Its growth in search accuracy was much faster at the beginning when only a small percentage of the providers were selected. Furthermore, its average precision over 1-30 document cut-offs could be very close to that of flooding even if its set-based recall was lower (which might be caused by the



(a) precision with real relevance judgments

(b) recall with real relevance judgments

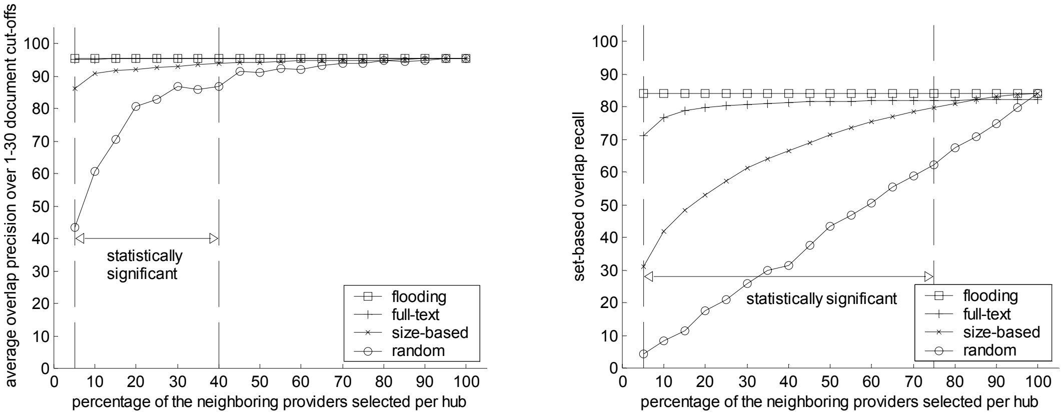
(c) precision with the "single collection" baseline

(d) recall with the "single collection" baseline

Figure 5.3 The search performance of different methods of hub-provider query routing for TREC queries 701-800 in the large-sized network.

wide distribution of relevant contents in the network), demonstrating that using term frequency information enabled resource selection to effectively estimate each provider's likelihood of satisfying the user's information need so that search didn't have to sacrifice efficiency for accuracy. The effectiveness of size-based selection relied on the assumption that information providers with more documents were more likely to contain relevant documents, which was the case for some queries but not for all queries. Therefore, although size-based selection was more effective than random selection for the tested queries, its search accuracy exhibited high variance among individual queries, and was far from comparable to the performance of full-text selection (especially recall) when reaching a small percentage of the providers.

Figures 5.1 (a)-(b) also include the evaluation results of using a centralized search engine to retrieve from the single collection aggregating all of the providers' contents (Section 5.2) so that we can



(a) precision with the "single collection" baseline

(b) recall with the "single collection" baseline

Figure 5.4 The search performance of different methods of hub-provider query routing for GOV queries in the large-sized network.

compare the performance of centralized search and federated search. Considering that the overall performance of full-text federated search was affected by the performance of every single component (resource representation, resource selection, document retrieval, result merging), and federated search mostly only required a small number of selected providers to each return up to 50 documents and merged them without any global corpus statistics, the very similar performance of centralized search and federated search using full-text resource selection is an encouraging sign of the effectiveness of full-text federated search in P2P networks.

To demonstrate the effectiveness of using the "single collection" baseline to evaluate search accuracy, Figures 5.1 (c)-(d) plot the results for TREC queries 451-550 in the medium-sized network based on the "single collection" baseline instead of real relevance judgments. If we compare them to Figures 5.1 (a)-(b), we can see that although the values were in different scales, the shapes of the curves look so similar that the same conclusions can be drawn from them with respect to the relative effectiveness of different methods. Therefore, the "single collection" baseline was effective in evaluating federated search performance.

Figures 5.2 (a)-(b) display the results for the 1,000 WT10g queries in the medium-sized P2P network using the "single collection" baseline. Compared with the results for TREC queries 451-550, the performance difference between full-text and random or size-based selection of providers was bigger and that between full-text and flooding was smaller. This can be explained by the fact that relevant contents for WT10g queries are more concentrated because they are distributed to a smaller number of information providers in the medium-sized network than those for TREC queries, making it more difficult for random or size-based selection to hit the providers with relevant contents by luck, but easier for full-text resource selection to locate most relevant contents even with only a small percentage of the providers selected. Despite the above difference, similar conclusions with regard to the relative effectiveness of different resource selection methods can be drawn using TREC queries and WT10g queries.

Figures 5.3 (a)-(d) show the experimental results for TREC queries 701-800 in the large-sized P2P network using real relevance judgments or the "single collection" baseline. The figures look very similar to those for TREC queries 451-550 but federated search in the large-sized P2P network had higher values for both precision and recall. The better results of the large-sized network were due to the facts that the contents from .gov sites usually have higher quality and less noise than the contents provided in the WT10g collection, and the number of relevant documents is in general much larger for TREC queries 701-800 than for TREC queries 451-550.

The results for the 1,000 GOV queries in the large-sized P2P network using the "single collection" baseline are depicted by Figures 5.4 (a)-(b). Compared with the results of WT10g queries in the medium-sized network, the difference between full-text and random or size-based selection was smaller for GOV queries in the large-sized network, again due to the differences in the distributions of "relevant" contents for different sets of queries. The distributions of "relevant" contents for GOV queries are much less concentrated in the large-sized P2P network, giving random or size-based selection more chances to hit relevant providers. Even so, random or size-based selection still significantly underperformed full-text resource selection in recall.

In summary, full-text resource selection gave a much better combination of search accuracy and efficiency than random selection, size-based selection or flooding for hub-provider query routing. With similar search efficiency, the improvement of full-text selection in precision over random or size-based selection was statistically significant when a small-to-medium percentage of the providers (< 30%-50%) were selected. Its advantage in recall was statistically significant in a much wider range of settings. In addition, although 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 as demonstrated by the not-so-high accuracy of centralized search for TREC queries 451-550), the same conclusion regarding the relative effectiveness of various methods could be drawn using either WT10g or GOV queries with the "single collection" baseline, or TREC queries with real relevance assessments or the "single collection" baseline. This indicates that the automatically generated queries and the "single collection" baseline are useful resources in studying federated search in P2P networks.

5.3.2 Thresholding for Resource Selection of Providers

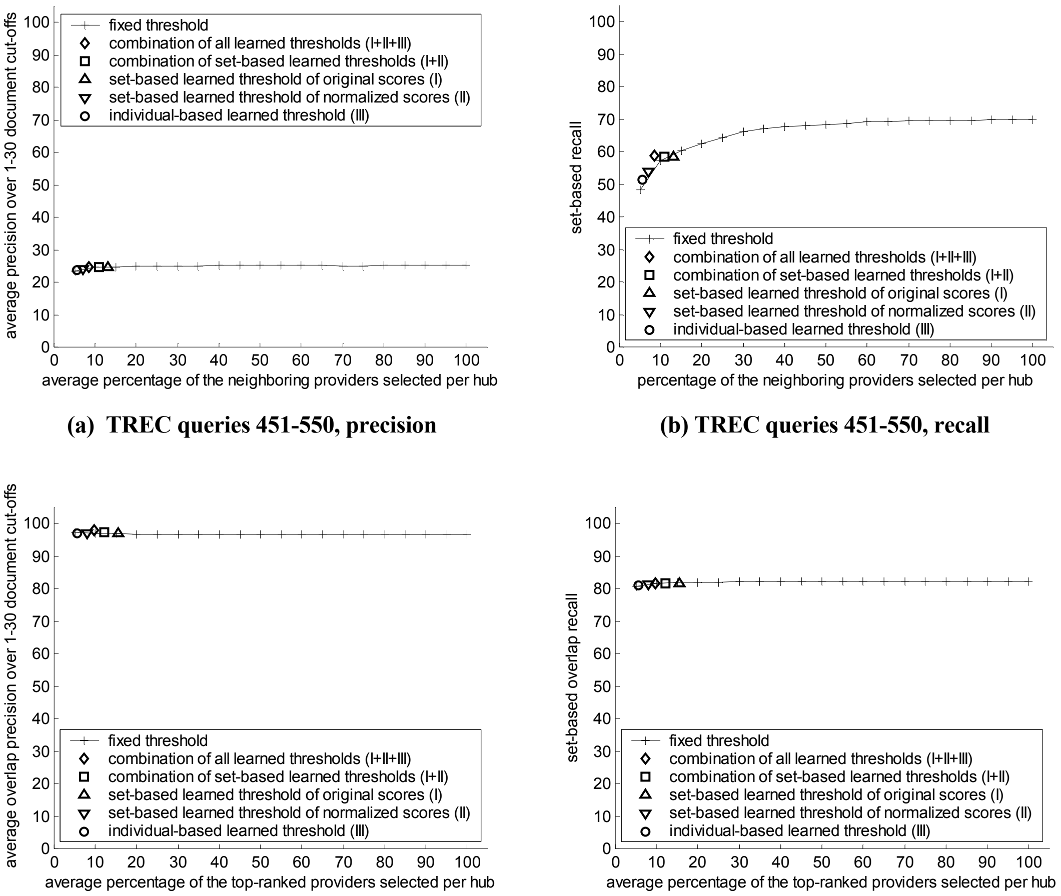
The effectiveness of different threshold learning methods (Section 4.3.2) for resource selection of providers was evaluated by comparing the performance of federated search using resource selection of providers based on the learned thresholds with that based on fixed thresholds. Resource selection of providers based on a learned threshold used i) set-based threshold learning with original ranking scores (*method I*), ii) set-based threshold learning with normalized ranking scores (*method II*),

iii) individual-based threshold learning with normalized ranking scores (*method III*), iv) the combination of methods I and II, or v) the combination of methods I, II, and III. Resource selection of providers based on a fixed threshold selected the top-ranked providers up to a certain percentage of the total number of neighboring providers.

The number of the top-ranked merged documents regarded as "relevant" documents for each training query (*r* value) was 50. The parameter *b* in individual-based threshold learning was empirically chosen to be 3 to value precision more than recall. The parameter *n* in set-based threshold learning which determined the minimum number of "relevant" documents that a provider must have for it to be considered relevant with respect to a query was set to 5. These parameter values worked effectively for hubs that had different numbers of neighbors and covered different content areas. The settings for the other components of full-text federated search were the same as those used in previous section.

TREC queries were used for both threshold learning and evaluating the effectiveness of the learned thresholds. Each experiment using the learned thresholds for TREC queries had several runs in a way similar to leave-one-out cross validation, i.e., 99 queries were used as training queries to learn each hub's 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 WT10g queries were used for testing in the medium-sized network, TREC queries 451-550 were used as training queries. When GOV queries were used for testing in the large network, TREC queries 701-800 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 TREC queries were not used to learn the thresholds for resource selection of providers.

Figures 5.5 (a)-(d) plot the search performance for TREC queries 451-550 and WT10g queries when different thresholding methods were used for full-text resource selection of providers in the medium-sized network. Figures 5.6 (a)-(d) show the results for TREC queries 701-800 and GOV queries in the large-sized network. The results from different query sets and networks consistently show that method I, method II, and method III yielded similar precision and slightly better recall compared with the corresponding fixed thresholds with the same search efficiency. Method I (set-based threshold learning with original ranking scores) selected the most providers on average among the tested threshold learning methods, but its search performance was negatively affected by the "outlier" queries whose average term probabilities in the hub's description didn't correlate with the providers' ranking scores for them. The methods based on normalized ranking scores (method II and method III) selected fewer providers on average than the other threshold learning methods because threshold learning with normalized ranking scores tended to exaggerate the differences between ranking scores and therefore underestimate the number of relevant providers. Method III (individual-based threshold learning with normalized ranking scores) was even more conservative since its parameter was set to value precision much more than recall. The combination of methods I and II, and the combination of methods I, II, and III enabled slightly better combination of search efficiency and accuracy compared with individual threshold learning methods. Particularly, the



(a) TREC queries 451-550, precision

(b) TREC queries 451-550, recall

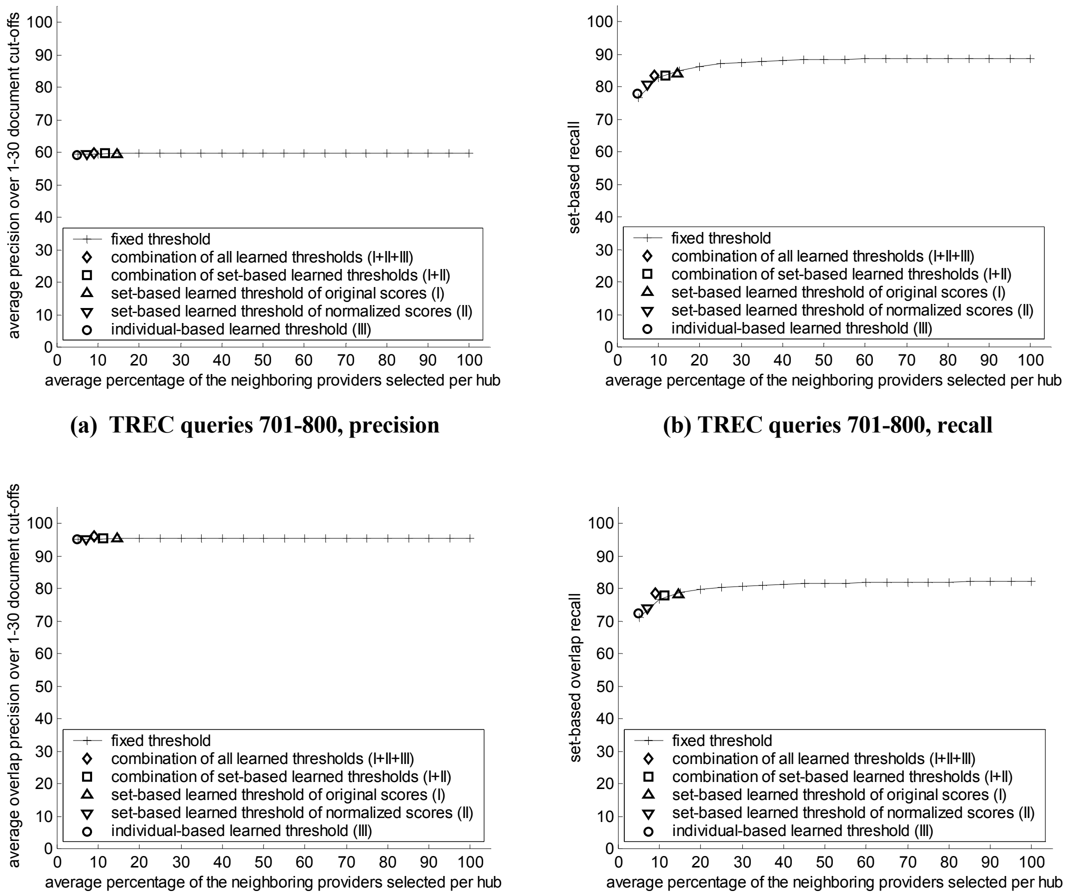
(c) WT10g queries, precision

(d) WT10g queries, recall

Figure 5.5 The search performance of different threshold learning methods for resource selection of providers in the medium-sized network.

combinations of methods I, II, and III yielded the best search performance among all the tested thresholding methods (learned or fixed) when accuracy and efficiency were considered together (with precision valued more than recall).

Table 5.8 includes the results of relative percentage change in accuracy and efficiency for different threshold learning methods, using the results of search with a fixed threshold of selecting the top 10% of ranked providers at each hub as the baseline. Overall, the results demonstrate that our threshold learning methods (particularly method II, and the combination of methods I, II, and III) were able to automatically determine thresholds to find the optimal combination of accuracy and efficiency for full-text resource selection of providers, which eliminates the need to manually choose selection thresholds.



(a) TREC queries 701-800, precision

(b) TREC queries 701-800, recall

(c) GOV queries, precision

(d) GOV queries, recall

Figure 5.6 The search performance of different threshold learning methods for resource selection of providers in the large-sized network.

5.3.3 Resource Selection of Hubs

Fixing the method of hub-provider query routing to be full-text resource selection with the threshold learned using the combination of individual-based and set-based threshold learning methods, we shift our attention to hub-hub query routing. A hub that received a query i) used the K-L divergence resource selection 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[19]

[19] Selecting two top-ranked neighboring hubs per hub would enable 15 hubs (almost 50% of all the hubs in the medium-sized network) to be reached with a search radius as small as 3. To avoid reaching a large percentage of the hubs quickly which might obscure the importance of effective resource selection, each hub only selected one top-ranked hub neighbor.

Table 5.8 Relative change in accuracy and efficiency of different threshold learning methods, compared against a fixed threshold of selecting the top 10% of ranked providers.

Queries		I	II	III	I+II	I+II+III
TREC 451-550	**Precision**	+0.20%	−2.24%	−3.47%	+0.41%	+0.82%
	Recall	+1.74%	−6.09%	−10.43%	+1.91%	+2.09%
	Efficiency	−30.52%	+28.69%	+43.08%	−7.94%	+15.59%
WT10g	**Precision**	+0.07%	−0.03%	+0.02%	+0.38%	+0.79%
	Recall	+0.02%	−0.40%	−0.65%	+0.09%	+0.21%
	Efficiency	−56.24%	+19.91%	+43.85%	−21.38%	+2.48%
TREC 701-800	**Precision**	+0.08%	−0.08%	−0.25%	+0.42%	+0.76%
	Recall	+1.20%	−2.65%	−6.02%	+0.36%	+0.36%
	Efficiency	−46.05%	+28.37%	+50.93%	−15.35%	+9.77%
GOV	**Precision**	+0.16%	−0.05%	−0.16%	+0.37%	+0.79%
	Recall	+2.22%	−3.27%	−5.23%	+1.96%	+2.61%
	Efficiency	−44.47%	+29.50%	+51.46%	−10.68%	+11.23%

that hadn't been reached for the query ("*full-text resource selection*"), ii) randomly forwarded the query to one of its neighboring hubs that hadn't been reached for the query ("*random selection*"), iii) ranked its neighboring hubs by their hub connection degrees (i.e., the number of hub neighbors) and forwarded the query to the one top-ranked neighboring hub that hadn't been reached for the query ("*degree-based selection*"), iv) ranked its neighboring hubs by the exponentially aggregated total number of documents in each one's neighborhood (Equation 4.2) and forwarded the query to the one top-ranked neighboring hub that hadn't been reached for the query ("*size-based selection*"), or v) flooded the query to all neighboring hubs ("*flooding*").

The resource descriptions of neighborhoods used by full-text resource selection were created using the procedure described in Section 4.2.3. Since the diameter of the hub-hub topology in the medium-sized P2P network is 4 and the maximum number of hops between 99% of the hub pairs in the large-sized network is 4 as well, the maximum number of iterations was chosen to be 4, which implies that each hub maintained neighborhood descriptions of radiuses from 1 to 4. Which neighborhood descriptions to use by a hub to select its hub neighbors depended on the TTL value of the query message it received. When the remaining TTL value of the query a hub received was less than 4, the neighborhood descriptions with a radius equal to the TTL value were used to conduct TTL-dependent full-text resource selection; otherwise, the neighborhood descriptions of radius 4 were used.

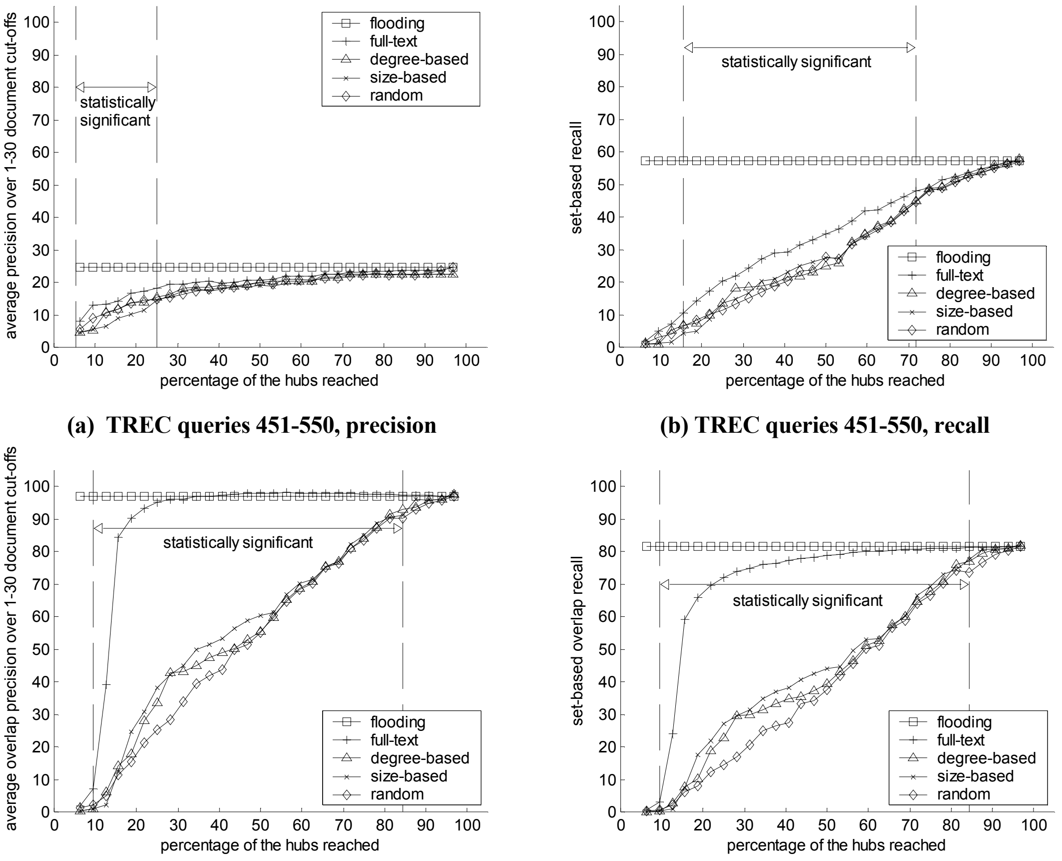
Duplicate query messages were avoided by augmenting each query message with a field for routing history and requiring each hub to check the field to avoid sending the query to those hubs indicated in the field.

For federated search using flooding, random, degree-based, or size-based selection for hub-hub query routing, the substitute corpus statistics at a hub for result merging were created by aggregating the descriptions of its neighboring providers. For federated search using full-text resource selection for hub-hub query routing, the substitute corpus statistics at a hub were generated by aggregating the descriptions of both neighboring providers and neighborhoods.

Similar to Section 5.3.1, paired two-sided sign tests were applied to the evaluation results of individual queries to test whether the difference between full-text hub selection and random, degree-based, or size-based selection in search accuracy was statistically significant at each level of search efficiency. The vertical dashed lines in the figures mark the ranges within which full-text hub selection had a statistically significant improvement at the 0.01 significance level.

Figures 5.7 (a)-(d) depict the experimental results for TREC queries 451-550 and WT10g queries using different methods of hub-hub query routing in the medium-sized network. Each point in the figures came from one experiment which ran a set of queries with a particular predetermined initial TTL value for query messages. Because each hub only selected one of its hub neighbors for hub-hub query routing when flooding was not used, the initial TTL value of each query message completely determined the percentage of the hubs that could be reached. The average (overlap) precision over 1-30 document cut-offs or set-based (overlap) recall (y-axis) are plotted against the percentage of the hubs reached by hub-hub query routing (x-axis). For both sets of queries, because the assumption of the amount of relevant contents being positively correlated with the total amount of contents reachable was not necessarily true, degree-based and size-based selection methods were barely more effective than random selection. In contrast, full-text resource selection of hubs based on neighborhood descriptions consistently outperformed random, degree-based, or size-based selection in search accuracy when they had similar search efficiency. However, the performance difference was much smaller for TREC queries than for WT10g queries due to the substantially larger percentages of the hubs covering relevant contents for TREC queries (45% on average) compared with that for WT10g queries (10% on average) in the medium-sized network. A wide distribution of relevant contents would diminish the margin of the advantage full-text resource selection had over content-independent (random, degree-based, and size-based) selection methods, which motivated us to develop the network evolution model, described in detail in the next chapter, to automatically construct network topologies with concentrated distributions of relevant contents. Despite that, full-text resource selection for hub-hub query routing still provided a much better combination of accuracy and efficiency even with random hub-hub topologies, which is demonstrated more clearly by Figures 5.8 (a)-(b).

Figures 5.8 (a)-(b) show the degradation in search accuracy for full-text, degree/size-based, and random resource selection relative to the performance of flooding (y-axis) when different amounts of query routing among the hubs were reduced compared with flooding (x-axis). From the figures we can see that even for TREC queries 451-550 with widely distributed relevant contents, the relative degradation in precision for full-text resource selection was about 20% when the amount of query routing reduced was as large as 70%. In contrast, degree/size-based or random selection



(a) TREC queries 451-550, precision

(b) TREC queries 451-550, recall

(c) WT10g queries, precision

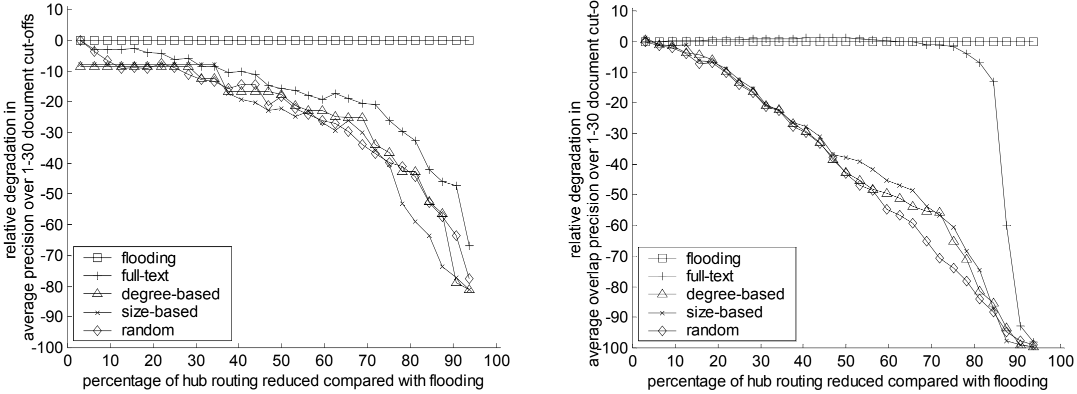
(d) WT10g queries, recall

Figure 5.7 The search performance of different methods of hub-hub query routing in the medium-sized network.

yielded almost 35%-40% relative degradations given the same amount of query routing. For WT10g queries, the amount of query routing could be reduced even further when full-text resource selection was used without significantly hurting search accuracy.

Among the tested methods of hub-hub query routing, full-text resource selection also provided the best combination of accuracy and efficiency for TREC queries 701-800 and GOV queries in the large-sized network, as illustrated by Figures 5.9 (a)-(d) and 5.10 (a)-(b). In fact, by using full-text resource selection, more than 80% of the query routing could be reduced compared with flooding when the relative degradation in precision was as small as 10%.

By comparing Figures 5.7 and 5.8 with Figures 5.9 and 5.10, we can see that a smaller amount of query routing relative to the network size was required in the large-sized network in order to keep



(a) TREC queries 451-550

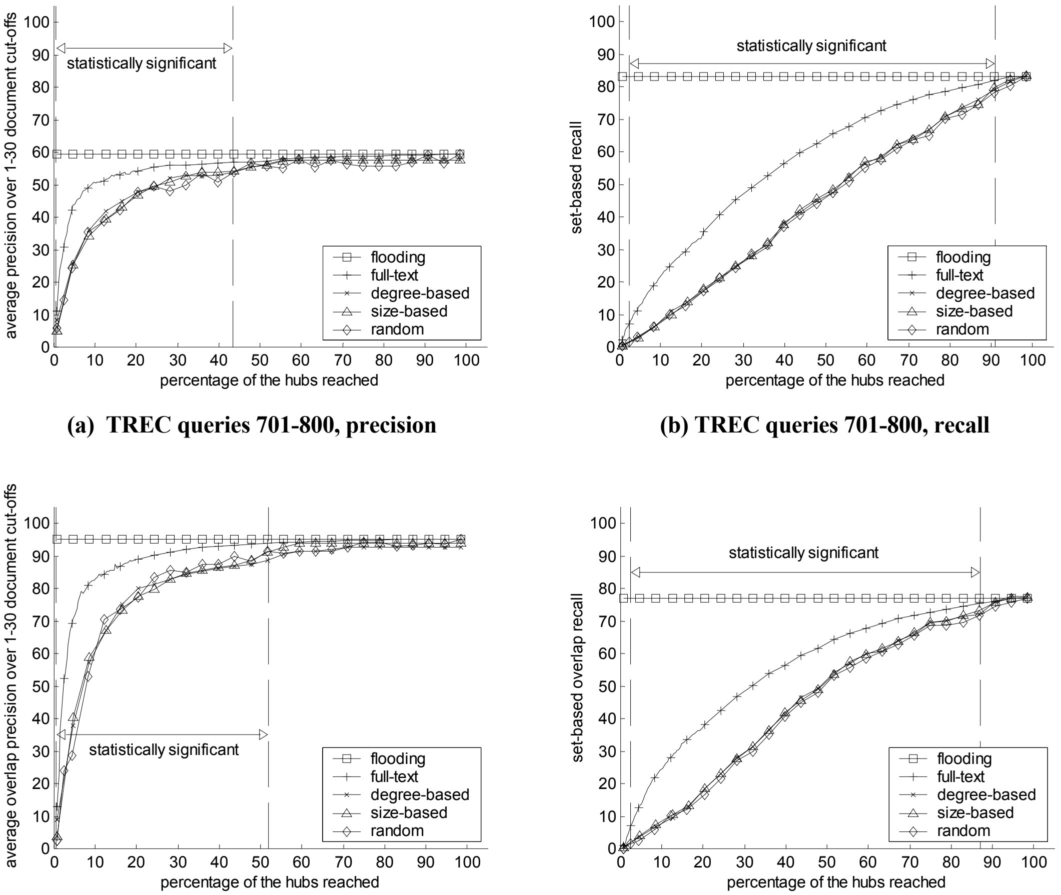
(b) WT10g queries

Figure 5.8 The relative degradation in precision of different methods of hub-hub query routing compared with flooding in the medium-sized network.

the relative degradation in accuracy below a certain level. In other words, when the network size was 8-10 times larger, the amount of query routing didn't necessarily have to be 8-10 times more so as to maintain the same level of accuracy. The sublinear relation between the amount of effective query routing and the network size gives us confidence on the scalability of full-text federated search in peer-to-peer networks.

Figures 5.8 and 5.10 show that the relative degradations in precision at top document ranks were small (< 15%) for full-text resource selection when 50% of the hubs (e.g., 16 hubs) were reached in the medium-sized network and 20% of the hubs (e.g., 50 hubs) were reached in the large network. Given these results, the experimental results presented later in this dissertation (Section 5.3.4, Section 5.3.5 and Chapter 7) are limited to the cases that up to 50% and 20% of the hubs were reached for the medium-sized network and the large network respectively, in order to focus on the performance of full-text federated search visiting only a small percentage of the network.

To summarize, similar to the evaluation on hub-provider query routing, full-text resource selection gave a better combination of search accuracy and efficiency than other more common resource selection methods for hub-hub query routing. With similar search efficiency, the improvement of full-text selection in precision over random, degree-based, or size-based selection was statistically significant when a small percentage of the hubs were reached. Its advantage in recall was statistically significant in a much wider range of settings. The experimental results also provide additional support on the effectiveness of using the automatically generated WT10g queries and the "single collection" baseline to evaluate the performance of federated search in P2P networks.



(a) TREC queries 701-800, precision

(b) TREC queries 701-800, recall

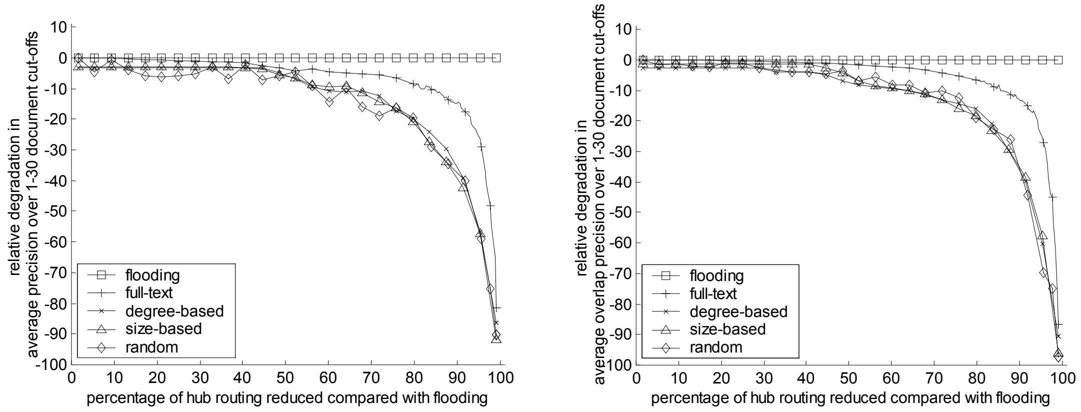
(c) GOV queries, precision

(d) GOV queries, recall

Figure 5.9 The search performance of different methods of hub-hub query routing in the large-sized network.

5.3.4 Resource Representation

This selection compares the effectiveness of different types of resource descriptions used by full-text resource selection for hub-hub query routing. On receiving a query, each hub used the K-L divergence resource selection algorithm to rank its neighboring providers and forwarded the query to the top-ranked providers based on the learned threshold. For hub-hub query routing, each hub used the K-L divergence resource selection algorithm to rank its neighboring hubs based on i) the exponentially decayed resource descriptions of neighborhoods (*"decayed neighborhoods*", Section 4.3.2), ii) the resource descriptions of its direct hub neighbors ("*direct neighbors*"), or iii) the non-decayed resource descriptions of neighborhoods ("*non-decayed neighborhoods*"), and forwarded the query to the one top-ranked neighboring hub that hadn't been reached for the query.



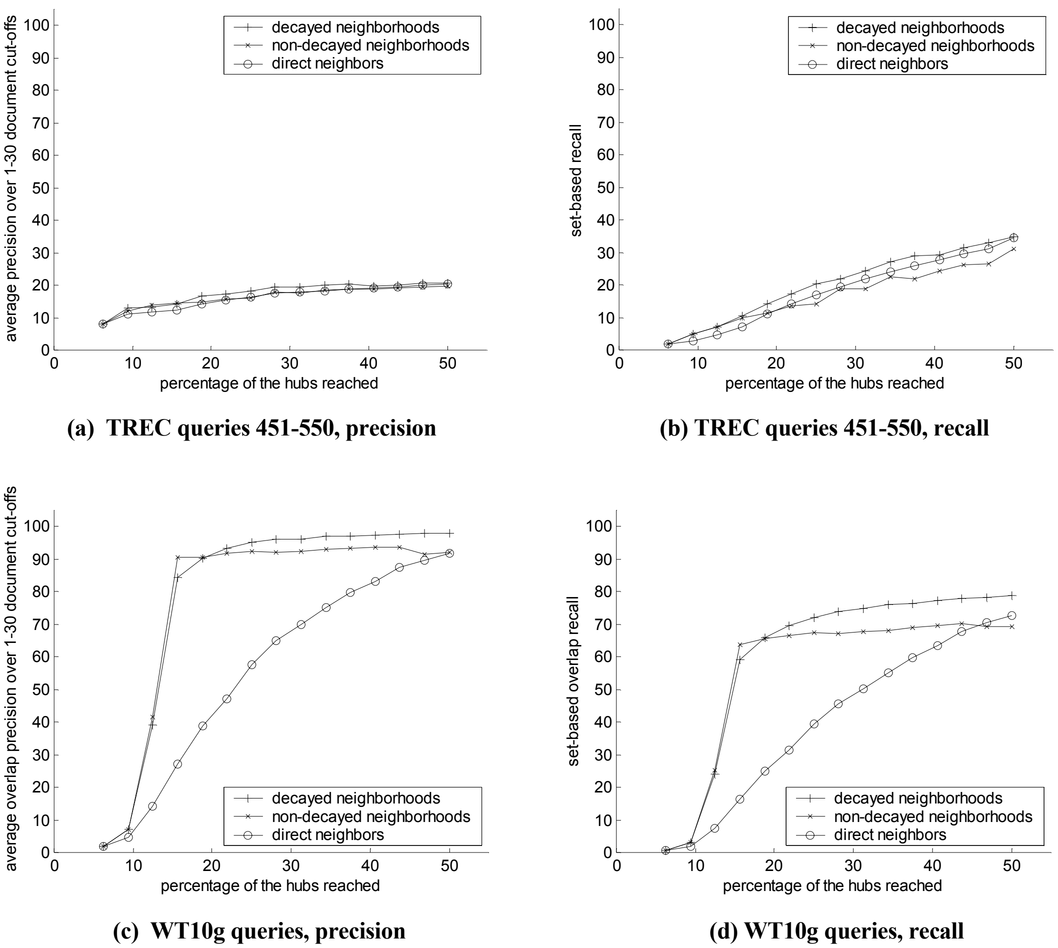
(a) TREC queries 701-800 **(b) GOV queries**

Figure 5.10 The relative degradation in precision of different methods of hub-hub query routing compared with flooding in the large-sized network.

The exponentially decayed neighborhood descriptions were created using the procedure described in Section 4.2.3, which generated neighborhood descriptions of gradually increasing radiuses in several iterations. Using the same setting as those in Section 5.3.3 for full-text hub selection, the maximum number of iterations was chosen to be 4. The same procedure was used to create the non-decayed resource descriptions for neighborhoods of different radiuses by setting the factor for exponential decay *F* in Equations 4.1-4.3 to 1. Compared with the exponentially decayed neighborhood descriptions a hub acquired, which essentially gave higher weights to contents located nearer to the hub, the non-decayed neighborhood descriptions treated contents at different distances to the hub equally. Each hub acquired the resource descriptions of its direct hub neighbors by requesting each neighboring hub to provide its own hub description (Section 4.2.2).

Full-text resource selection based on the decayed or non-decayed neighborhood descriptions was TTL-dependent with the constraint that when the remaining value of the query message's TTL exceeded 4, the neighborhood descriptions of radius 4 were used. This is consistent with the setting used in the previous section.

The experimental results for different sets of queries using different types of resource descriptions are depicted in Figures 5.11 (a)-(d) for the medium-sized P2P network when up to 50% of the hubs (i.e., 16 hubs) were reached. Table 5.9 shows the relative percentage change in search accuracy comparing resource selection using the decayed or non-decayed neighborhood descriptions against that using the descriptions of direct hub neighbors. The figures and the table show that full-text resource selection using the decayed neighborhood descriptions outperformed that using the descriptions of direct hub neighbors, particularly when the percentage of the hubs reached was small. This was expected because as already pointed out in Section 4.2.2, resource selection using the descriptions of direct hub neighbors was only based on the information of the contents located



(a) **TREC queries 451-550, precision**

(b) TREC queries 451-550, recall

(c) **WT10g queries, precision**

(d) WT10g queries, recall

Figure 5.11 The search performance of different resource descriptions for hub-hub query routing in the medium-sized network.

within one hop (providers directly connecting to the neighboring hubs), resulting in ineffective query routing when relevant contents were located multiple hops away. In contrast, the neighborhood descriptions contained information about what contents could be reached if the query traveled several hops beyond each hub neighbor until its TTL value reached zero. By looking beyond the immediate horizon, the query could be routed along the shortest path to the hubs most likely to cover relevant contents within the limits of the search radius determined by the query's TTL. In order for resource selection based on the descriptions of direct hub neighbors to have similar search accuracy compared with that based on the decayed neighborhood descriptions, a large initial query message TTL value was required to reach enough hubs so that most contents were located within one hop, yielding lower search efficiency.

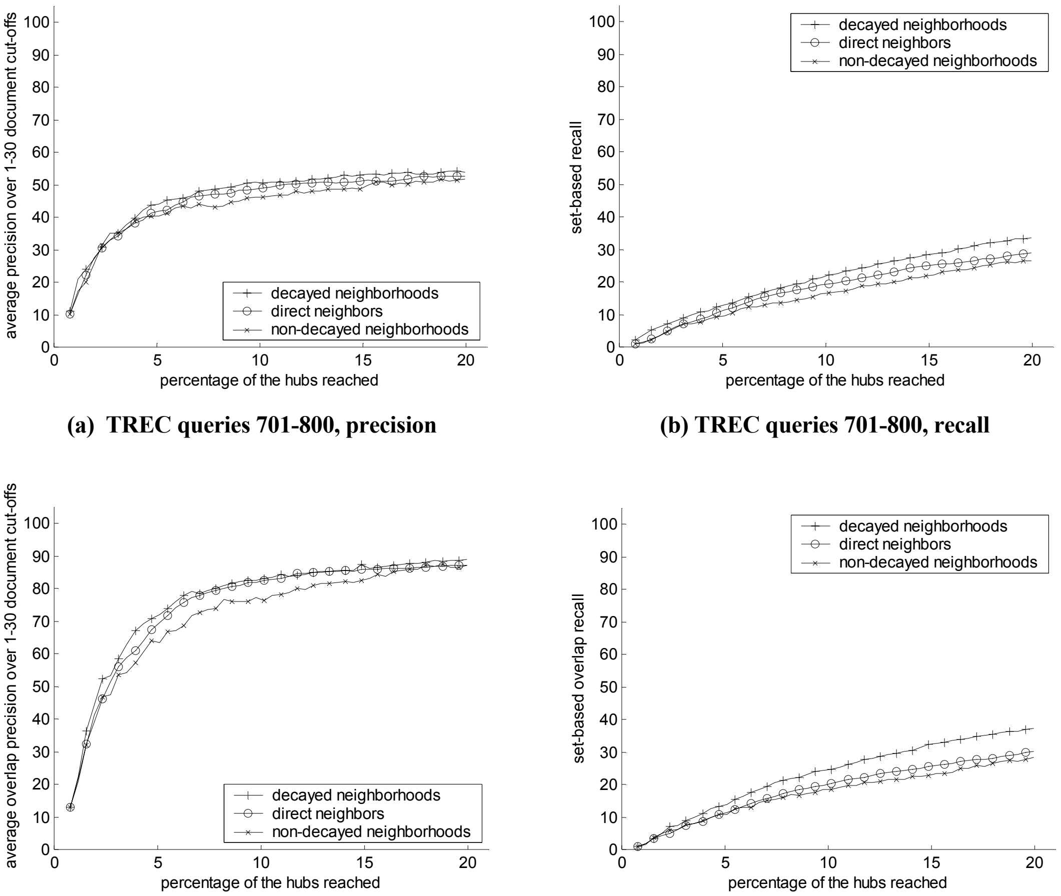
Table 5.9 Relative change in accuracy of using decayed or non-decayed neighborhood descriptions, compared against using descriptions of direct hub neighbors, for the medium-sized network.

Queries	Descrip-tions	Accuracy	10% of the hubs reached	20% of the hubs reached	30% of the hubs reached	40% of the hubs reached	50% of the hubs reached
TREC 451-550	Decayed	Precision	+16.73%	+15.89%	+8.53%	+3.84%	+1.16%
		Recall	+83.49%	+28.04%	+12.30%	+5.24%	+0.37%
	Non-decayed	Precision	+7.33%	+4.33%	−1.38%	−1.36%	−3.30%
		Recall	+84.50%	+3.61%	−13.13%	−12.61%	−10.37%
WT10g	Decayed	Precision	+53.49%	+132.88%	+37.51%	+17.01%	+7.00%
		Recall	+60.41%	+164.97%	+48.32%	+21.64%	+8.39%
	Non-decayed	Precision	+53.92%	+134.02%	+32.12%	+12.49%	+0.51%
		Recall	+63.33%	+164.55%	+34.75%	+9.66%	−4.47%

Table 5.10 Relative change in accuracy of using decayed or non-decayed neighborhood descriptions, compared against using descriptions of direct hub neighbors, for the large-sized network.

Queries	Descrip-tions	Accuracy	1% of the hubs reached	5% of the hubs reached	10% of the hubs reached	15% of the hubs reached	20% of the hubs reached
TREC 701-800	Decayed	Precision	+7.65%	+6.64%	+2.85%	+3.48%	+2.71%
		Recall	+174.57%	+13.95%	+14.08%	+12.84%	+16.17%
	Non-decayed	Precision	+1.03%	−2.38%	−6.22%	−4.71%	−2.28%
		Recall	0.00%	−11.30%	−13.71%	−12.98%	−7.56%
GOV	Decayed	Precision	0.00%	+2.89%	+0.82%	+1.91%	+1.73%
		Recall	0.00%	+23.82%	+20.45%	+25.74%	+23.55%
	Non-decayed	Precision	0.00%	−6.88%	−7.32%	−3.95%	−0.55%
		Recall	0.00%	+2.23%	−9.49%	−11.27%	−6.87%

When the initial query message TTL value was large in order to reach a larger percentage of the hubs, each hub was prompted to look farther. Because a non-decayed neighborhood description didn't distinguish between contents located at different distances to the hub that conducted resource selection, the part of the description concerning relevant contents might be overwhelmed by the part describing non-relevant contents. As a result, when relevant contents were located near the hub, looking farther than necessary might affect the effectiveness of query routing negatively, which explained the inferior performance of resource selection based on the non-decayed neighborhood descriptions when a large percentage of the hubs was reached due to a large initial TTL value for



(a) TREC queries 701-800, precision

(b) TREC queries 701-800, recall

(c) GOV queries, precision

(d) GOV queries, recall

Figure 5.12 The search performance of different resource descriptions for hub-hub query routing in the large-sized network.

query messages. In contrast, by discounting contents located farther away, even if a larger than necessary neighborhood was considered, nearby relevant contents could still stand out, giving resource selection using the exponentially decayed neighbor descriptions better performance.

Figures 5.12 (a)-(d) and Table 5.10 show the results of hub selection using different types of resource descriptions in the large-sized network when up to 20% of the hubs (i.e., 50 hubs) were reached. Because there are more relevant documents for TREC queries 701-800 and GOV queries than for TREC queries 451-550 and WT10g queries, the differences in average (overlap) precision over 1-30 document cut-offs were smaller among resource selection using different types of resource descriptions. Even so, it is clear that hub selection based on the decayed neighborhood descriptions outperformed that based on the descriptions of direct hub neighbors or the non-decayed neighborhood descriptions. The advantage of using the decayed neighborhood descriptions is demonstrated more clearly by evaluating search accuracy with set-based (overlap) recall. Resource selection using the non-decayed neighborhood descriptions resulted in even worse performance than that using the descriptions of direct hub neighbors because the size of each neighborhood was larger in the large-sized network due to a larger number of hub neighbors each hub had, which again demonstrates that looking farther ahead didn't necessarily lead to higher quality of query routing if we were not careful on choosing the right type of neighborhood descriptions.

In a few words, compared with using the descriptions of direct hub neighbors or the non-decayed neighborhood descriptions, full-text resource selection of hubs based on the exponentially decayed neighbor descriptions was more effective at selecting the hubs most likely to cover the relevant contents, and it was more robust to the variance in the predetermined search radius (initial TTL value of the query messages).

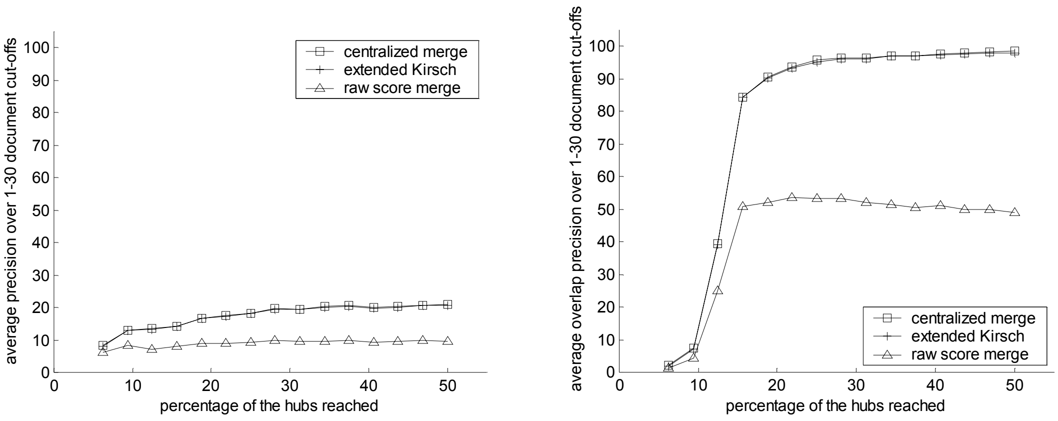
5.3.5 Result Merging

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). The top-ranked merged documents for a query (50 for the medium-sized network and 200 for the large-sized network) were returned by each hub 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 merged the documents returned from federated search by their corresponding scores returned from search in a centralized collection which was the aggregation of all the providers' contents in the network ("*centralized merge*"). The lower bound baseline method directly merged the documents from different providers using the initial document scores they provided ("*raw score merge*").

Query routing was fixed to be full-text resource selection, i.e., the K-L divergence resource selection algorithm was used by hubs to select neighboring providers (based on the descriptions of providers) and hubs (based on the decayed descriptions of neighborhoods), and each hub routed the query it received to the top-ranked neighboring providers based on the learned threshold and the one top-ranked neighboring hub.

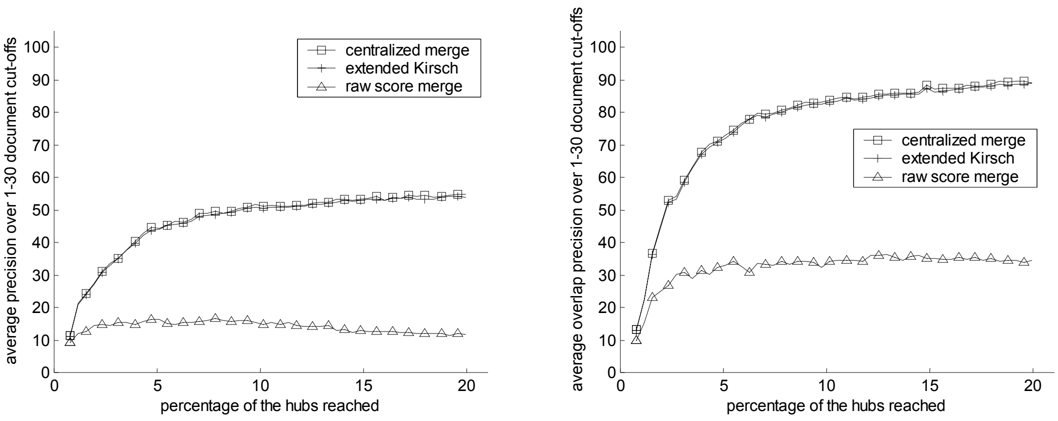
Figures 5.13 (a)-(b) show the performance of TREC queries 451-550 and WT10g queries using different result merging methods in the medium-sized hierarchical P2P network. Figures 5.14 (a)-(b) show the performance of TREC queries 701-800 and GOV queries in the large-sized network. Only the results measured in precision are shown here because different result merging methods



(a) TREC queries 451-550

(b) WT10g queries

Figure 5.13 The search performance (precision) of different result merging methods in the medium-sized network.



(a) TREC queries 701-800

(b) GOV queries

Figure 5.14 The search performance (precision) of different result merging methods in the large-sized network.

didn't affect much the values of set-based recall. The extended Kirsch's algorithm had near "optimal" performance compared with the upper bound and worked much better than the lower bound. Its performance losses in average precision relative to the upper bound were negligible. Its relative improvements in average precision over the lower bound were 109.9% on average for TREC queries and 96.5% on average for WT10g queries in the medium-sized network. The advantage of the extended Kirsch's algorithm over raw score merge was even more significant in the large-sized network. This was because the more homogeneous contents of each information provider in the large-sized network resulted in more biased corpus statistics and less globally comparable document scores, making raw score merge even less effective. In summary, with a small amount of cooperation from information providers, satisfactory performance could be obtained for result merging in the hierarchical P2P network without global corpus statistics.

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 *evaluation* of 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 testbeds of larger scales containing real documents. We make our contribution by creating two new P2P testbeds consisting of thousands of text digital libraries from the TREC WT10g and .GOV2 datasets, which are among the largest testbeds to be used so far for research on P2P systems. The sizes of our P2P networks are comparable to what might be encountered in medium- to large-sized corporate environments ("*enterprise search*"), and small- to medium-sized Web information sharing applications. In addition to providing content, our P2P testbeds also include tens of thousands of automatically generated queries and queries from a search engine query log, which have been proved by our experimental results to be useful in evaluating federated search performance of P2P networks. The large number 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 search performance over time.

Based on our P2P testbeds, we evaluate various components of our network search model against existing common alternatives: i) full-text resource selection was compared against flooding, random selection, and size/degree-based selection for query routing among hubs, and from hubs to providers, ii) the performance of resource selection of providers based on automatically learned thresholds was measured against the performance of search using a predetermined fixed threshold, iii) different types of resource descriptions were studied and compared in terms of their support for full-text resource selection of hubs, and iv) the extended Kirsch's algorithm for result merging was evaluated and compared with merging using global corpus statistics and raw score merge. The overall conclusion is that the network search model provides more sophisticated search techniques and offers a better combination of accuracy and efficiency for full-text federated search in P2P networks.

Our evaluation results using different sets of queries in different networks also indicate that the amount and distribution of relevant documents relative to the network size greatly affects the performance of federated search, particularly resource selection. When the number of relevant documents is small and relevant content is concentrated in a small part of the network, as with the case of WT10g queries in the medium-sized network, full-text resource selection has a bigger advantage than content-independent resource selection methods due to its ability to route queries towards network regions most likely to contain relevant content. When relevant documents are in abundance and scattered in the network, as with the cases of TREC queries 701-800 and GOV queries in the large network, although full-text resource selection is still superior in recall, different resource selection methods can achieve similar performance in precision at a few top document ranks since it is quite likely to hit relevant content just by luck. As the network grows to have a larger size and to contain more heterogeneous contents, the amount of relevant content for any specific query is most likely to be small relative to the network size, which makes content-independent resource selection less likely to perform well and full-text resource selection more likely to shine.

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

In this chapter, 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.

6.1 Hub-Provider Topology

As defined in Section 3.2.1, a hub-provider topology with content-based locality is 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 controlled-vocabulary representations based on a global classification hierarchy or ontology. This approach to deciding and representing the content area covered by 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, or which may not distribute contents evenly across the network. Another approach to describing a hub's content area is to use its full-text resource description obtained by aggregating the resource descriptions of its neighboring providers, which is proposed in our network search model (Section 4.2.2). 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 Sections 6.1.1-6.1.4, we present the design of our hub-provider topology evolution algorithm in consideration of the unique characteristics of hierarchical P2P networks and heterogeneous, open-domain contents, followed by the detailed description of the algorithm in Section 6.1.5.

6.1.1 Decentralization and Role Differentiation

When a provider requests to join a hierarchical P2P network, since there is no centralized server to decide which cluster(s) it should join, decisions must be made in a decentralized manner about which hubs the provider can connect to so as to maintain content-based locality in hub-provider topology. Because in a hierarchical P2P network, hubs typically have more processing power and connection bandwidth, they play a more active role in the evolution of hub-provider topology so that providers with limited resources can minimize the computation and communication costs associated with topology evolution.

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 the final decision about which hub(s) to connect to. Each hub that receives the provider's resource description is responsible for calculating the degree of match between the provider's content and the hub's content-based cluster, providing this information to the provider to facilitate its decision making, and propagating the provider's resource description to other hubs.

6.1.2 Dynamic 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.

In an ideal situation, each hub in the network is responsible for covering a specific content area and uses a similarity threshold to decide whether to accept a provider into its cluster and integrate the provider's content into its content area. When a provider's content is sufficiently dissimilar to all existing content areas, an unoccupied hub is contacted to create a new content-based cluster to accommodate this provider. This approach may work well when appropriate threshold values are chosen to control the granularity of the content area covered by each hub so that the number of content areas matches the number of hubs in the network. However, in real, operational environments, because the number of content areas cannot be predetermined for open-domain contents, and the number of hubs is often limited, it is difficult to choose similarity threshold values in advance to satisfy the above condition, particularly when there is no central coordination and control. On the one hand, tighter similarity thresholds result in more cohesive and homogenous clusters representing narrower content areas so that more hubs than those available may be needed to cover the contents in the network. On the other hand, looser threshold values decrease the homogeneity of content-based clusters and thus reduce the degree of content-based locality. To solve this problem, instead of solely relying on a single similarity threshold to determine the granularity of a hub's content area, we use a more flexible clustering strategy which cultivates multiple sub-clusters within each hub's content-based cluster and spins off a sub-cluster to create a new content area when and only when the sub-cluster has grown to a certain size.[20] With this strategy, while the granularity of each sub-cluster may be relatively fixed by using a similarity threshold, the granularity of each hub's content area can be dynamically adjusted by increasing its number of sub-clusters or transferring its sub-clusters to other hubs. Although such dynamic adaptive clustering may decrease the homogeneity of some content-based clusters, the potentially small reduction in content-based locality is offset by its ability to adjust autonomously based on the actual network conditions.

6.1.3 Load Balancing

Because the amount of contents available in the network is often biased for different topics, hubs that cover popular content areas may be overwhelmed by the hub-provider connections they need to maintain and the query load they have to handle. To avoid overloading some hubs with popular contents and information requests while wasting other hubs' resources on marginal or unpopular contents, hubs need to average their responsibilities in order to achieve load balance in the network. Hubs' responsibility of serving popular contents can be balanced by transferring sub-clusters of heavily connected hubs to lightly connected ones. The degree of content-based locality is expected

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

to be affected only slightly if hubs take advantage of the content-based small-world properties of hub-hub topology to locate appropriate recipient hubs based on the content areas they cover. By using distributed cultivation of sub-clusters, hubs can also share the responsibility of serving unpopular contents instead of dedicating a single hub to handle them.

6.1.4 Selective Propagation of Providers' Contents

Because constructing and adjusting hub-provider topology requires additional computation and communication costs, it is cost-efficient to propagate the resource description of each joining 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. Selective propagation 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 become a problem when the number of providers joining the network becomes large.

Selective propagation of a provider's resource description can be conducted in a way similar to hub-hub query routing in full-text federated search. Hubs need to know about the contents covered in their neighborhoods so as to decide where to propagate the provider's resource description. Since hubs need to collect neighborhood content information anyway for hub-hub query routing, selective propagation of providers' resource descriptions adds few additional costs.

6.1.5 Algorithm

An information provider's content is described using its full-text resource description. A sub-cluster is represented by the aggregation of the resource descriptions of its provider members. A hub's content area is represented by its resource description, generated by aggregating the representations of its sub-clusters. We use two similarity thresholds to distinguish among three levels of similarity between a provider's content and the contents covered by a sub-cluster: *high*, *marginal*, and *low*.[21] A provider's first priority is to join the most similar sub-cluster among all the sub-clusters to which it has a *high* similarity level. If it fails to find sub-clusters with *high* similarity, but has at least one sub-cluster with *marginal* similarity, it will join the content-based cluster of the hub that has the most similar sub-cluster with *marginal* similarity by initiating a new sub-cluster at this hub. The distinction between *high* and *marginal* similarity values is for controlling the granularity of each sub-cluster while allowing the granularity of a hub's content-based cluster to dynamically change by including more sub-clusters. If the provider has a *low*

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

similarity level to all existing sub-clusters in the network, it requests an empty[22] hub to initiate a new sub-cluster within a new content-based cluster. When all the hubs are non-empty, the provider initiates a new sub-cluster within the content-based cluster of the hub that has the most similar sub-cluster.

The join process of an information provider *P* proceeds as follows. If it was active in the network before, it first tries to connect to the hubs it previously connected to. If it fails or if it is completely new to the network, it finds an initial set of hubs by querying host-cache servers or by pinging the network. Then it arbitrarily chooses a hub from the list to connect to temporarily, which is responsible for acquiring *P*'s resource description and starting its propagation among the hubs within a certain radius specified by the TTL of the message. Each non-empty hub that receives *P*'s resource description executes two operations. First, the hub computes the similarity between *P*'s resource description and the description of each of its sub-clusters, and sends back to *P* a message containing the value of the highest similarity along with the similarity level based on its thresholds. Second, the hub selects some neighboring hubs to propagate *P*'s resource description based on the similarity between *P*'s resource description and the resource description of each of its neighborhoods. Each empty hub that receives *P*'s resource description directly forwards it to its hub neighbors. *P* collects the messages from the hubs and chooses a hub to connect to based on the strategies described in the previous paragraph. Figure 6.1 illustrates primary activities of the join process. The solid lines indicate the propagation of *P*'s resource description, and the dashed lines represent the messages that the hubs send back to *P* about the similarity values and levels.

When an information provider *P* contains documents on multiple topics, it may be more appropriate for *P* to join multiple content-based clusters instead of a single one. The strategies described earlier can be easily extended to support "soft clustering". For example, if *P* discovers multiple sub-clusters with high or marginal similarity levels, it can rank these sub-clusters based on its content similarity to them, and join multiple top-ranked sub-clusters of high similarity, or initiate new sub-clusters in the content-based clusters that contain the top-ranked sub-clusters of marginal similarity. However, for simplicity, this dissertation considers only the hard-clustering scenario in which *P* joins only the best single hub.

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

[22] A hub is "empty" if it has an empty content-based cluster, i.e., if it doesn't connect to any information provider. Otherwise, it is non-empty.

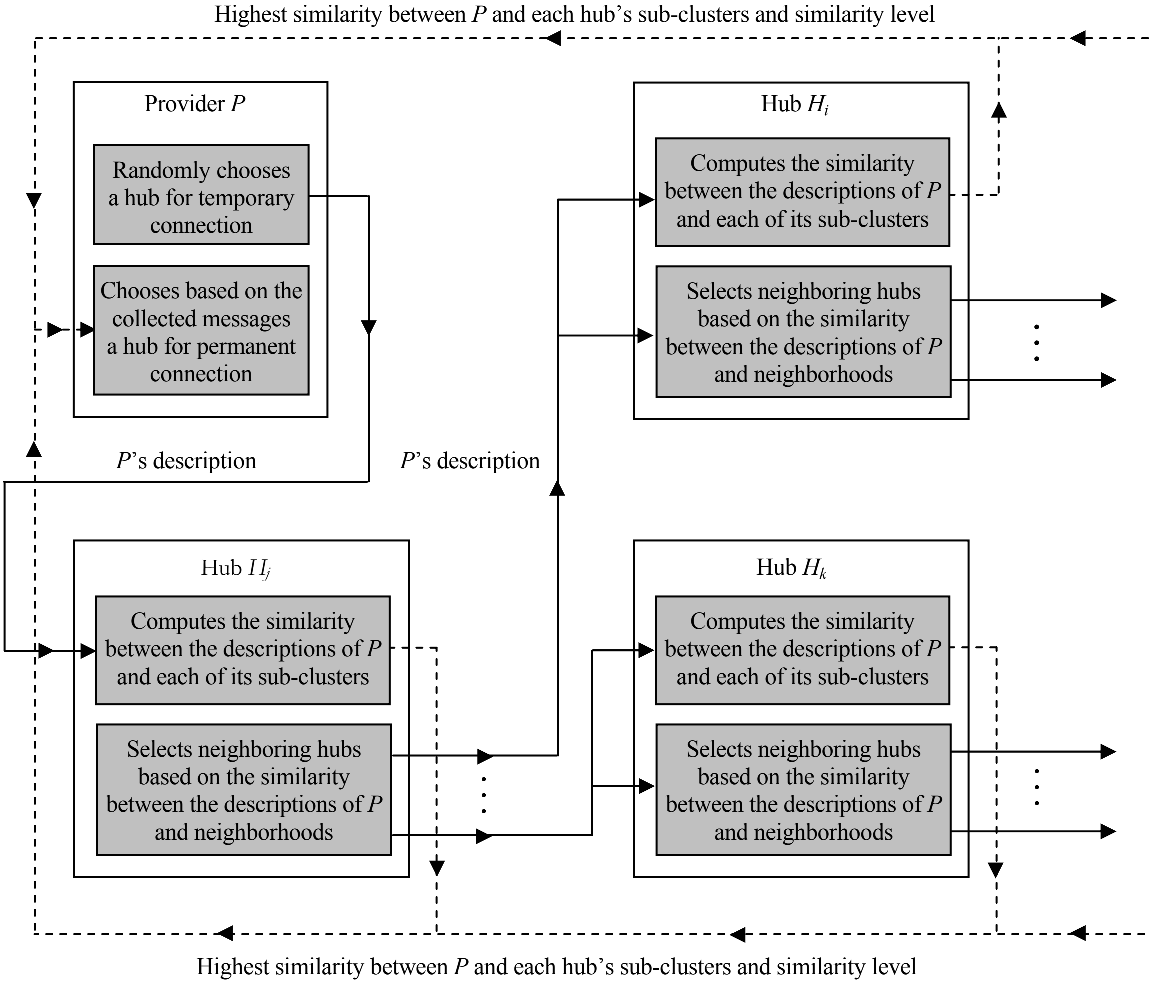

Highest similarity between P and each hub's sub-clusters and similarity level
Provider P
Randomly chooses a hub for temporary connection
Chooses based on the collected messages a hub for permanent connection
Hub H_i
Computes the similarity between the descriptions of P and each of its sub-clusters
Selects neighboring hubs based on the similarity between the descriptions of P and neighborhoods
P's description
P's description
Hub H_j
Computes the similarity between the descriptions of P and each of its sub-clusters
Selects neighboring hubs based on the similarity between the descriptions of P and neighborhoods
Hub H_k
Computes the similarity between the descriptions of P and each of its sub-clusters
Selects neighboring hubs based on the similarity between the descriptions of P and neighborhoods
Highest similarity between P and each hub's sub-clusters and similarity level

Figure 6.1 Illustration of the join process of an information provider *P*

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

To balance query load and to reduce the path length for effective query routing, sometimes it may be beneficial for an information provider (especially if it contains popular content) to connect to hubs with more computing power and network connections even if the match between their content areas and the provider's content is suboptimal. One could extend our algorithm to support this type of topology evolution by allowing hub to attach additional information about their resources and workloads to the messages returned to the joining provider. Content-based and popularity-based information could be used by the provider separately to determine different sets of connections, or in combination for one set of connections. For the former approach, the provider might choose content-based and popularity-based connections separately, and each hub might maintain both a content-based cluster and a popularity-based cluster, and apply different search strategies to them. For the latter approach, the provider might use a utility function to combine different factors and choose connections that optimize the utility, and a utility-based criterion might replace the content-based criterion used for resource selection at each hub. This extension is not explored in this dissertation; it is mentioned here to indicate that the hub-provider topology evolution algorithm can be extended to include a wider range of information than just content-based similarity.

6.2 Hub-Hub Topology

As defined in Section 3.2.2, 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 2000) (Kleinberg 2001). Sections 6.2.1-6.2.4 describe these features; Section 6.2.5 describe in detail the evolution algorithm.

6.2.1 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 to and disconnect from it, and new content areas may emerge in the network. Hence each hub needs to adjust its connections to other hubs whenever necessary to accommodate these changes and maintain content-based small-world properties. A hub's connection adjustment procedure will be invoked when the degree of its content or connection change exceeds a certain threshold. To determine when connection adjustment is necessary at a hub due to changes in other hubs, one simple mechanism is to require each hub to issue a short message setting a flag when its content area or connections change dramatically, which is propagated to the hubs within a certain radius. Each hub collects such messages from other hubs to monitor the dynamism of the network, based on which it decides when to adjust its hub connections.

6.2.2 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. By iteratively adjusting its connections when the difference between its maximum and minimum content distances to neighboring hubs is below a threshold, a hub is able to connect to "globally" close and remote hubs because connection adjustment keeps bringing new hubs to its local network region. Experimental results indicate that the number of iterations required is only a small constant larger than the resulting (small) diameter of the hub-hub topology, which means that the hub-hub topology can converge to one with small-world properties fairly quickly.

6.2.3 Degree Balancing

Because connections at the hub level are major channels for query routing and propagation 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 connection degrees without undermining content-based small-world properties. Experimental results demonstrate that the resulting hub-hub topology can avoid query routing hotspots, and the removal of any hub does not dramatically increase the path lengths between the remaining hubs.

6.2.4 Algorithm

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

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

1. *H* uses a threshold ρ* to distinguish between similar and dissimilar hubs among the hubs within two hops from it;

2. *H* connects to M_{ol} most similar hubs whose incoming hub connection capacities have not reached their maximally allowed values M_i, where M_{ol} is the maximum number of outgoing local hub connections *H* can have;

3. If all of *H*'s similar hubs have reached their maximum incoming hub connection capacities M_i, *H* requests them to recommend their similar hub neighbors, which may be repeated recursively until *H* establishes at least one local hub connection or the number of requests reaches a limit before it succeeds in finding any similar hub available for connection;

4. *H* selects M_{og} dissimilar hubs that have not reached their maximum incoming hub connection capacities M_i with probability:

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

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

By dynamically adapting each hub's outgoing[23] connections at the hub level, hub-hub topology effectively maintains content-based small-world properties. Since each hub adjusts its connections only based on its local knowledge of the hubs that are located within two hops from it and possibly their recommended local contacts, no global information or control are necessary for the evolution of hub-hub topology. The step of limiting each hub's connection capacity and recommending other similar hubs when its own connection capacity becomes full helps in distributing connections at the hub level in a less skewed manner to avoid concentrating a large number of connections at a few hubs. 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

[23] 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.

scales" ("similarity scales"), which allows hubs to route any query efficiently towards its targeted content area (Kleinberg 2000).

6.3 Hub-Consumer Topology

As mentioned in Section 3.2.3, a consumer *C* may perform *characteristic search* for which information requests are closely related to the user's persistent interests in specific topics, and *uncharacteristic search* for which information requests are ad-hoc, transient in nature. While a consumer cannot do much with regard to hub-consumer topology to optimize the performance of uncharacteristic search, for the benefit of characteristic search, *C* should establish permanent connections to hubs that cover content areas most similar to its interests in order to take advantage of interest-based locality. In addition, if *C* is interested in several different topics, then the optimal set of hubs it should connect to may vary by topic. Based on the dynamic observation of initial hub selection conducted by the consumer for characteristic queries (Section 4.4), the consumer can establish permanent connections to those hubs that are most frequently selected, and periodically adjust its connections to adapt to the change in user's interests and hubs' contents.

6.4 Summary

In this chapter, we describe a network evolution model to dynamically and autonomously 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) 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 content partitions (e.g., based on an ontology or a classification hierarchy) 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 and doesn't scale well. Both approaches would not work well with open-domain full-text representations of contents 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 to avoid partitioning the content space ahead of time. In addition, most work is assigned to hubs to fully utilize their high connection bandwidth and processing power. With selective propagation of providers' content information at the hub level and the mechanism designed to achieve better load balance, our hub-provider topology evolution algorithm also enables higher efficiency and scalability.

To provide an adaptive, cost-efficient solution to constructing 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 long-range connections should be based on a power-law instead of uniform distribution. 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 2000) (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. Techniques developed by (Renda and Callan 2004) to handle hub failures can be applied to further improve the robustness of the network.

Although some previous research recognizes the existence of characteristic search (i.e., a consumer's information requests expressing his persistent, long-term information needs) in P2P networks and designs topology evolution and/or search algorithms to improve its performance (Ramanathan et al. 2002) (Sripanidkulchai et al. 2003) (Shao and Wang 2004), the distinction between characteristic and uncharacteristic search, and the distinction between characteristic search for different topics of interest have not been studied for federated search in P2P networks. In contrast, based on the dynamically learned user model at each consumer which uses query clusters to represent a user's different interests, our hub-consumer topology evolution is able to make query-specific adjustments to connection topology so that each user's search pattern can be learned and taken advantage of to achieve optimal search performance.

Chapter 7

EVALUATION OF NETWORK EVOLUTION MODEL

This chapter evaluates the effectiveness of our network evolution model in building a hierarchical P2P network topology with desired search-enhancing properties. The same datasets used for evaluating the network search model (Section 5.1) were adopted to evaluate the network evolution model. Because previous work has provided extensive experimental results and detailed analysis on the robustness of full-text federated search in hierarchical P2P networks with peer departures (Renda and Callan 2004), the evaluation of the network evolution model focuses primarily on the join process by assuming that all the hubs remain active in the course of topology evolution and no providers depart the network after joining it. In order to be able to directly compare the performance of full-text federated search in different network topologies, full-text federated search was conducted at the end of the network evolution with all the information providers' contents available. The evolution algorithms for different components of the network topology (hub-provider, hub-hub, and hub-consumer) were applied and evaluated one at a time by taking the same progressive evaluation approach used in Chapter 5. The same performance measures described in Section 5.2 were used. As discussed in Section 5.3.3, to focus on the performance of federated search by visiting a small percentage of the peers, the results shown in this chapter are limited to up to 50% of the hubs reached in the medium-sized network and up to 20% of the hubs reached in the large network.

7.1 Hub-Provider Topology

This section studies the effectiveness of the hub-provider topology evolution algorithm described in Section 6.1 in the following two aspects: i) whether the dynamically constructed hub-provider topology exhibited desired properties such as content-based locality and load balance, and ii) whether it was effective in enhancing search performance compared with a randomly generated hub-provider topology. Experimental settings were described in the next section, followed by experimental results and analysis in Section 7.1.2.

7.1.1 Experimental Settings

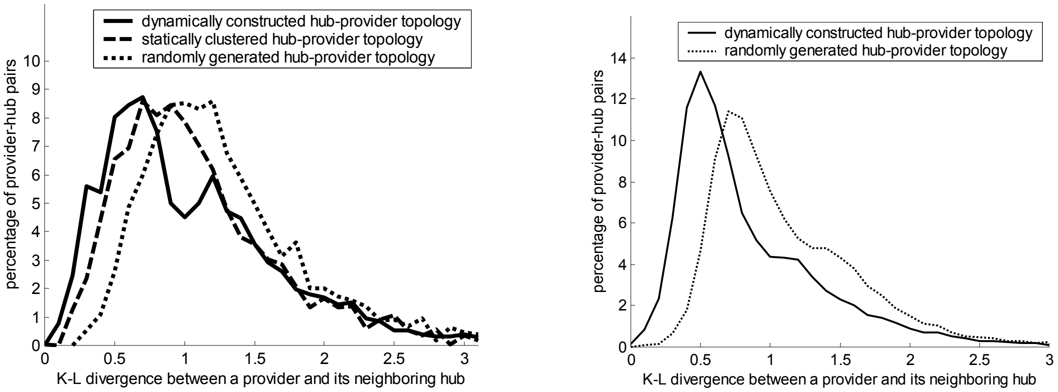
The hierarchical P2P network to be evaluated was initialized to be a network of empty hubs (32 for the medium-sized network and 256 for the large-sized network) randomly connecting with one another without any providers. To simplify the evolution process, information providers were assumed to join the network one at a time. The connections between hubs remained static in order to focus on the evolution of hub-provider topology.

The hub-provider topology evolution algorithm uses a dynamic, adaptive clustering strategy which cultivates multiple sub-clusters of information providers within each hub's content-based cluster in order to dynamically adjust the granularity of the content area covered by each hub. Two thresholds are needed to categorize the similarity between a provider's content and the contents covered by a sub-cluster (measured by the negative of the K-L divergence between their full-text resource descriptions). The *high-marginal threshold* is used to distinguish between high and marginal similarity values in order to determine whether to accept the provider into an existing sub-cluster, or to accommodate the provider by initializing a new sub-cluster within an existing content-based cluster. The *marginal-low threshold* is used to distinguish between marginal and low similarity values so that a provider with a low similarity to all existing sub-clusters can initiate a new content-based cluster when there is an empty hub available. In our experiments, the high-marginal threshold was set to -1.0 and the marginal-low threshold was -2.0.

To avoid propagating a provider's resource description in the network indefinitely, each message carrying a provider's resource description has a finite TTL (Time-To-Live) value so that the provider's content is only propagated within a certain radius from the initial hub. The TTL was set to 4 to be consistent with the setting used in Chapter 5. The threshold used by a hub to determine which neighboring hubs to select for further propagation of a provider's description (based on the K-L divergence between the provider's description and neighborhood descriptions) was 1.5.

In our algorithm, when there is an empty hub available, a sub-cluster exceeding a certain size limit can be spun off to create a new content-based cluster in representing a new emerging content area. 5 was used in the experiments for the spin-off size limit of a sub-cluster. The size of the content-based cluster at each hub also has a limit for the purpose of load balance. A hub stops accepting new providers once the limit has been reached. Its value was set to 275.

Full-text federated search in the dynamically constructed or the randomly generated hub-provider topology was based on the settings consistent with those used in Chapter 5, i.e., each query was issued with increasing initial TTL values by a consumer connected to a hub located farthest on average from relevant content, each hub that received the query message forwarded it with a decreased TTL value to the top-ranked neighboring providers selected based on provider descriptions and the learned threshold, and the top one neighboring hub selected based on exponentially decayed neighborhood descriptions until the TTL value reached zero, each provider returned up to 50 top-ranked documents, and hubs used the extended Kirsch's algorithm for result merging and returned the top-ranked documents (50 for the medium-sized network and 200 for the large-sized network) to the consumer. The consumer used the raw score merge to merge the results returned by multiple hubs.



(a) medium-sized network

(b) large-sized network

Figure 7.1 The within-cluster divergence distributions of different hub-provider topologies in the networks of different sizes.

7.1.2 Experimental Results

The degree of content-based locality in the dynamically constructed hub-provider topology was first measured by the cohesion of each hub's content-based cluster formed by its neighboring providers. The K-L divergence between each information provider and its connecting hub was calculated and the distribution of the calculated divergence values for all connecting provider-hub pairs is shown in Figure 7.1 (a) for the medium-sized network. As a comparison, Figure 7.1 (a) also includes the within-cluster divergence distributions of a randomly generated hub-provider topology and a hub-provider topology created by statically clustering all 2,500 information providers into 32 clusters (equal to the number of hubs in the network) using the K-means clustering algorithm. Figure 7.1 (b) plots the within-cluster divergence distributions of the dynamically constructed and the randomly generated hub-provider topologies in the large-sized P2P network. Due to the computational costs of clustering 25,000 providers with large-sized representations, the statically clustered hub-provider topology was not generated for the large-sized network. Both figures show that the dynamically constructed hub-provider topology had smaller mean in the distribution of the within-cluster divergence than the randomly generated hub-provider topology, indicating a higher degree of cohesion for content-based clusters than random clusters. The humps between the K-L divergence values of 1.0 and 1.5 and the long tails for the distribution curves of the dynamically constructed hub-provider topologies were due, respectively, to the existence of provider members with marginally similar or remotely related contents in each cluster, resulting from the diverse nature of the contents and the hubs' sharing responsibility for serving providers with unpopular contents. Figure 7.1 (a) also shows that content-based clusters constructed using the topology evolution algorithm and those created using the K-means clustering algorithm had similar degree of cohesion, demonstrating that our incremental, adaptive clustering approach was able to generate clusters with a satisfactory degree of content-based locality.

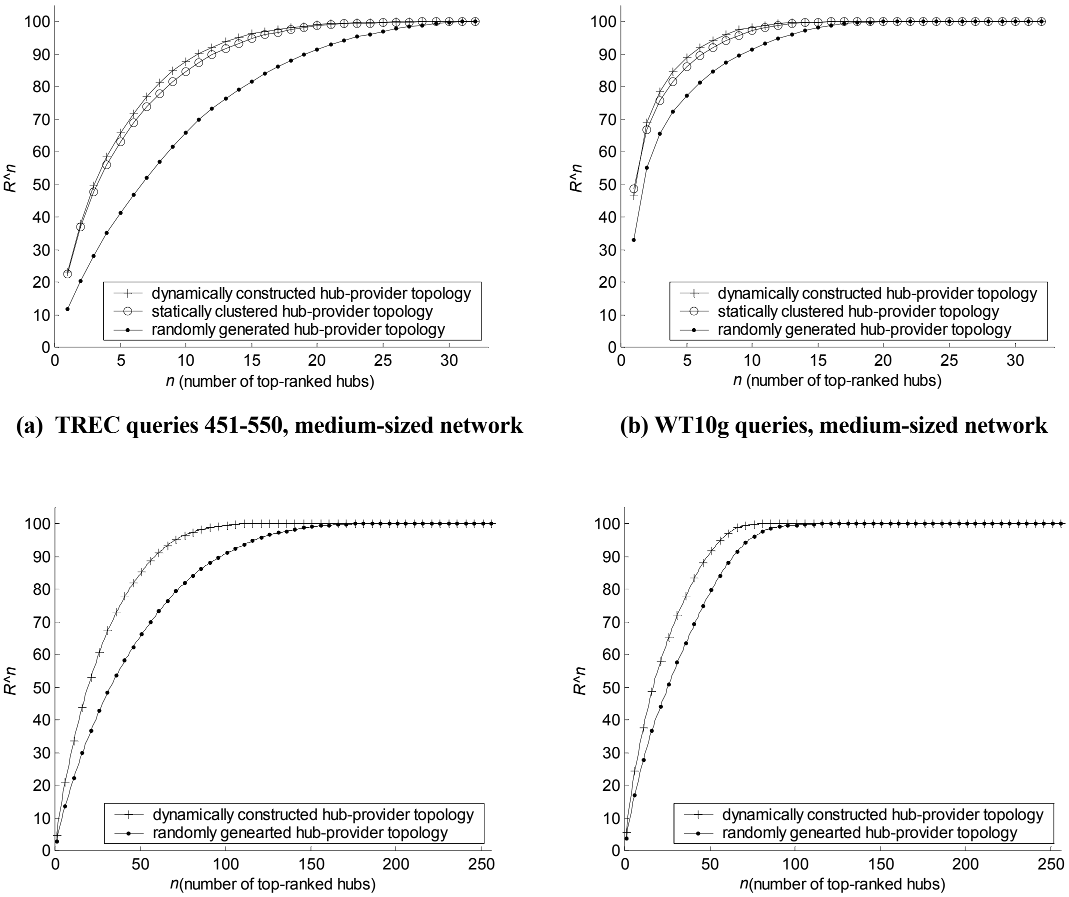
If a hierarchical P2P network exhibits good content-based locality, then most contents relevant to an information request are expected to be concentrated in a small part of the network so that only a small percentage of the hubs need to be contacted in order to obtain sufficient relevant documents. Therefore, the degree of content-based locality can also be measured by the degree of concentration of information providers with relevant contents ("*relevant providers*") among different hubs. Given a set of queries with (real or pseudo) relevant judgments, for each hub-provider topology, we ranked the hubs by the number of relevant providers[24] contained in their clusters, and defined $\hat{R}_n$ to be the percentage of the relevant providers that had been accumulated via the *n* top-ranked hubs. The values of $\hat{R}_n$ for different queries were averaged. A similar metric has been used to evaluate the effectiveness of resource selection in traditional distributed information retrieval (French et al. 1998).

Figures 7.2 (a)-(d) show the results for different hub-provider topologies using different sets of queries in two network sizes. From the figures we can see that the degree of relevant content concentration in the network with a dynamically constructed hub-provider topology was consistently higher than that in the network with a random hub-provider topology. The dynamically constructed and the statically clustered hub-provider topologies yielded similar relevant content concentration, again illustrating that the dynamic topology evolution algorithm was able to generate a hub-provider topology with the desired content-based locality.

The slightly worse performance of the statically clustered topology in Figure 7.1(a) and Figures 7.2 (a)-(b) was due more to the random choice of initial cluster centroids for the K-means algorithm, and the network's heterogeneous contents with varied qualities, than to its actual inferiority to the dynamically generated hub-provider topology. The difference is not significant nor would it necessarily be expected to repeat in an experiment with a slightly different configuration. Therefore, dynamic clustering with our local algorithm and static clustering with a global algorithm are expected to yield similar degree of cluster cohesion and relevant content concentration.

An indicator of whether a hub-provider topology has balanced load is the distribution of the sizes of clusters formed by hub-provider connections. Figures 7.3 (a)-(b) compare the cluster size distributions of the dynamically constructed and the statically clustered hub-provider topologies in the medium-sized network. Most content-based clusters had less than 100 provider members as shown in both figures. However, compared with the distribution plotted in Figure 7.3 (a) for the statically clustered hub-provider topology, the distribution in Figure 7.3 (b) for the dynamically

[24] Ranking the hubs by the number of relevant documents contained in their clusters yielded similar figures.



(a) TREC queries 451-550, medium-sized network

(b) WT10g queries, medium-sized network

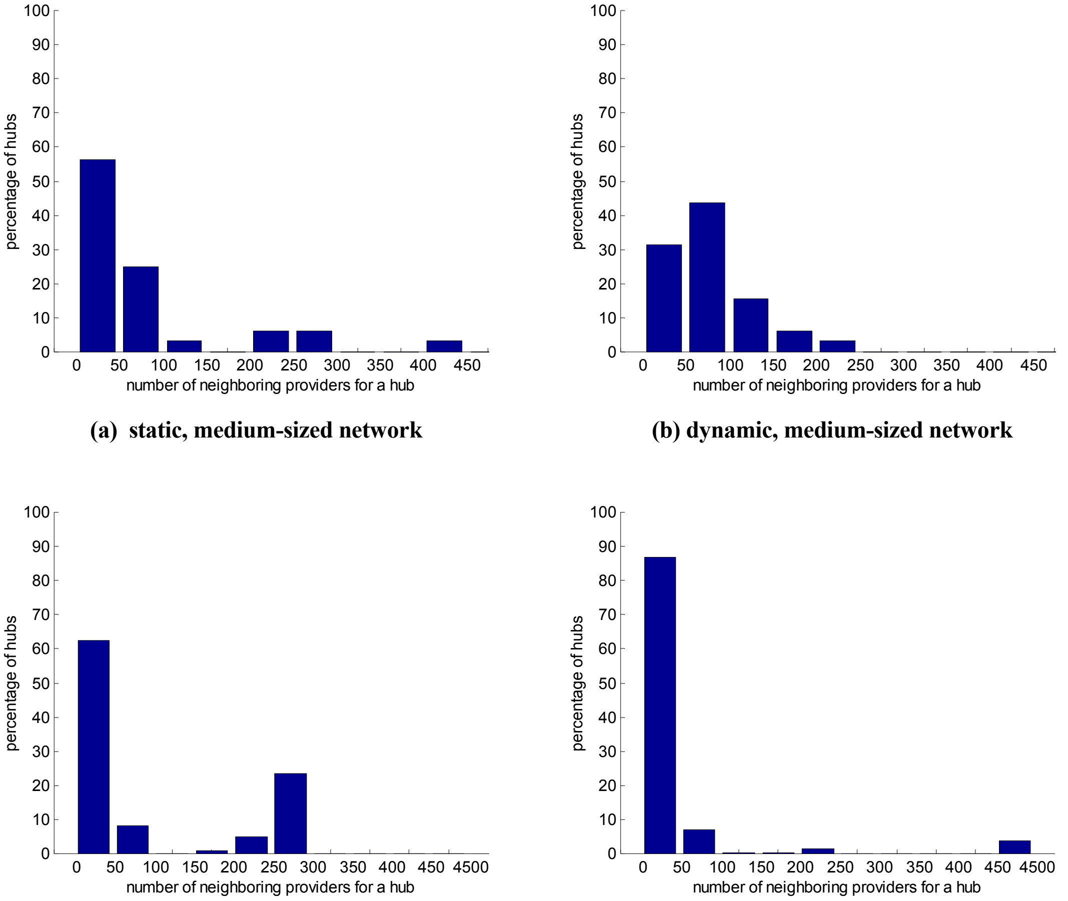
(c) TREC queries 701-800, large-sized network

(d) GOV queries, large-sized network

Figure 7.2 The cumulative distributions of the relevant providers among hubs for different hub-provider topologies in the networks of different sizes.

constructed hub-provider topology was less skewed, as evident by its smaller percentage values of small and large clusters, and larger percentage values of medium-sized clusters.

Figure 7.3 (c) depicts the cluster size distribution of the hub-provider topology in the large-sized network constructed using the dynamic topology evolution algorithm. As a comparison, we applied to the large-sized network the same dynamic hub-provider topology evolution algorithm *without* limiting cluster sizes or transferring sub-clusters from heavily connected hubs to lightly connected ones, and the resulting cluster size distribution is shown in Figure 7.3 (d). The differences between Figure 7.3 (c) and Figure 7.3 (d) indicate that without the steps of balancing the load, dynamic topology evolution based on distributed, adaptive clustering criteria may overload a few hubs with a large number of information providers sharing popular contents while wasting the resources of many other hubs on serving a small number of providers with less-than-popular contents. Load



(a) static, medium-sized network

(b) dynamic, medium-sized network

(c) dynamic w/ balance, large-sized network

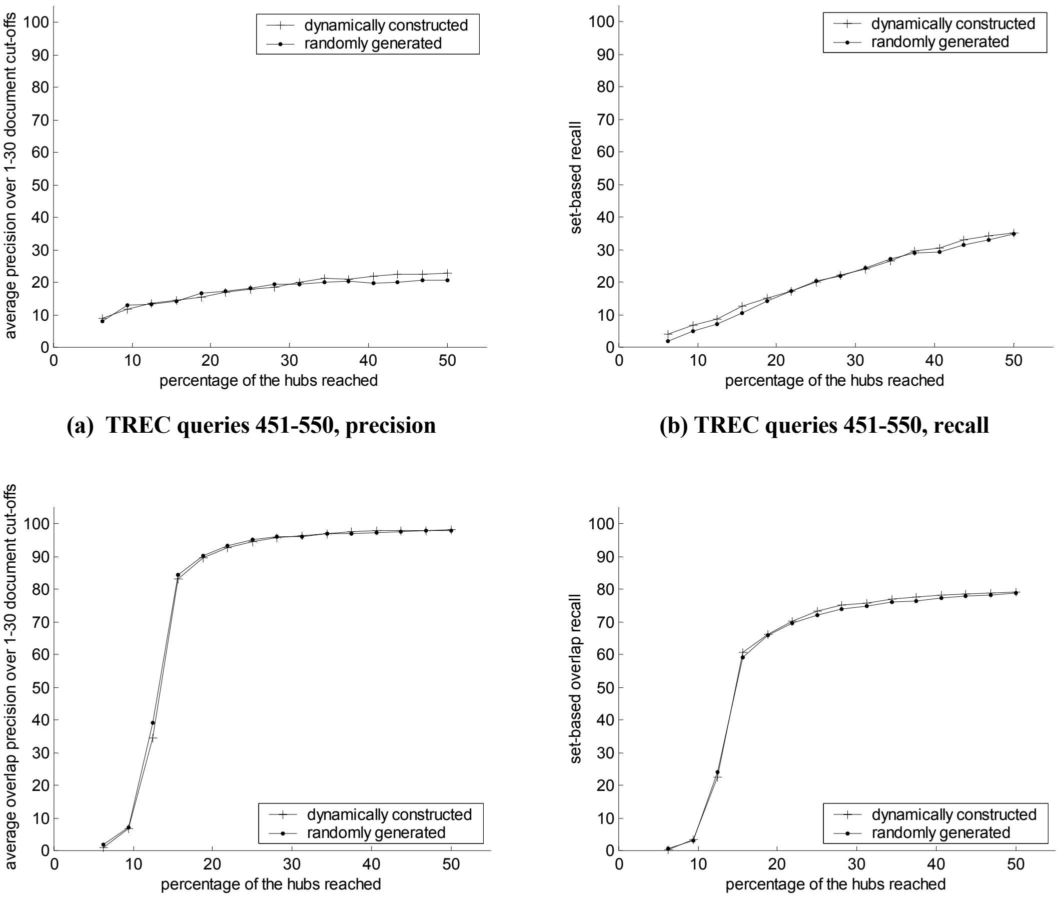
(d) dynamic w/o balance, large-sized network

Figure 7.3 The distributions of cluster sizes for different hub-provider topologies in the networks of different sizes.

balancing in combination with distributed cultivation of sub-clusters was able to spread the responsibility of serving popular contents to more hubs as well as reduce the number of hubs solely occupied by unpopular contents, resulting in a less skewed cluster size distribution.

In a few words, the cluster size distributions of the dynamically constructed hub-provider topologies in different network sizes demonstrate that our topology evolution algorithm's attempt to achieve more balanced load was successful.

The above experimental results demonstrate that the dynamic hub-provider topology evolution enabled a high degree of content-based locality and load balance. However, the aspect of topology evolution that concerns federated search most is whether a dynamically constructed hub-provider topology can further enhance search performance compared with a randomly generated hub-



(a) TREC queries 451-550, precision

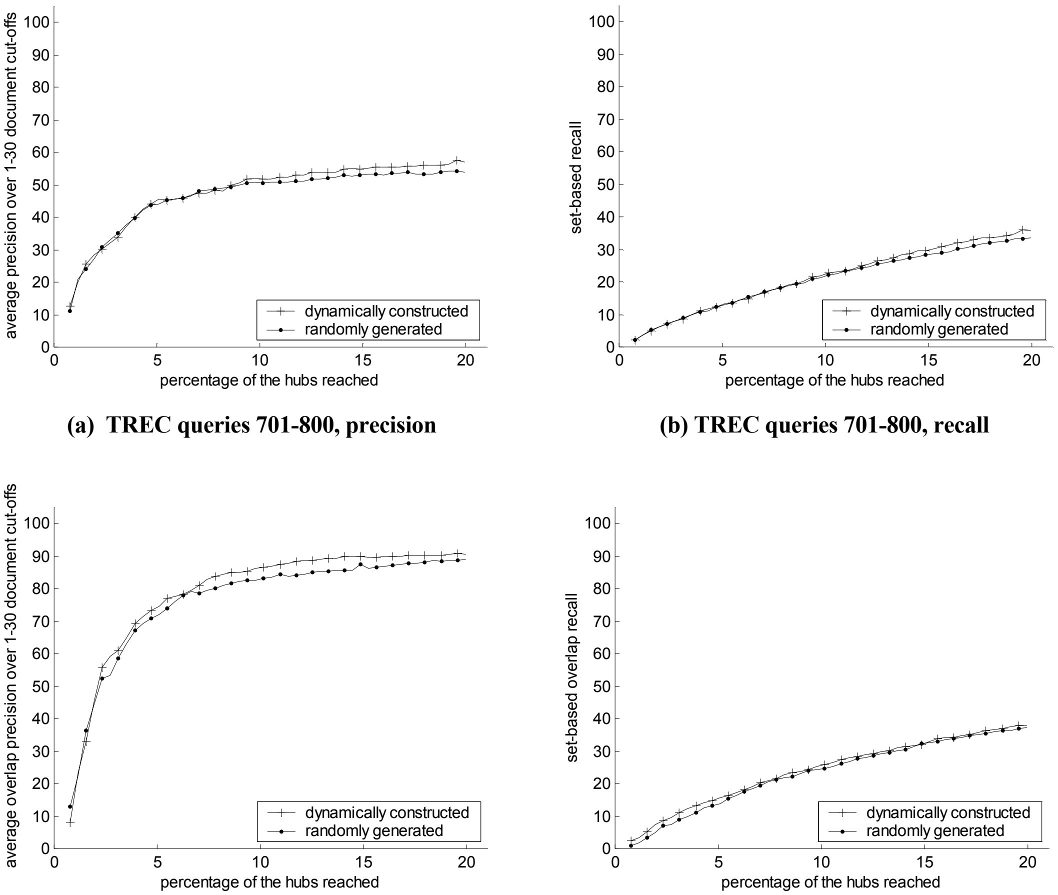
(b) TREC queries 451-550, recall

(c) WT10g queries, precision

(d) WT10g queries, recall

Figure 7.4 The search performance of different hub-provider topologies in the medium-sized network.

provider topology. Figures 7.4 (a)-(d) and 7.5 (a)-(d) compare the performance of federated search in precision and recall for different sets of queries in medium- and large-sized networks with dynamic or random hub-provider topologies. The figures show no significant performance improvement (except Figure 7.5 (c)) for the dynamically constructed hub-provider topologies, which seems to suggest that content-based locality didn't help in federated search. To explain the unattractive results, let's note that at this stage of the evaluation, although hub-provider connections were determined by content-based clustering, hub-hub connections were still random. The advantage of having each hub cover a cohesive content area was limited without the support of good navigability among the hubs, because the effectiveness of query routing at the hub level was the main factor that affected federated search performance. Therefore, content-based locality must be combined with content-based small-world properties in order to facilitate more effective and efficient federated search, the results of which are shown in the next section.



(a) TREC queries 701-800, precision

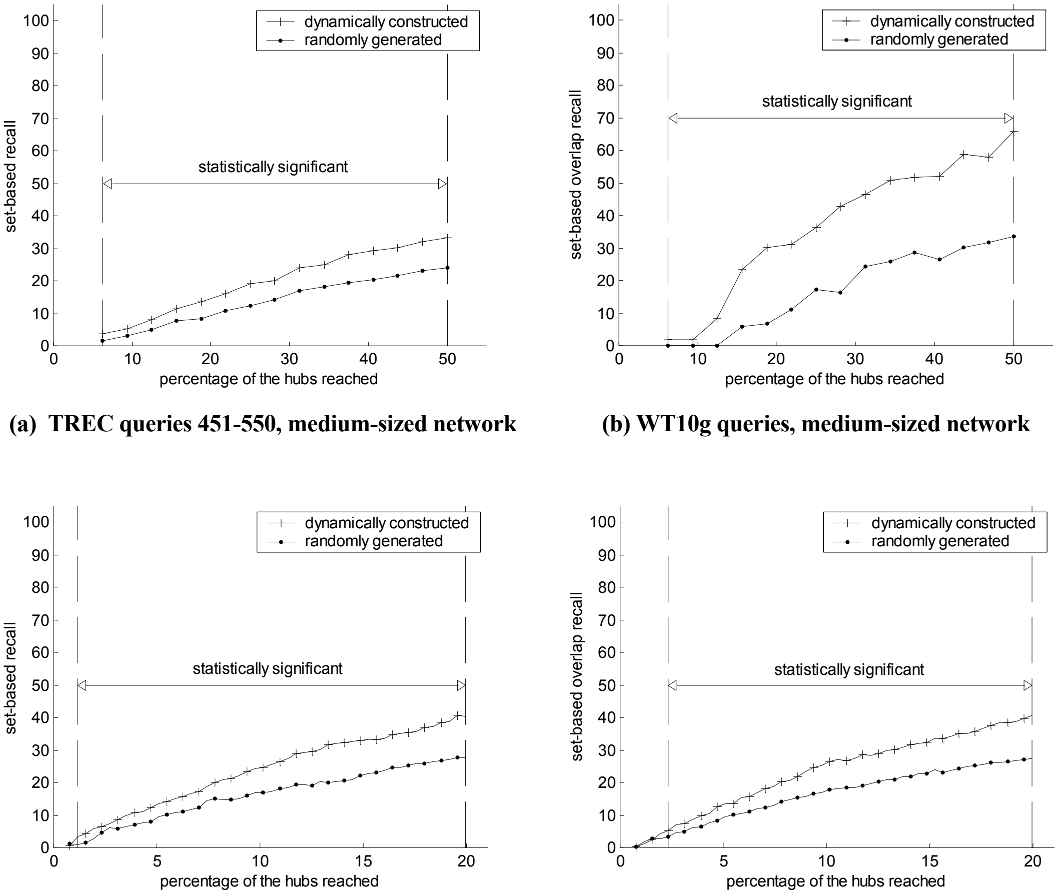
(b) TREC queries 701-800, recall

(c) GOV queries, precision

(d) GOV queries, recall

Figure 7.5 The search performance of different hub-provider topologies in the large-sized network.

The experiments above were run in a "static" network setting where the provider and neighborhood descriptions maintained by each hub were most up-to-date. To test whether the dynamically constructed hub-provider topologies could have a bigger advantage over random hub-provider topologies for full-text federated search based on dated resource descriptions in the face of dynamic content change in P2P networks, the following experiments were conducted. Given a dynamic or random hub-provider topology, for each query, an information provider containing relevant documents was randomly chosen with a probability proportional to its number of relevant documents, and its resource description was segregated from the description of the hub it connected to (in each of the hub-provider topologies to be evaluated) and consequently from the descriptions of all neighborhoods containing this hub. However, the description of the chosen provider was *not* removed from the storage of its connecting hub, so it was still visible to this hub but not to the rest of the network. This was to simulate the scenario in dynamic environments when a hub already



(a) TREC queries 451-550, medium-sized network

(b) WT10g queries, medium-sized network

(c) TREC queries 701-800, large-sized network

(d) GOV queries, large-sized network

Figure 7.6 The search performance (recall) of different hub-provider topologies based on "incomplete" neighborhood descriptions in the networks of different sizes.

acquired the description of a newly joined information provider but hasn't updated the corresponding neighborhood descriptions yet since it takes more resources and longer time to compute and update neighborhood descriptions.

Figures 7.6 (a)-(d) show the performance of federated search in medium- and large-sized networks with different hub-provider topologies based on "incomplete" neighborhood descriptions. One can expect that for a query with a sufficient number of relevant documents, the average precision at top-ranked documents was not likely to be affected very much even if the relevant provider whose description was excluded could not be reached, but the set-based recall was more likely to suffer in this case. Therefore, only the results in set-based (overlap) recall are included to highlight the differences in search performance. Since failure to reach one particular relevant provider wouldn't result in any substantial loss if the number of relevant documents it contributed was too small compared with the total number of relevant documents, the figures only show the results for those queries that had at least 10% of the relevant documents contained in the "excluded" relevant providers.

The figures illustrate that in the simulated "dynamic" settings, the dynamically constructed hub-provider topologies with content-based locality enabled substantially more relevant contents to be reached by federated search than the randomly generated hub-provider topologies. It was especially the case with WT10g queries, which have in general a very small number of relevant providers for each query so that any miss could significantly degrade search performance. Paired two-sided sign tests revealed that the improvement of search accuracy in the dynamically constructed hub-provider topologies was statistically significant at the 0.01 significance level when various percentages of the hubs were reached.

Compared with the results in the static settings, the dynamically constructed hub-provider topologies with content-based locality resulted in much less performance degradation than the randomly generated hub-provider topologies in the simulated "dynamic" settings. By having content-based locality, the volatility of hub and neighborhood descriptions in dynamic environments was greatly reduced so that even if the description of a relevant provider was excluded, neighborhood descriptions still provided enough clues to guide resource selection towards the right direction in locating this provider. Our experimental results show that as neighborhood descriptions became more out-of-date and failed to include the descriptions of the providers covering a larger percentage of the relevant documents, the performance of federated search in the hub-provider topologies with content-based locality degraded more gracefully. Even if random selection was used for query routing among the hubs due to the lack of updated resource descriptions, the search performance was still better in the content-based hub-provider topologies than in the random topologies, making federated search more resilient to dynamic content change in the network.

In summary, our topology evolution algorithm for hub-provider topology was effective in dynamically constructing a hub-provider topology with a high degree of content-based locality and load balance in an efficient and scalable manner, and the resulting topology enabled slightly more effective and much more robust full-text federated search compared with a random hub-provider topology.

7.2 Hub-Hub Topology

Basing hub-provider topology evolution on the dynamic topology evolution algorithm evaluated in previous section, this section focuses on the effectiveness of the hub-hub topology evolution algorithm described in Section 6.2. We are primarily interested in i) whether the dynamically constructed hub-hub topology exhibited content-based small-world properties and balanced hub degrees, and ii) whether it was effective in enhancing search performance compared with a randomly generated hub-hub topology.

7.2.1 Experimental Settings

The same testbeds and settings described in Section 7.1.1 were used with the exception that the connections between hubs were adjusted dynamically using the hub-hub topology evolution algorithm instead of remaining static.

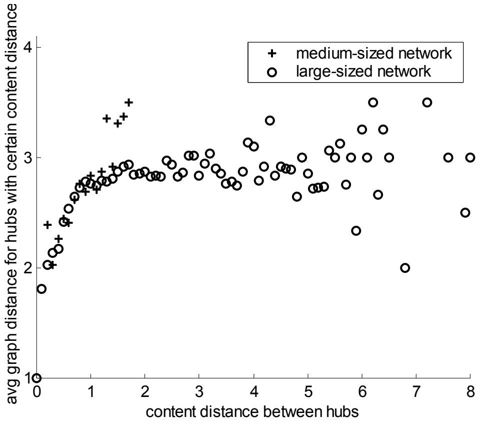
Because in reality large-scale P2P networks are typically sparse, and it is quite expensive for each hub to acquire and maintain neighborhood descriptions for a large number of hub neighbors, we chose small values for various maximum connection capacities to simulate the topology evolution of a sparse network. For the medium-sized network, a hub's maximum number of outgoing local hub connections M_{ol} was 2, its maximum number of outgoing long-range hub connections M_{og} was 3 minus its actual number of outgoing local hub connections, and its maximum number of incoming hub connections M_i was 3. For the large-sized network, the values of M_{ol}, M_{og}, and M_i were 4, 4, and 12 respectively. For both networks, the exponent β for the power-law distribution was 2.0.

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

7.2.2 Experimental Results

The characteristics of content-based small-world properties at the hub level are locational proximity of similar content areas and short global separation of dissimilar content areas. To evaluate whether the dynamically constructed hub-hub topology had content-based small-world properties, both the content distance (K-L divergence between hub descriptions) and the graph distance (number of hops) between each pair of hubs were calculated and the graph distances for different pairs of hubs with the same range of content distance were averaged. Figure 7.7 plots the relations between content distances and graph distances for hub pairs in both the medium-sized network and the large-sized network. The figure shows not only small graph distances between hubs with similar content areas (K-L divergence no more than 1.0), but also small-to-medium graph distances between hubs with dissimilar content areas. Particularly, for the large-sized network with as many as 256 hubs and a network density of 0.0388, the average graph distances didn't exceed 4 and the maximum graph distance was 5 for hub pairs with very large content distances. Therefore, the dynamically constructed hub-hub topologies exhibited content-based small-world properties.

Figure 7.8 depicts the degree distributions at the hub level for the dynamically constructed hub-hub topologies. Although the degree distribution of the medium-sized network shows nothing interesting due to its small scale, the degree distribution of the large-sized network illustrates that the dynamically constructed hub-hub topology for the network of 256 hubs didn't have a power-law



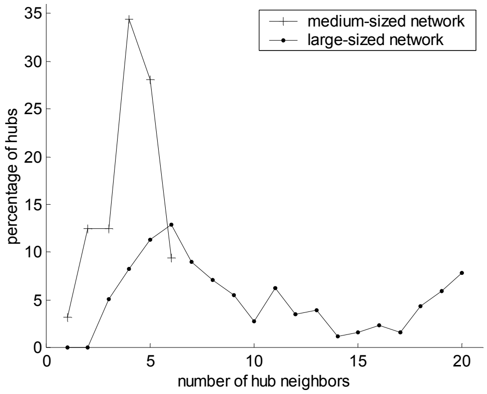
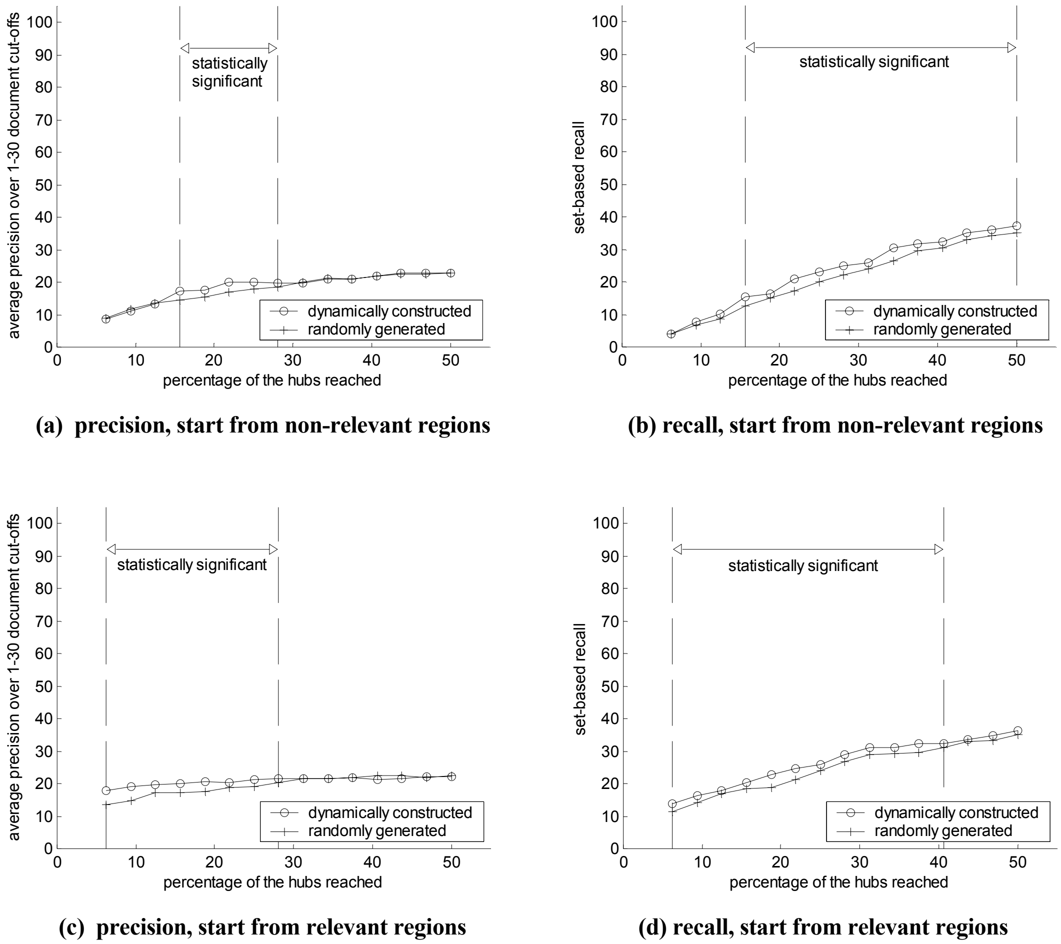


Figure 7.7 Content distances vs. graph distances. **Figure 7.8 Hub-hub degree distributions.**

distribution due to the constraints of the topology evolution algorithm on the minimum and maximum numbers of hub connections for each hub, which was designed on purpose to avoid overloading a few hubs with a large number of connections by distributing these connections to a larger number of hubs. When the network scales to an even larger size, the middle range of the distribution curve is expected to behave more and more like a power-law distribution, but instead of a long, diminishing tail to the right side which is typical of a power-law distribution, the curve will turn upward before finishing at the maximum allowed connection capacity. Because the highly connected peers in a network topology with a power-law degree distribution can easily become bottlenecks of information flow or targets of malicious attacks, balancing hub degrees by truncating the power-law degree distribution can effectively alleviate these problems.

If a hub-hub topology exhibits content-based small-world properties, when an information request is initiated from a network region containing relevant contents, which is most likely the case with characteristic queries of persistent interests issued by consumers to selected hubs (Section 4.4) in a dynamically constructed hub-consumer topology (Section 6.3), 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 region, which may happen for uncharacteristic queries of transient interests, the request should only need to travel along a short path to reach a relevant region thanks to the short path length between hubs of dissimilar content areas. Therefore, whether content-based small-world properties can enhance search performance can be measured by the effectiveness and efficiency of hub-hub query routing when queries start from different regions.

Figures 7.9-7.12 show the performance of federated search for different sets of queries in different networks with the dynamically constructed topologies. To start search in a relevant region, each query was initiated by a consumer connected to a hub selected among those nearest to relevant content. To start search in a non-relevant region, each query was issued by a consumer connected to a hub located farthest on average from relevant content. The results of federated search in the networks with random hub-hub topologies are included for comparison. Paired two-sided sign tests



(a) precision, start from non-relevant regions

(b) recall, start from non-relevant regions

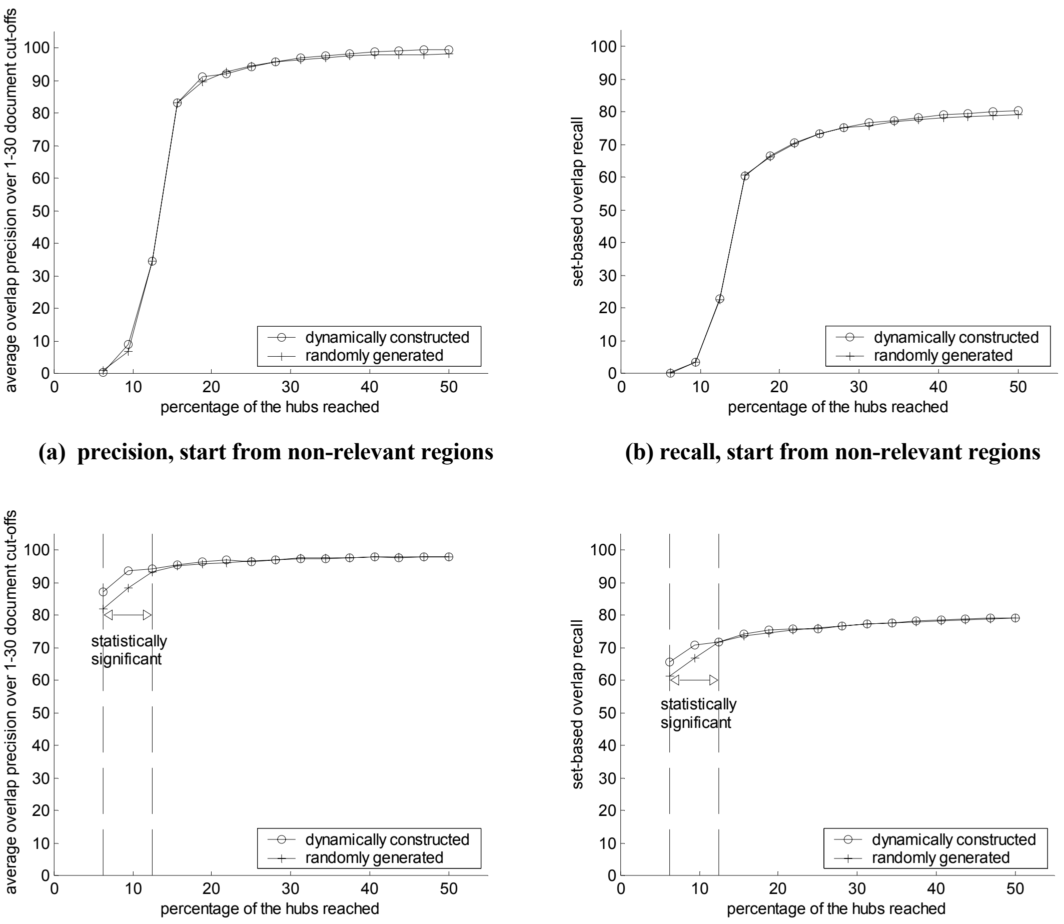
(c) precision, start from relevant regions

(d) recall, start from relevant regions

Figure 7.9 The search performance of different hub-hub topologies for TREC queries 451-550 in the medium-sized network with search started from different regions.

were applied, and the vertical dashed lines in the figures mark the ranges within which the dynamically constructed topologies yielded statistically significant improvement at the 0.01 significance level compared with the random topologies. Lack of vertical dashed lines in some figures indicates that the results shown in these figures had no statistically significant difference.

The figures indicate that when queries were initiated from relevant regions, compared with search in the random hub-hub topologies, federated search in the dynamic hub-hub topologies had better performance with statistical significance right from the beginning when a very small percentage of the hubs were reached. The improvement can be explained by the fact that similar contents (and therefore more relevant contents) were near to one another in the dynamically constructed hub-hub topologies but most certainly not so in the random hub-hub topologies. When queries were initiated from non-relevant regions, search in the dynamically constructed hub-hub topologies had



(a) precision, start from non-relevant regions

(b) recall, start from non-relevant regions

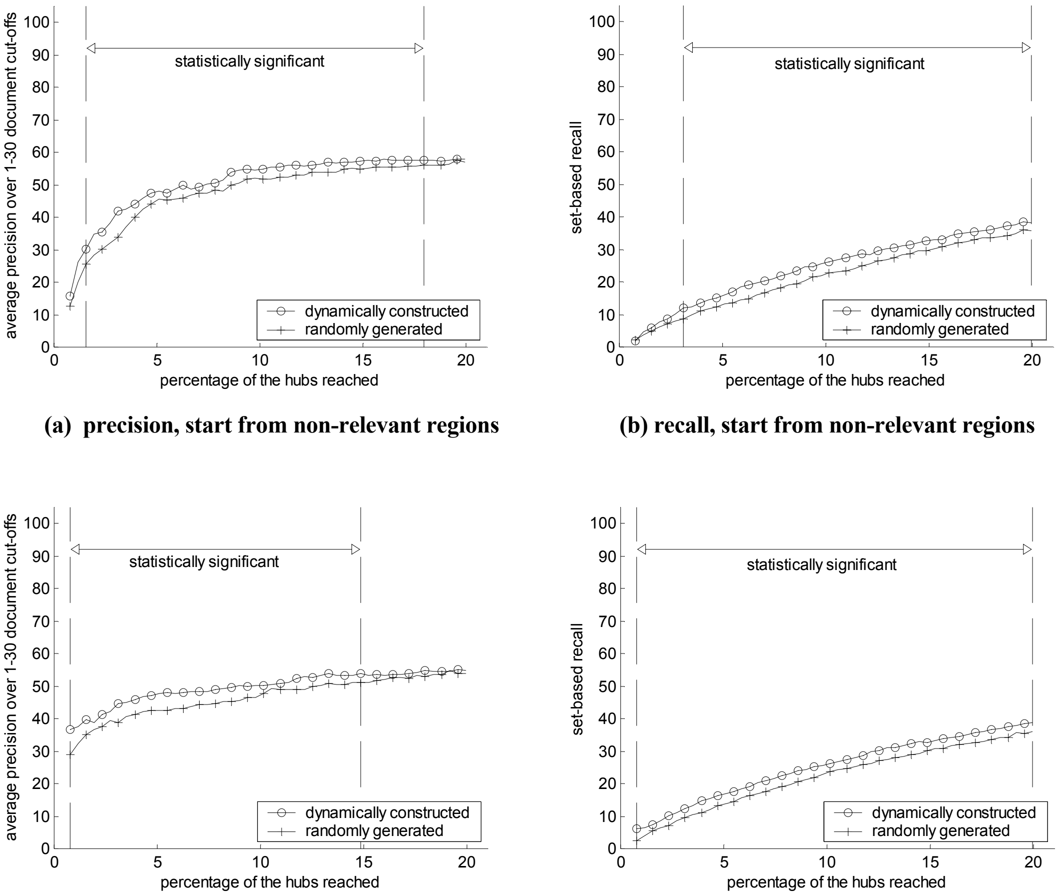
(c) precision, start from relevant regions

(d) recall, start from relevant regions

Figure 7.10 The search performance of different hub-hub topologies for WT10g queries in the medium-sized network with search started from different regions.

statistically significantly better performance for TREC queries (precision and recall) and GOV queries (recall), and similar performance for WT10g queries. This was not only because short global separation between dissimilar content areas enabled hub-hub query routing to reach relevant contents quickly, but also because locational proximity of similar content areas resulted in additional relevant contents (if any) to be located in short distance once relevant regions were reached.

The range of settings within which the dynamically constructed hub-hub topologies had statistically significant improvement over the random topologies was mostly wider for queries initiated from relevant network regions than for those from non-relevant regions. Routing a query from a non-relevant neighborhood to a relevant neighborhood, and then to the most relevant hub required a sequence of correct decisions made by the resource selection component of each hub along the path.



(a) precision, start from non-relevant regions

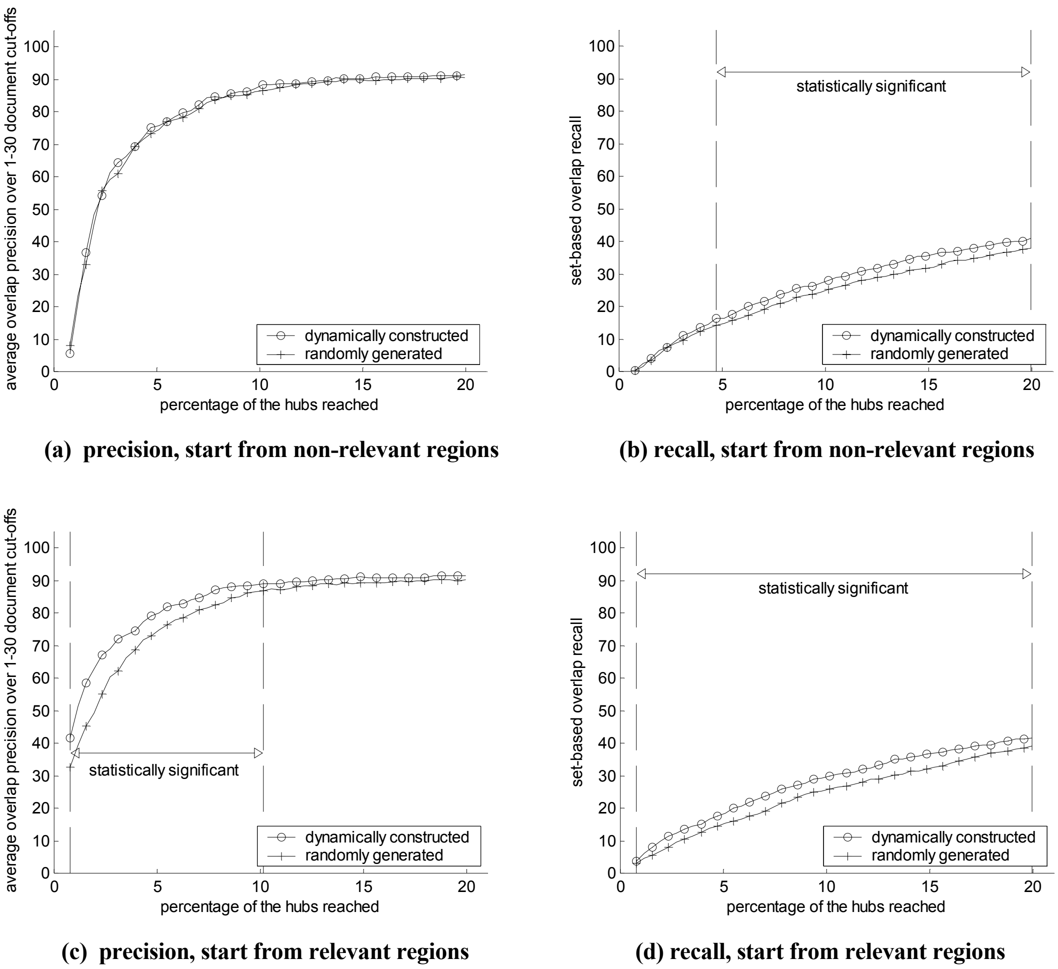
(b) recall, start from non-relevant regions

(c) precision, start from relevant regions

(d) recall, start from relevant regions

Figure 7.11 The search performance of different hub-hub topologies for TREC queries 701-800 in the large-sized network with search started from different regions.

Restricting hubs to selecting only one neighboring hub in our experiments yielded longer path length for query routing and higher likelihood of selecting a suboptimal hub somewhere along the path, thus increasing the chance for query routing to deviate from the optimal path and diminishing the improvement in the search performance of the dynamically constructed topologies. Possible methods to enhance the advantage of the dynamically constructed content-based topologies include allowing each hub to route the query it receives to multiple neighboring hubs, selecting a variable instead of fixed number of hub neighbors for query routing based on their estimated relevance for the query, or making an effort to start the query in a relevant neighborhood in order to reduce the path length to reach relevant content. In this dissertation, we focus on the last method, using user modeling to dynamically construct a hub-consumer topology and conduct interest-based hub selection, so that queries of persistent user interests can start in relevant neighborhoods. The next section (Section 7.3) describes in detail the evaluation on the effectiveness of this method.



(a) precision, start from non-relevant regions

(b) recall, start from non-relevant regions

(c) precision, start from relevant regions

(d) recall, start from relevant regions

Figure 7.12 The search performance of different hub-hub topologies for GOV queries in the large-sized network with search started from different regions.

By comparing Figures 7.9-7.12, we can see that the performance gain by using a dynamically constructed topology was bigger and more consistent in the large network than in the medium-sized network. Another observation is that the performance differences between different topologies were bigger for queries whose relevant documents were mostly covered by multiple hubs (TREC queries 451-500, TREC queries 701-800, GOV queries) instead of just one or two hubs (WT10g queries). These indicate that as the network grew larger to include more contents and connections, efficient resource location became a more difficult task, therefore having content-based small-world properties for good navigability yielded a bigger advantage.

To summarize, by only utilizing limited local information about the network, the topology evolution algorithm for hub-hub topology was effective in dynamically constructing a hub-hub topology with content-based small-world properties and relatively balanced hub degrees, and the resulting topology could further improve the effectiveness (particularly recall) and efficiency of full-text federated search compared with a random hub-hub topology.

7.3 Hub-Consumer Topology

Because the establishment and adjustment of each consumer's permanent connections to the hubs during topology evolution are completely based on the continuous observation about which hubs are most frequently selected by the consumer in recent search, the effectiveness of hub-consumer topology evolution can be evaluated by the performance of initial hub selection conducted by the consumer. Section 4.4 proposes a resource selection method that uses a user model constructed by clustering past queries based on their top-ranked search results to recognize different types of user interests ("*characteristic*", persistent interests versus "*uncharacteristic*", transient interests) and different topics of characteristic interests, and applies different search strategies accordingly in order to take advantage of the strengths of both resource selection by consumers and resource selection by hubs. Since its effectiveness largely depends on regulated content placement and carefully controlled topology evolution, it naturally becomes the last component to be evaluated in our progressive evaluation of the network evolution model.

Given the support of the dynamically evolved hub-provider and hub-hub topologies, the performance of the proposed method was measured for both characteristic queries and uncharacteristic queries and compared against the performance of several other hub selection methods. Before presenting the experimental results and analysis in Section 7.3.4, we devote Section 7.3.1 to introducing the hub selection methods used for comparison, Section 7.3.2 to presenting the query sets created to include different types of queries in different topics, and Section 7.3.3 to describing experimental settings.

7.3.1 Hub Selection Methods

The hub selection method described in Section 4.4 is referred to as *hub selection based on clustered user modeling* in order to distinguish it from other hub selection methods introduced in this section. It uses interest-based initial hub selection based on the clustered user model for characteristic queries representing persistent interests, and random initial hub selection followed by full-text hub selection conducted by hubs for uncharacteristic queries of transient information needs.

Hub selection based on non-clustered user modeling generates a non-clustered user model by aggregating the contents of the top-ranked retrieved documents for previous queries, and uses the same performance measure as hub selection based on clustered user modeling to evaluate the hubs. Because it constructs a user model explicitly, it has the ability to separate uncharacteristic queries from characteristic ones in order to apply different search strategies. However, it can only measure each hub's performance for past queries as a whole without distinguishing the differences in performance for different interests.

Performance-based hub selection uses the same measure for the hubs' resource location performance as hub selection based on user modeling. However, it maintains the hubs' measured performance by accumulating the total number of top-ranked merged documents returned by each hub for previous queries without explicitly constructing a user model from document contents. As a result, it lacks the ability to distinguish between characteristic queries representing persistent user interests and uncharacteristic queries expressing transient information needs, which means that only a single search strategy can be applied to all the queries.

Content-based hub selection uses the K-L divergence resource selection algorithm based on the content models of the hubs learned from previous search results. Each consumer cumulatively constructs its own hub models using the contents of the top-ranked documents it received from the hubs for previous queries. Similar to performance-based hub selection, content-based hub selection neither constructs a user model nor distinguishes different interests. Therefore, it also applies a single search strategy to all the queries.

Random hub selection randomly selects hubs from the list of hubs maintained by each consumer to issue queries.

Random+full-text hub selection randomly selects one hub to start search and continues with full-text hub selection conducted by hubs for multiple-hop hub-hub query routing.

To highlight the differences between different hub selection methods in terms of search strategies, hub selection based on clustered or non-clustered user modeling uses two different strategies for characteristic and uncharacteristic queries. Hub selection based on clustered user modeling uses interest-based initial hub selection by consumers for characteristic queries, while hub selection based on non-clustered user modeling uses performance-based initial hub selection by consumers for characteristic queries. Both methods use random initial hub selection by consumers followed by full-text hub selection conducted by hubs for uncharacteristic queries. Performance-based, content-based, and random hub selection methods treat all queries as characteristic queries and only use initial hub selection conducted by consumers to reach the hubs. Random+full-text hub selection treats all queries as uncharacteristic queries and uses random initial hub selection by consumers followed by full-text hub selection conducted by hubs to reach the hubs.

7.3.2 Queries

Two sets of queries were selected from the queries automatically generated by extracting key terms from the documents in WT10g (Section 5.1.3) to be used in the medium-sized P2P network. The first query set ("*WT10g-broad*") consists of 563 characteristic queries manually chosen to represent

Table 7.1 Stemmed sample queries from WT10g-broad query set.

Topic	# queries	Sample queries
Music	72	Billy Joel, Adam Ant album, Jesse Jones play band
Finance	67	capital Macquire, common share Chrysler, mortgage market product
Education	80	elementary educate, Stanford university program, ohio college
Health	78	medical rehabilitate, home care nurse, primary care physician Santara
Technology	75	BSDI Internet, free agent software, secure product kerbero
Law	64	supreme court, law resource legal federal, war crime international law
Religion	67	lord Samuel Israel, holy spirit testament, god Jesus church sin
Government	60	tax cut, budget deficit govern, federal govern department
Uncharacteristic	437	Ocean Spray, CraftWEB bookstore, Torreblanca resort Acapulco

Table 7.2 Stemmed sample queries from WT10g-narrow query set.

Topic	# queries	Sample queries
Classical music	50	Bach sonata, Richard Strauss record, ninth symphony Beethoven
Stock	50	stock split, Dow Jones index, Alcoa pay bonus dividend
Online education	50	distance educate, enroll online course, university phoenix online
Personal health	50	nutrition vitamin, calorie fat pretzel, fat oil cholesterol
Image processing	50	Adobe Photoshop, image browse Kudo, Epson photo image software
Civic regulation	50	water hazard rule, waste pollution control, sewage sludge regulate
Religious study	50	Christian theology, religion history study, lecture Islamic Muslim
Tax issues	50	income tax, tax reform, tax cut legislate

a user's persistent interests in 8 relatively broad topics, and 437 uncharacteristic queries automatically selected to express the user's transient information needs not related to the aforementioned 8 topics. The topics were determined by soft-clustering the 2,500 providers based on their resource descriptions and inspecting the most frequent non-stopword terms from each cluster. Therefore, these topics are representative of the contents provided in the medium-sized network. Table 7.1 shows for each "broad" topic a general description, the number of queries selected for the topic, and sample queries with query terms stemmed using the k-stem stemmer (Krovetz 1993). Among the 8 topics, "Finance", "Education", "Health" and "Technology" are popular in the network with a large number of providers providing related contents. By comparison, "Music", "Law", "Religion", and "Government" are much less popular. Samples of uncharacteristic queries are also included in the table. The second query set ("*WT10g-narrow*") includes 400 characteristic queries in 8 topics which can be regarded as sub-topics of the above "broad" topics and 600 uncharacteristic queries. Table 7.2 shows sample queries for these "narrow" topics.

Table 7.3 Sample queries from GOV-broad query set.

Domain	# queries	Sample queries
USGS	100	pelicans, bald eagle, ruby throated hummingbird
NIH	100	cirrhosis, kidney stones, restless leg syndrome
NPS	100	sequoia, grand canyon, great smoky mountains
NASA	100	jupiter, electromagnetic spectrum, columbia space shuttle
BLS	100	paralegal, respiratory therapist, occupational outlook handbook
Uncharacteristic	500	commonwealth, us senators, energy policy act of 2005

Table 7.4 Sample queries from GOV-narrow query set.

Domain	# queries	Sample queries
IRS	100	1040ez, tax refund, earned income tax credit
NOAA	100	storms, doppler radar, national weather service
FTC	50	debt, cash loans, credit report companies
ED	60	mentoring, school finder, post secondary education
HUD	90	mortgages, buying a house, first time home owner
DOT	50	airbags, car safety, drunk driving laws in south Carolina
DOL	50	unemployment, labor laws, injury compensation for federal employees

Queries for the large-sized P2P network were selected from the query set provided by AOL for *.gov domains. Each query was associated with the domain of the search result clicked by one or more users. Queries associated with the same domain were considered to be from the same topic. The *GOV-broad* query set consists of 500 characteristic queries manually chosen from 5 relatively broad domains, and 500 uncharacteristic queries not related to these 5 domains. The *GOV-narrow* query set consists of 500 characteristic queries selected from 7 relatively narrow domains, and 500 uncharacteristic queries from other domains. Tables 7.3 and 7.4 show respectively sample queries for these two query sets.

7.3.3 Experimental Settings

The testbeds used were the same as those defined in Section 5.1 and used for evaluating the network search model (Section 5.3), the hub-provider topology evolution algorithm (Section 7.1), and the hub-hub topology evolution algorithm (Section 7.2). The hub-provider and hub-hub topologies were constructed by using the topology evolution algorithms described in Sections 6.1 and 6.2, and evaluated in Sections 7.1 and 7.2.

Given a query set, queries in the set were issued by an information consumer in a random order. The information consumer was *not* given information about which queries represented which types or topics of interests. Regardless of the hub selection method, the first 50 queries (among the 1,000 queries in a given query set) were issued to one randomly selected hub with a TTL value of 15 for the medium-sized network or 50 for the large-sized network so that the consumer could learn sufficient information about the hubs in the network. For performance-based, content-based, and random hub selection, all the queries after the first 50 were issued as characteristic queries. For random+full-text hub selection, all the queries were issued as uncharacteristic queries. For hub selection based on clustered or non-clustered user modeling, after the first 50 queries, it was up to the consumer to decide whether to issue a query as a characteristic query or as an uncharacteristic query. Specifically, the K-L divergence values between the query and existing query clusters whose size exceeded S_{min} = 5 were calculated.[25] If none of the divergence values were smaller than the classification threshold $T_{classify}$ which was set to 7.0 for hub selection based on clustered user modeling and 9.0 for that based on non-clustered user modeling[26], then the query was regarded as uncharacteristic; otherwise, it was considered characteristic. A query issued as characteristic query was sent to the top-ranked hubs selected using initial hub selection conducted by the consumer with a TTL value of zero for hub routing so that federated search completely relied on initial hub selection to reach the hubs. In this case, the percentage of the hubs reached for the query was controlled by the number of the top-ranked hubs selected by initial hub selection. A query issued as uncharacteristic was sent to a randomly selected hub with a non-zero TTL value so that federated search used full-text hub selection conducted by a hub to reach other hubs in the network. The value of TTL determined the percentage of the hubs reached for the query.

For resource selection by hubs, each hub that received the query message with a non-zero TTL value forwarded it to the top-ranked neighboring providers based on the learned threshold and the top one neighboring hub that hadn't been reached for the query with a decreased TTL value. For document retrieval, each provider returned up to 50 top-ranked documents. For result merging, the hubs used the extended Kirsch's algorithm and returned the top-ranked documents (50 for the medium-sized network and 200 for the large-sized network) to the consumer. The consumer used the raw score merge to merge the results returned by multiple hubs.

For hub selection based on clustered user modeling, the consumer used the contents of D_{top} = 10 top-ranked merged documents to generate a more detailed representation of the query, and its similarity to existing query clusters was measured by the K-L divergence values between the query representation and the representations of these clusters. The clustering threshold $T_{cluster}$ used to determine whether to include the query into existing clusters or to create a new cluster was set to 1.5. The maximum number of recorded clusters N_{max} was 50. Therefore, when the number of

[25] For hub selection based on non-clustered user modeling, there was only one single query cluster for all previous queries that were considered characteristic.

[26] The classification threshold $T_{classify}$ for hub selection based on non-clustered user modeling had a looser value because the representation of the single query cluster for all characteristic queries of various interests was more heterogeneous than the representation of a query cluster for queries of a particular interest.

clusters reached 50 but a new cluster was needed, the smallest cluster among the $r = N_{max} /4$ least recently used clusters was removed to make room for the new cluster. The information about how many documents returned by each hub appeared among the D_{top} top-ranked merged documents was used to update the hubs' measured resource location performance for the cluster which the query joined or created.

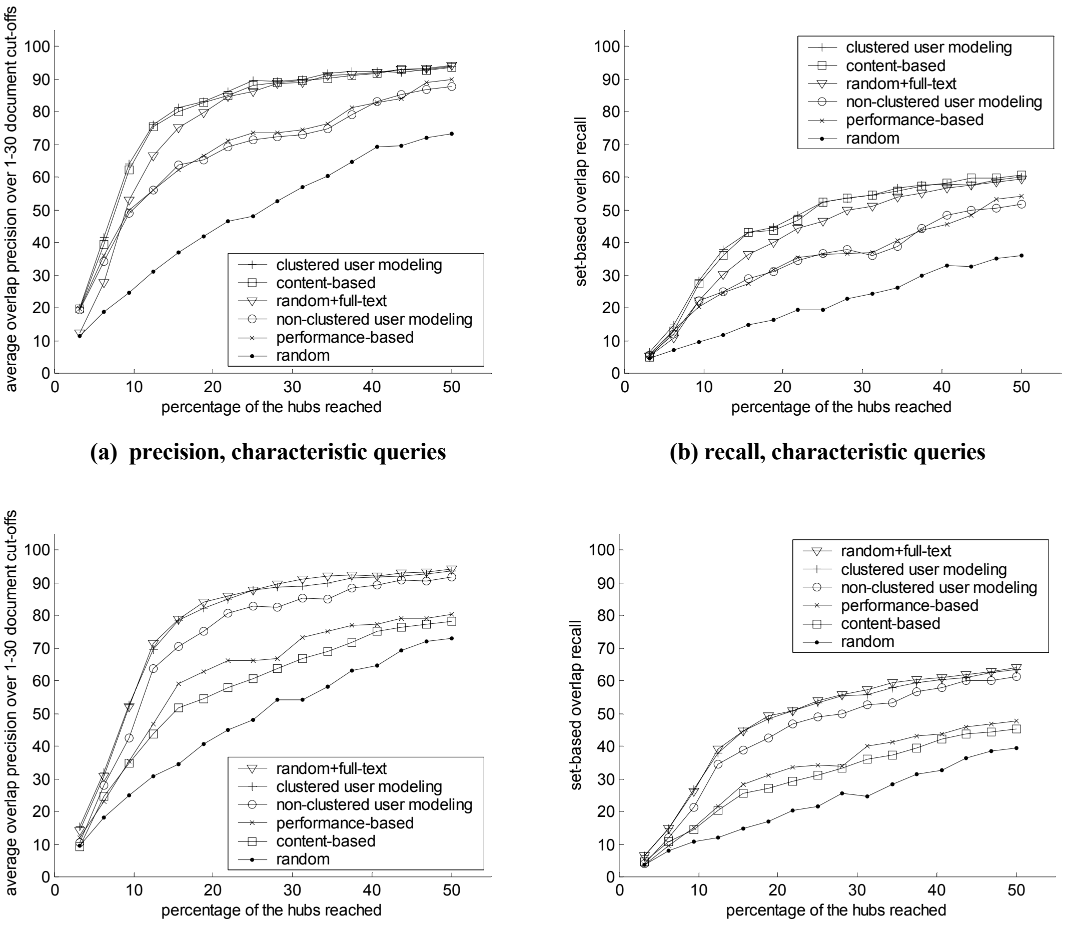
For hub selection based on non-clustered user modeling, whether to integrate the new representation generated from the D_{top} top-ranked merged documents into existing user model depended on whether the K-L divergence between them was less than $T_{cluster}$. The information about how many documents returned by each hub appeared among the D_{top} top-ranked merged documents was accumulated as a way to measure the hubs' resource location performance for hub selection based on non-clustered user modeling and for performance-based hub selection. Content-based hub selection used the contents of D_{top} top-ranked documents returned by the hubs to update the corresponding hub content models maintained by the consumer.

The similarity thresholds $T_{classify}$ and $T_{cluster}$ could be tuned automatically by the system given the types of a small number of queries as training data. Our previous experimental results show that the performance of hub selection based on user modeling is quite robust when the similarity threshold values are chosen within a certain range (Lu and Callan 2006b). Small values for S_{min} (the minimum cluster size for representing a persistent interest) and D_{top} (the number of the top-ranked documents per query used by hub selection methods) were used since our previous work indicates that a small amount of training data are sufficient in learning a user model with satisfactory accuracy (Lu and Callan 2006b).

7.3.4 Experimental Results

This section compares the performance of full-text federated search using different methods of hub selection in the dynamically evolved network topologies.

Figures 7.13 (a)-(d) plot the search accuracy (y-axis) against the percentage of the hubs reached (x-axis) for the WT10g-broad query set using different hub selection methods in the medium-sized P2P network. The average performance values for characteristic queries and uncharacteristic queries are shown separately. Figures 7.13 (a)-(b) show that hub selection based on non-clustered user modeling underperformed that based on clustered user modeling for characteristic queries. This was because although the best hubs for different interests were most likely to be different due to content-based locality, without using query clustering to distinguish between different characteristic interests, hub selection based on non-clustered user modeling could only select hubs that had best resource location performance for characteristic queries in general but not necessarily for those of a particular interest. By comparison, hub selection based on clustered or non-clustered user modeling yielded similar performance for uncharacteristic queries since they were both capable of distinguishing uncharacteristic queries from characteristic ones so that federated search could



(a) precision, characteristic queries

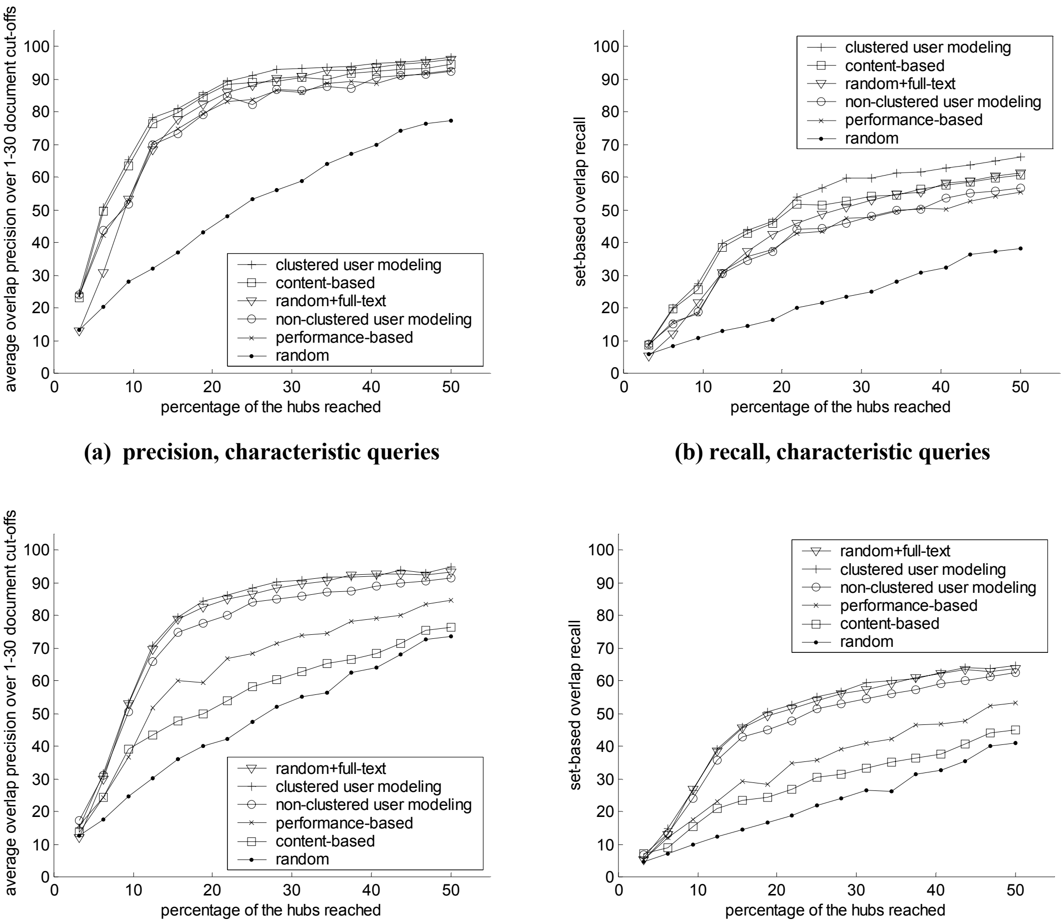
(b) recall, characteristic queries

(c) precision, uncharacteristic queries

(d) recall, uncharacteristic queries

Figure 7.13 The search performance of different hub selection methods for WT10g-broad queries in the medium-sized network.

rely on full-text hub selection conducted by hubs to guarantee effectiveness. Performance-based hub selection had similar search performance for characteristic queries compared with hub selection based on non-clustered user modeling since they essentially used the same measure for hubs' resource location performance. However, due to its inability to recognize uncharacteristic queries and apply a different search strategy accordingly, performance-based hub selection was much less effective than hub selection based on user modeling for uncharacteristic queries. Content-based hub selection based on the learned hub content models was very effective for characteristic queries but quite ineffective for uncharacteristic queries, which was not a surprise since the limited information the consumer learned about the hubs as a byproduct of past search was only helpful in guiding initial hub selection for future queries similar to past ones. As expected, random hub selection which blindly selected hubs resulted in the worse performance among all hub selection methods. Compared with random+full-text hub selection, hub selection based on clustered user modeling had



(a) precision, characteristic queries

(b) recall, characteristic queries

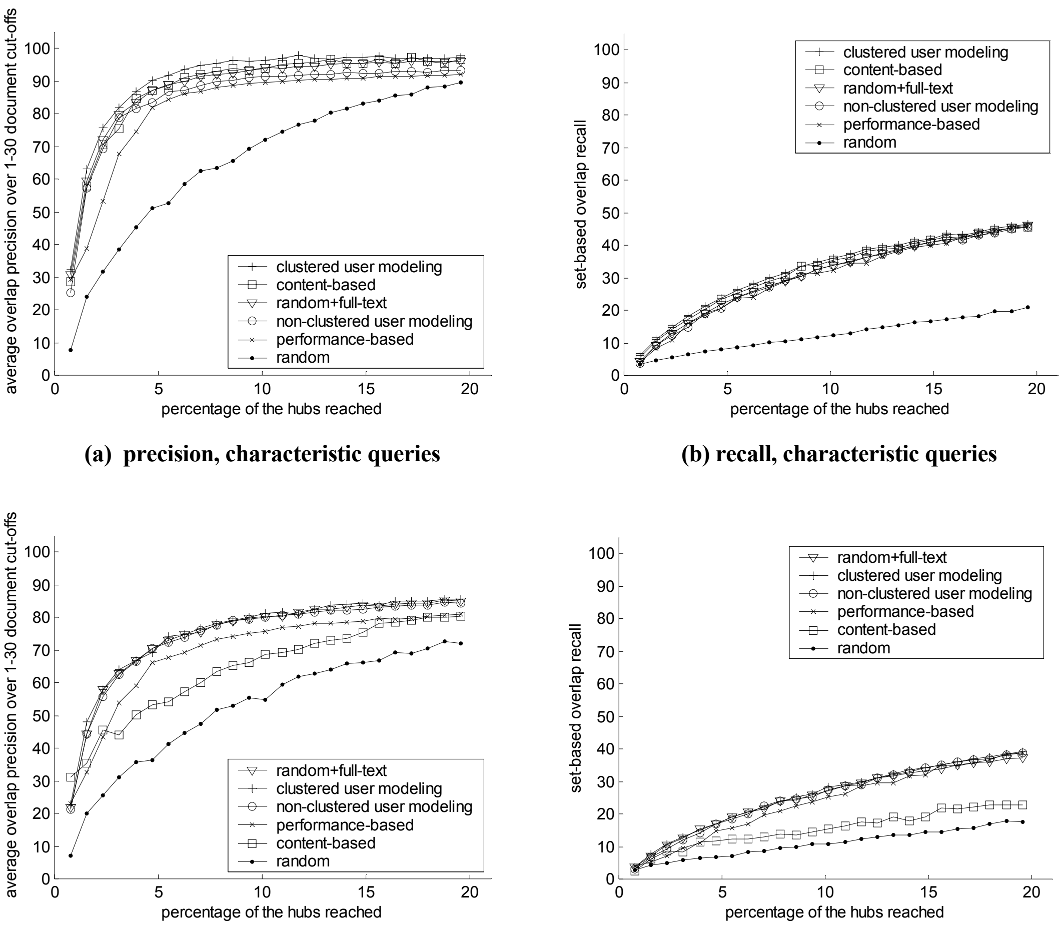
(c) precision, uncharacteristic queries

(d) recall, uncharacteristic queries

Figure 7.14 The search performance of different hub selection methods for WT10g-narrow queries in the medium-sized network.

better performance for characteristic queries and comparable performance for uncharacteristic queries, demonstrating that it was able to combine the strengths of both resource selection by consumers and resource selection by hubs, which outperformed federated search that only relied on the power of resource selection by hubs. Overall, hub selection based on clustered user modeling consistently gave near-best performance for both characteristic and uncharacteristic queries of the WT10g-broad query set in the medium-sized network.

Figures 7.14 (a)-(d) show the experimental results for the WT10g-narrow query set using different hub selection methods in the medium-sized P2P network, which led to the same conclusions with respect to the relative effectiveness of different hub selection methods as those drawn from the results of the WT10-broad query set. The main difference between the results for these two query sets is that hub selection based on non-clustered user modeling and performance-based hub



(a) precision, characteristic queries

(b) recall, characteristic queries

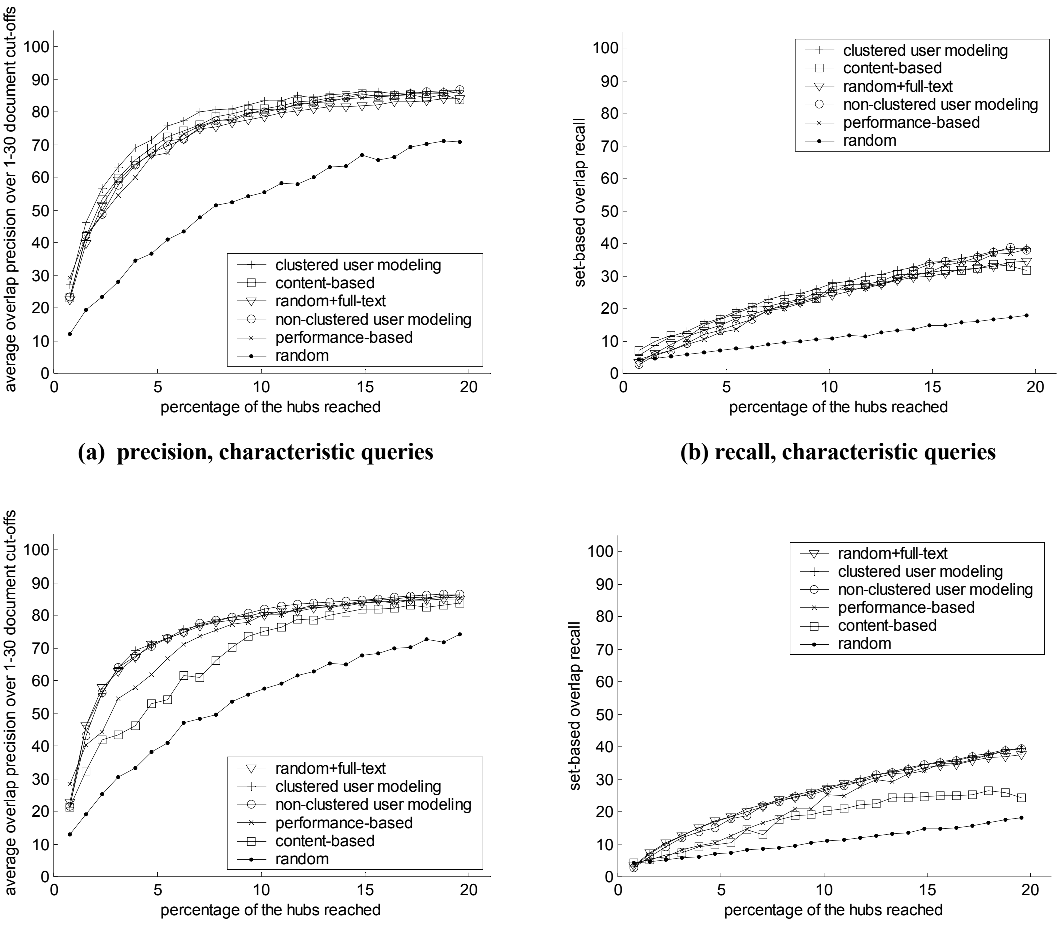
(c) precision, uncharacteristic queries

(d) recall, uncharacteristic queries

Figure 7.15 The search performance of different hub selection methods for GOV-broad queries in the large-sized network.

selection were much more effective for characteristic queries in the WT10g-narrow query set than for characteristic queries in the WT10g-broad query set. One possible reason to explain the difference was that queries of more focused interests (as those in the WT10g-narrow query set) required fewer hubs to cover most relevant contents and these hubs more easily stood out even without distinguishing the hubs' resource location performance for different interests, which made performance-based initial hub selection more effective as long as the number of hubs to be selected was not too small.

One might expect that user interests that were more focused could be better modeled and could enable more effective interest-based initial hub selection. However, because each query cluster used to represent a persistent interest was created automatically and adaptively instead of manually or statically, each broad interest provided in the WT10g-broad query set could be automatically



(a) precision, characteristic queries

(b) recall, characteristic queries

(c) precision, uncharacteristic queries

(d) recall, uncharacteristic queries

Figure 7.16 The search performance of different hub selection methods for GOV-narrow queries in the large-sized network.

broken down into interests of finer granularity. Therefore, as shown in the results, interest-based initial hub selection used by hub selection based on clustered user modeling for characteristic queries only had slightly superior results for the WT10g-narrow query set than for the WT10g-broad query set, further demonstrating the effectiveness of our query clustering approach to modeling different user interests with various granularities.

The experimental results for the GOV-broad query set and the GOV-narrow query set using different hub selection methods in the large-sized P2P network are shown Figures 7.15 (a)-(d) and 7.16 (a)-(d) respectively. If we compare the results in the large-sized network with those in the medium-sized network, we can see that the advantage of hub selection based on clustered user modeling over other hub selection methods for characteristic queries was smaller in the large-sized network, and performance-based hub selection worked much better for uncharacteristic queries in the large-sized network than in the medium-sized network. These differences can be explained by the fact that the GOV queries selected for the query sets used in the large-sized network are better-defined and require a smaller percentage of the hubs to cover most relevant contents even for uncharacteristic queries, making it easier to select hubs based on the hubs' performance measured by the number of top-ranked documents each hub returned. Despite the changes in the performance of several hub selection methods, our proposed method of hub selection based on clustered user modeling still consistently yielded one of the best performances compared with other hub selection methods in the large-sized network.

To conclude, as demonstrated in the experimental results, hub selection based on clustered user modeling was effective and robust for various sets of queries, network sizes, and content granularities, thanks to its ability to adaptively model diverse user interests and apply different search strategies to different types of queries in optimizing the overall search performance. Because the evolution of hub-consumer topology depends on hub selection conducted by consumers, the consistently good performance of hub selection based on clustered user modeling gives us confidence in the effectiveness of hub-consumer topology evolution.

7.4 Summary

Although the evaluation in Chapter 5 demonstrates that even without carefully controlled content placement and topology evolution, our network search model is able to provide a better combination of accuracy and efficiency than existing common alternatives for full-text federated search, the results and analysis in this chapter show that the network evolution model is still desired for the effectiveness, robustness, and scalability of federated search in peer-to-peer networks.

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

Chapter 8

CONCLUSION

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

8.1 Summary

The work discussed in this dissertation 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, efficient, robust and scalable full-text federated search. The components of query routing, document retrieval, and result merging required for federated full-text ranked retrieval are incorporated in a *network search model*, which utilizes the functionalities and search-enhancing properties of the network overlay model to optimize search performance. The problem of constructing the proposed network overlay dynamically and autonomously is tackled with a *network evolution model*, which enables effective, efficient, and scalable topology evolution in decentralized, open-domain environments.

The network overlay model defines the functionalities and organization of peers in a P2P network to support full-text federated search. Peers that share or request information (providers or consumers) are located at the lower level of the network, free from the responsibilities of directing unrelated query traffic. Peers that provide regional directory services (hubs) are located at the upper level to form the backbone of the network, sharing all the necessary responsibilities related to search and topology evolution. Connections between peers are distinguished by the functionalities of the peers they link and the purposes they serve, so that peers can be organized to achieve desired content distribution and navigability by taking advantage of interest-based locality, content-based locality, and small-world properties. The defined network overlay is revolutionary in its effective incorporation of multiple search-enhancing network properties in a single framework, which weaves otherwise disorganized peer connections into an integrated platform that actively supports more effective, efficient, and robust federated search.

The network search model provides a comprehensive full-text federated search mechanism which includes new developments for each main component of federated search from resource representation to resource selection to result merging. First, to target the unique characteristic of resource selection in P2P networks due to multiple-hop query routing, we define the concept of a neighborhood and an exponentially decayed resource representation to effectively describe the contents reachable along each path at the hub level, and propose a resource selection method based on search radius-dependent neighborhood descriptions for each hub to select neighboring hubs for query routing. This approach has been proven to provide more effective resource location than the commonly used resource selection based on descriptions of direct neighbors or non-decayed neighborhood descriptions. Second, to enable each hub to learn its provider selection threshold autonomously without manual heuristic threshold tuning, we develop several unsupervised threshold learning methods that utilize the pseudo relevance feedback provided by result merging, and use a hybrid approach to combine their complementary strengths, which is able to automatically find the optimal threshold in consideration of both accuracy and efficiency. Third, in addition to effectively distinguishing different user interests to better establish and take advantage of interest-based locality, our approach of user modeling for resource selection of hubs by consumers is the first to recognize the need to use different search strategies for different types of queries (e.g., persistent, long-term versus transient, short-term), and to support it for optimizing the overall search performance. Fourth, by extending the Kirsch's algorithm for result merging to use substitute corpus statistics, accurate relevance-based document rankings can be generated without requiring global corpus statistics.

The network evolution model dynamically and autonomously constructs the network overlay described in the network overlay model in order to best support the network search model by providing desired search-enhancing network properties. It is designed to work effectively and efficiently for open-domain, unstructured text contents without relying on central coordination or control. Furthermore, the algorithms developed for the network evolution model also put extra effort on load balancing for the benefit of robustness and scalability. Specifically, content-based locality is established by using adaptive clustering that dynamically adjusts the granularity of the content area covered by each hub by cultivating multiple sub-clusters of providers within each hub's content-based cluster. Content-based small-world properties are the product of hubs periodically adjusting local and long-range hub connections based on limited local information and a power-law distribution of content similarity, which enables good network navigability without requiring global knowledge of the network. By observing resource selection of hubs conducted by consumers based on user modeling and adjusting connections accordingly, the network can obtain interest-based locality with few additional costs.

The detailed descriptions of the network overlay, search, and evolution models are accompanied by extensive experiments and analysis using two large-scale P2P testbeds created based on TREC test collections of real-content documents. Comprehensive evaluations measure the performance of each main component in the framework and compare it with existing common alternatives. Experimental results provide strong empirical evidence for the effectiveness of the approaches proposed in this dissertation for full-text federated search in P2P networks.

8.2 Contributions

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

The objective of this dissertation is to study federated search in P2P networks from an information retrieval perspective, and to develop new techniques to complement existing approaches in P2P networks that are mostly only applicable to limited domains. Particularly, we aim at providing comprehensive full-text ranked retrieval capability for federated search of text digital libraries in P2P networks. Our development offers one of the first sets of practical solutions to enable full-text federated search in P2P networks, which can be deployed *now* for networks of at least a few tens of thousands of small- and medium-sized digital libraries. It not only broadens the application territory of sophisticated information retrieval techniques, but also expands the use of federated search in P2P networks to more domains.

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

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

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

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

In addition to the models, the dissertation also provides valuable resources for the evaluation of federated search in large-scale P2P networks with realistic settings. Two P2P testbeds with large numbers of text collections and queries have been developed and used for evaluating existing and new approaches to full-text federated search and providing useful guidance for future research. Both testbeds have been published in a form that allows them to be used by other researchers.[27] Hopefully, our effort of building a useful evaluation platform with a flavor of large-scale P2P environments in the real world will benefit future research in this field.

8.3 Future Work

To minimize the computation and bandwidth usage of peers with limited resources, the network overlay defined in the proposed framework relies on hubs at the upper level of the network to act as communication gateways so that leaf peers (providers and consumers) don't need to connect directly among themselves. However, some lightweight communications at the lower level of the network may greatly improve search performance without significantly increasing costs. For instance, similar to content-based clusters formed by providers with similar contents, consumers

[27] http://www.cs.cmu.edu/~jielu/testbed.html

having similar interests can form interest-based clusters and share their user models within each cluster so that the performance of resource selection conducted by consumers can be improved collectively. There has also been some recent work on distributed collaborative filtering in P2P networks, in which the goal is to recommend other items to a user based on the items he/she previously downloaded (Wang et al. 2005). It would be interesting to extend our framework to incorporate collaborative search and filtering in P2P networks.

The proposed framework for full-text federated search in P2P networks currently includes models focusing primarily on search accuracy measured in precision/recall and efficiency measured in the percentage of the network reached for query messages. However, the framework is sufficiently flexible to allow more factors to be considered in addition to accuracy and efficiency in measuring search performance. For example, a utility-based approach can be used in the network search and evolution models to replace the similarity-based approach in ranking resources or establishing connections. The utility function can combine content-dependent features that we use for our work, as well as content-independent but resource-dependent features such as authority, reliability, response time, latency, and monetary cost, etc. The decision-theoretic model proposed in (Nottelmann and Fuhr 2007) provides such an approach for decentralized query routing, which may be extended to network evolution as well. This development will certainly make full-text federated search more practical and useful in a wider range of distributed environments.

The algorithms in our models assume that the document collection at each information provider is moderately sized and contains relatively homogeneous contents in a small number of topics. However, real applications may include a few giant information sources that provide a much larger number of documents in various topics and act more like large commercial Web search engines. Because resource selection without very accurate size normalization is unlikely to work well when comparing resource descriptions that vary greatly in the magnitudes of their vocabulary sizes and term frequencies, the search model needs to be extended in order to incorporate giant information providers. Different search strategies might be applied to typical versus giant information providers, and techniques developed for data fusion might be adapted to take advantage of the usually high degree of overlap between the document collections of giant information providers. Resource descriptions of giant providers might also be compared at the same level as neighborhood descriptions that aggregate content information of small providers. For network topology evolution, the approaches briefly described in Section 6.1.5 (pages 98-99) might be used for a giant information provider to establish content-based and/or popularity-based connections to multiple hubs to facilitate more effective and efficient search.

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

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

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

As described in Chapter 5 and Chapter 7, our evaluation on full-text federated search in P2P networks uses real-world document collections and queries. Due to the difficulty of doing research on large-scale P2P networks in an academic environment without thousands of networked computers and real users, the properties of the underlying physical network layer and the interactions between logical protocol layer and physical network layer are largely ignored. Because the simplification of network settings and peer behaviors might disguise problems that could emerge in the real world, evaluation of our models using a large real-world P2P application with real users is needed. In addition, the performance of search on large (huge) digital libraries remains to be studied in order to provide a more comprehensive evaluation of full-text federated search in P2P networks.

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.

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

Bailey P, Craswell N and Hawking D (2001) Engineering a multi-purpose test collection for web retrieval experiments. *Information Processing and Management*.

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

Bender M, Michel S, Triantafillou P, Weikum G and Zimmer C (2006) P2P content search: Give the Web back to the people. In *Proceedings of the 5th International Workshop on Peer-to-Peer Systems (IPTPS2006)*.

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, Lu Z and Croft W B (1995) Searching distributed collections with inference networks. In *Proceedings of the 18th 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.

Castiglion R and Melucci M (2007) An evaluation of a recursive weighing scheme for information retrieval in peer-to-peer networks. Submitted to *the 29th European Conference on Information Retrieval Research (ECIR 2007)*.

Caverlee J, Liu L and Bae J (2006) Distributed query sampling: A quality-conscious approach. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Clarke C, Craswell N and Soboroff I (2004) Overview of the TREC 2004 Terabyte Track. In *Proceedings of the 2004 Text Retrieval Conference*.

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*.

Craswell N, Bailey P and Hawking D (2000) Server selection on the World Wide Web. In *Proceedings of the 5th ACM Conference on Digital Libraries.*

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*.

French J, Powell A, Callan J, Viles C, Emmitt T, Prey K and Mou Y (1999) Comparing the performance of database selection algorithms. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Gao J (2004) A distributed and scalable peer-to-peer content discovery system supporting complex queries. Ph.D. thesis, School of Computer Science, Carnegie Mellon University.

Glance N (2001) Community search assistant. In *Proceedings of the 2001 International Conference on Intelligent User Interfaces.*

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*.

Gravano L, García-Molina H and Tomasic A (1999) GlOSS: Text-source discovery over the Internet. *ACM Transactions on Database Systems*, 24(2).

Hawking D and Thistlewaite P (1999) Methods for information server selection. *ACM Transactions on Information Systems*, 17(1).

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.

Ipeirotis P and Gravano L (2002) Distributed search over the hidden web: Hierarchical database sampling and selection. In *Proceedings of the 28th International Conference on Very Large Databases (VLDB).*

Ipeirotis P and Gravano L (2004) When one sample is not enough: Improving text database selection using shrinkage. In *Proceedings of the ACM-SIGMOD International Conference on Management of Data.*

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.

Klampanos I, Poznanski V, Jose J and Dickman P (2005) A suite of testbeds for the realistic evaluation of peer-to-peer information retrieval systems. In *Proceedings of the 27th European Conference on Information Retrieval Research (ECIR 2005)*.

Kleinberg J (2000) The small-world phenomenon: an algorithmic perspective. In *Proceedings of 32nd ACM Symposium on Theory of Computing*.

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

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-based digital libraries in hierarchical peer-to-peer networks. In *Proceedings of the 27th European Conference on Information Retrieval Research (ECIR 2005)*.

Lu J and Callan J (2006a) Full-text federated search of text-based digital libraries in peer-to-peer networks. *Journal of Information Retrieval, Volume 9, Number 4.*

Lu J and Callan J (2006b) User modeling for full-text federated search in peer-to-peer networks. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

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*.

Nottelmann H and Fuhr N (2007) A decision-theoretic model for decentralized query routing in hierarchical peer-to-peer networks. In *Proceedings of the 29th European Conference on Information Retrieval Research (ECIR 2007)*.

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

Ramanathan M, Kalogeraki V and Pruyne J (2002) Finding good peers in peer-to-peer networks. In *Proceedings of 16th International Parallel and Distributed Processing Symposium.*

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)*.

Shokouhi M, Zobel J, Scholer F and Tahaghoghi S (2006) Capturing collection size for distributed non-cooperative retrieval. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

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 (2004a) The effect of database size distribution on resource selection algorithms. Distributed Multimedia Information Retrieval. LNCS 2924, Springer.

Si L and Callan J (2004b) Unified utility maximization framework for resource selection. In *Proceedings of the 13rd International Conference on Information Knowledge Management (CIKM 2004)*.

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*.

Tang C, Dwarkadas S and Xu Z (2004) On scaling latent semantic indexing for large peer-to-peer systems. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.*

Tang C and Dwarkadas S (2004) Hybrid global-local indexing for efficient peer-to-peer information retrieval. In *Proceedings of the 1st Symposium on Networked System Design and Implementation.*

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*.

Viles C and French J (1995) Dissemination of collection wide information in a distributed information retrieval system. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Voorhees E, Gupta N and Johnson-Laird B (1995) Learning collection fusion strategies. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

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*.

Wang J, Reinders M, Lagendijk R and Pouwelse J (2005) Self-organizing distributed collaborative filtering. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

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

Wen J, Nie J and Zhang H (2002) Query clustering using user logs. *ACM Transactions on Information Systems*, 20(1).

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).*

Yuwono B and Lee D (1997) Server ranking for distributed text retrieval systems on Internet. In *Proceedings of the 5th International Conference on Database Systems for Advanced Applications.*

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).*