Federated Search of Text Search Engines in Uncooperative Environments

Luo Si

Thesis Proposal

Language Technology Institute
School of Computer Science
Carnegie Mellon University
lsi@cs.cmu.edu

Thesis Committee:
Jamie Callan (Carnegie Mellon University, Chair)
Jaime Carbonell (Carnegie Mellon University)
Yiming Yang (Carnegie Mellon University)
Luis Gravano (Columbia University)

July, 2004

TABLE OF CONTENTS

Chapter 1: Introduction

Web contains huge amount of information. It was estimated that Web is composed of about 9 million sites by various information providers (OCLC, 2002)[1]. Conventional search engines such as Google or AltaVista can access some of these contents and make the information visible to the users. This type of information is called "*visible Web*" contents. They can be automatically copied and collected, and thus the conventional search engines can take the advantage to crawl all the visible Web contents and copy them into a single centralized database. However, this is not true for all the contents on the Web. Web is more diverse and there is much more to explore beyond these visible Web contents.

"*Hidden Web*" contents (also called "invisible" or "deep" Web contents) are information that cannot be accessed by the conventional search engines such as Google or AltaVista. The information providers that contain these hidden Web contents are called hidden information sources. There are several cases that the contents in the hidden information sources can not be crawled: i) the owners of the information sources only grant public access of the contents through the corresponding search interfaces, and the users are not allowed to automatically download and collect all the information; ii) although the contents of some information providers can be crawled by conventional search engines, the information is updated too frequently by the information providers and it is hard to be crawled immediately by conventional search engines to reflect the changes; and iii) the access of some information sources is subject to fees or subscriptions (this kind of information only consists about five percent of the hidden Web as indicated by previous study (Bergman, 2001)). Thus, the distinction between the visible Web information and the hidden Web information that can only be accessed through the search engines of hidden information sources determines the solutions of the corresponding search problems.

The research in this dissertation targets the problem of searching useful information among the hidden Web, where the contents can only be accessible through the search interfaces of hidden information sources. In a hidden information source, the contents are hidden from web crawlers but the information source itself is visible and can be accessed via its search engine. Therefore, the two terms of hidden information source and information source put emphasis on different aspects of the same object, and they are used in this dissertation interchangeably.

The size of the hidden Web contents has been estimated to be 2-50 (Sherman, 2001) larger than the size of visible Web contents. Although the exact size of hidden Web contents is still unknown, previous studies indicated that hidden Web should be larger (if not much larger) than visible Web and the hidden Web contents receive more monthly traffic (Bergman, 2001) than the visible Web contents. A large amount of the hidden web is made up of the contents in millions of specialized information sources. These hidden information sources cover a wide range of topics, for example, the ACM digital library[2] contains hundreds of scientific journals and conference proceedings, and the GPO[3] (U.S. Government Printing Office) portal connects to a large amount of government agency databases. Many these hidden information sources are maintained by domain experts and are updated frequently. Therefore, the contents on hidden Web are valuable and the solution of searching the hidden Web contents distributed among many hidden information sources is an important research topic.

---

[1] http://wcp.oclc.org/stats.html

[2] http://portal.acm.org/portal.cfm

[3] http://www.gpoaccess.gov/databases.html

1.1 Federated search vs ad-hoc retrieval

Ad-hoc information retrieval solves the search problem of visible Web contents and is utilized by most conventional search engines. It is a single database retrieval model that copies and indexes all the searchable information into a local centralized database. This solution is effective when the information providers expose and share their contents. However, as it is indicated above that this is not true for the hidden Web, where the information is distributed among hidden information sources and cannot be crawled into a single centralized database.

Federated search provides the solution of the search problem for hidden Web contents. It builds a single search interface, which connects all the search engines of hidden information sources and provides a uniform entrance to the hidden Web contents. In contrast to the ad-hoc retrieval, federated search is a multi-database search model. In this framework, the searching process is conducted in the search engines of individual information sources and it is not necessary to construct the centralized database containing all the searchable information, which reflects the distributed location and control of the hidden Web contents. However, this solution is composed of several components and some other problems arise. More specifically, three main sub-problems need to be addressed in the federated search task.

-*Resource description*. The contents of each hidden information source must be learned and represented in a suitable form for further use. Previous research usually represented the hidden information sources by their content statistics such as vocabulary or term frequencies (Gravano et al., 1994)(Callan et al., 1995b)(Voorhees et al., 1995)(Viles & French, 1995)(Ipeirotis & Gravano, 2004);

-*Resource selection*. Given an information need (a query) and the resource descriptions, a set of relevant information sources must be selected to do the search (Gravano et al., 1994)(Voorhees et al., 1995)(Callan et al., 1995b)(Viles & French, 1995)(Fuhr, 1996)(Fuhr, 1999)(Craswell, Hawking & Thistlewaite, 1999)(French et al., 1999)(Callan, 2000)(Ipeirotis & Gravano, 2002)(Nottelmann & Fuhr, 2003b); and

-*Results merging*. All the returned ranked lists from the selected hidden information sources can be merged into a single list before it is presented to the end user (Voorhees et al., 1995)(Kirsch, 1997)(Craswell, Hawking & Thistlewaite, 1999)(Si & Callan, 2002a).

The solutions of the above sub-problems serve as components of different federated search applications.

1.2 Applications of federated search

Similar with the case for visible Web contents where users might be interested in different types of tasks such as browsing or retrieval (Baeze-Yates & Ribeiro-Neto, 1999), users of federated search systems may prefer different types of solutions.

Invisible-web.net[4] provides structure guided browsing of hidden Web contents in hidden information sources. It collects the resource descriptions of the hidden information sources and builds hierarchy of classes that group the hidden information sources with similar topics together. Users can browse a specific favorite hidden information source or can simply browse the whole hidden information source space for broader interesting references.

This browsing model of hidden information sources works well when the users are not interested in posing a specific query and are willing to spend some time exploring a wide range of the hidden information source space. However, when the users have explicit information needs expressed by text queries and want to find the relevant information directly to save searching efforts, more complex applications such as the information source recommendation system or the federated document retrieval system are necessary.

[4] http://www.invisible-web.net

An information source recommendation system can be seen as an extended solution of the browsing model of invisible-web.net. It recommends most relevant information sources to users' information needs of text queries. This system is very useful when the users want to browse the selected information sources by themselves instead of asking the system to retrieve relevant documents automatically. It is composed of two components as resource description and resource selection.

Federated document retrieval is a more sophisticated search application than the information source recommendation system. It selects relevant information sources for users' queries as what the information source recommendation system does. Furthermore, users' queries are forwarded to the corresponding selected information sources and the returned individual ranked lists are merged into a single list to present to the users. Therefore, all the three components of federated search: resource description, resource selection and results merging are employed in the federated document retrieval application.

The solutions of the three sub-problems of federated search task are highly influenced by different environmental characteristics. In a small local area network such as small company environments, the information providers may cooperate to provide corpus statistics or use the same type of search engines (Callan, 2000)(Gravano et al., 1997)(Si et al., 2002b). On the other side, in a wide area network such as very large corporate environments or on the Web there are many types of search engines and it is difficult to assume that all the information providers can cooperate as they are required (Si & Callan, 2002a)(Si & Callan, 2003a). Even they are willing to cooperate in these environments, it may be hard to enforce a single solution for all the information providers. For example, a word in the stop word list of one information source which exists in almost every document may be quite indicative and cannot be thrown away for another information source and vice versa. Furthermore, it is often hard to detect whether information sources provide the correct information as they are required.

There has been considerable early research focused on the *cooperative environments* such as the local area network (Voorhees et al., 1995)(Viles & French, 1995)(Callan, 2000)(Gravano et al., 1997)(Kirsch, 1997). A recent trend is to study the more complex situation of *uncooperative environments* such as the wide area network or the Web (Si & Callan, 2002a)(Ipeirotis & Gravano, 2002)(Si & Callan, 2003a)(Si & Callan, 2003b)(Bergholz & Chidlovskii, 2004), and this is the focus of this dissertation.

1.3 The contribution of this dissertation

Federated search is a popular research topic as there has been considerable research on different sub-problems. Most of them are concentrated on the cooperative environments and relatively few can be applied in the uncooperative environments. In this dissertation, we will first describe new research to address the three main sub-problems of federated search in uncooperative environments separately, and then propose a single probabilistic framework to integrate them together.

For resource description, previous research mainly focused on how to acquire corpus statistics of hidden information sources such as the vocabulary or term infrequencies (Callan & Connell, 2001)(Gravano et al., 1997). However, in either the task of the information source recommendation system (to recommend information sources that contain as many relevant documents as possible) or the federated document retrieval system, the ultimate unit a user evaluates is a document. Therefore, it is necessary to estimate the characteristics of individual documents among the hidden information sources and is necessary to know how many documents each hidden information source contains (Liu et al., 2001). For example, the resource selection algorithms need to estimate the number of relevant documents each information source contains and thus the information source size estimates are very important to adjust (normalize) the information source selection scores (Si & Callan, 2003a). However, information source size estimation is a major unsolved problem until now. Previous research (Liu et al., 2001) required huge amount of communication costs to estimate information source sizes especially for large information sources. In this dissertation, a much more efficient Sample-Resample algorithm is proposed for this problem. This method utilizes sampled documents from query-based sampling and calculates the information source size estimates by sending resample queries and scaling the sampled document size with the ratio of document frequencies of

these queries in the whole information source to the document frequencies in the sampled documents (Si & Callan, 2003a).

For resource selection, most prior research followed the "big document" strategy, which treats the information sources as "big documents" and calculates the similarities between user queries and the "big documents" to make the selection decision (Yuwono & Lee, 1997)(Callan, 2000)(Craswell, 2000)(French et al., 1999)(Xu & Croft, 1999)(Si et al., 2002b). However, as it is pointed out above that the ultimate units should be documents, the "big document" approach loses the boundaries between individual documents by simply treating an information source as a large document (Si & Callan, 2003a). This problem is serious as empirically studies have shown that the "big document" resource selection algorithms do not normalize the lengths of information sources well. They often have strong disfavor bias against either small hidden information sources or large hidden information sources and thus miss large amount of relevant documents in these information sources (Craswell, 2000)(Si & Callan, 2003a)(Si & Callan, 2003c). In contrast, a resource selection algorithm is proposed to explicitly estimate relevant document distribution across available information sources for information source recommendation application by making full use of the information source size estimates and the content representations from the resource description component (Si & Callan, 2003a). This approach is not only more theoretically solid but also provides better empirical results.

Results merging is the last step for a federated document retrieval system, which merges the individual ranked lists from the selected information sources into a single final ranked list. It is a difficult job especially in uncooperative environments as different hidden information sources may use different retrieval algorithms or have different corpus statistics. Previous results merging algorithms either used heuristic formula to calculate final comparable scores or assumed each hidden information source to return query term frequencies of the retrieved documents for computing consistent scores across information sources (Callan et al., 1995b)(Voorhees et al., 1995)(Kirsch, 1997). However, these methods are not very effective or require cooperation that is not valid in uncooperative environments. A Semi-Supervised Learning (SSL) results merging algorithm is proposed instead. It applies a centralized retrieval algorithm on the sampled documents acquired by query-based sampling. The sampled documents with both information source independent scores and information source specific scores (returned from selected information sources) are used as training data. Linear models are learned from the training data to transform information source specific scores to corresponding information source independent scores. Furthermore, the linear models are applied on all the returned documents to approximate the comparable information source independent scores, and thus the final result list can be obtained with these source independent scores. When there is not enough training data in the sampled documents, a variant of the SSL algorithm downloads a minimum number of documents "on the fly" to create additional training data (Si & Callan, 2002a)(Si & Callan, 2003b). The SSL algorithm has been shown to produce rather accurate final document rankings with small costs.

Preliminary experiments have shown that all the proposed new methods are effective for the corresponding sub-problems. However, the relationship between the sub-problems has not been well studied. Some prior research pointed out that a good resource selection algorithm for an information source recommendation system may not work well for a federated document retrieval system (Craswell, 2000)(Si & Callan, 2003a). The behavior is formalized in this research as the inconsistency between the high-recall goal of information source recommendation system and the high-precision goal of the federated document retrieval system. Based on this observation, a single probabilistic model is proposed to integrate all the separate components into a unified utility maximization framework. Specifically, when used for information source recommendation, the framework is optimized for high-recall, and when used for federated document retrieval, it is optimized for high-precision. This framework gives a more theoretical and unified solution of the federated search task. Thorough empirical studies have shown that this unified framework produces more accurate results for both the information source recommendation application and the federated document retrieval application; detail analysis indicates that the power of this framework comes from explicitly modeling and optimizing the utilities of different applications.

One key contribution of this dissertation shows the successful utilization of a *centralized sample database* (CSDB). Some previous research (Craswell, 2000)(Si & Callan, 2002b) discarded the downloaded documents of query-based sampling after the resource descriptions were built. However, these documents

can also serve another purpose. The sampled documents can be combined into a single searchable database as the centralized sample database (Ogilvie & Callan, 2001a). This centralized sample database can be imagined as a representative sample of a single global database if all the documents from all the hidden information sources could be crawled together (this is not true in federated search environments). The vocabulary and frequency information in the centralized sample database is expected to approximate the vocabulary and frequency patterns across the complete set of hidden information sources. Ogilvie and Callan's work (Ogilvie & Callan, 2001a) was one of the earliest research to utilize centralized sample database. They used centralized sample database for query expansion in a federated document retrieval system, which did not work well due to the failure of local context analysis on the limited size of sampled documents and the extra difficulties of results merging introduced by query expansion. This dissertation shows that centralized sample database can be utilized in many other problems of federated search system and proves its success with empirical evidences. Specifically: i) document frequency statistics on the sampled documents are used for information source size estimation; ii) the probabilities of relevance of all the documents among the hidden information sources are inferred from the probabilities of relevance of the sampled documents in the centralized sample database; and iii) the documents in the centralized sample database serve as training data to learn query-specific and information source specific linear models to transform information source dependent scores into information source independent scores in results merging.

The second key contribution of this dissertation is to recognize the insufficiency of the "big document" resource selection approach and explicitly estimate the probabilities of relevance for all (mostly unseen) documents in federated search environment to achieve effective results of various applications. Previous "big document" resource selection algorithms only consider hidden information sources as big documents and rank the information sources by their similarities with user queries, which can be misleading as a hidden information source with high similarity with a user query may contain very little amount of relevant documents due to its small size or very skewed word distribution (e.g., only very few documents contain a large number of query terms). In contrast, our new approach recognizes that it is necessary to estimate the probabilities of relevance of all the documents across the available hidden information sources for accurate resource selection decision. Furthermore, the unified utility maximization framework described above incorporates the estimated probabilities of relevance of all the documents. This framework can be optimized to obtain good results of various applications accordingly.

Federated search is a hot research topic with many difficult problems especially in uncooperative environments. This dissertation advances the state-of-the-art in the main sub-problems of federated search separately and also makes an important step forward to propose formal framework to integrate the new research together and optimize various goals of different federate search applications. The new framework is more theoretically solid and is open to be extended for specific consideration of different real operational applications (e.g., considering the retrieval effectiveness of individual information sources). This indicates that the new research in this dissertation not only provides more solid theoretical basis for federated search task but is also more promising to be utilized in real operational environments.

Chapter 2: Modeling Federated Search Problems

This chapter first discusses different choices of federated search environments to evaluate federated search algorithms. The best approach is to test these algorithms in real operational environments. An on-going federated search project is introduced and the possibilities and limitations of evaluation with this real world application are discussed. Furthermore, the characteristics of different federated search environments are analyzed and several commonly used federated search testbeds are introduced to model these federated search environments. Experimental methodology of modeling multiple types of search engines is also addressed. Finally, this chapter briefly introduces experimental metrics of federated search applications.

2.1 Data in the real world application

The best evaluation methodology is to study the effectiveness of different algorithms in real world applications. FedStats [5] is a government web site. It provides searching service of federal agency information sources that contain official statistic reports. The existing search solution copies and indexes the contents of more than one hundred information sources into a single centralized database. However, the centralized contents are not synchronous with the contents of individual information sources because the efforts of crawling and updating all the contents very frequently are not affordable. Thus, a federated search solution was requested and an on-going project is conducted in Carnegie Mellon University to develop a prototype for the FedStats Web site.

FedStats is a real world application and it is an ideal candidate to evaluate federated search algorithms. One problem lies in that not enough relevant judgments are available to tune the behavior of the federated search system or to evaluate its effectiveness. For example, the evaluation of a resource selection algorithm requires the knowledge of the distribution of all relevant documents across the information sources, which is very difficult to acquire in this type of environment where we do not have full control of every component. Although it is hard to conduct complete and thorough experiments, user studies will be arranged in the future work to report possible evaluations with this real federated search system.

FedStats provides the federated search solution for information sources within a large organization. Another example of federated search solution within a large company is the West[6] system where thousands of legal, financial and news text information sources are accessed by up to a quarter-million users each day (Conrad, 2002). The FedStats and the West systems are similar to each other in several aspects: i) the information is scatted among different information sources due to either the maintenance and policy issues or technique difficulties; ii) the information sources are often created and maintained by different providers, so the overlapping among their contents tend to be rather low; iii) the information sources contain a relatively large number of documents, for example, the number of documents in the information sources of FedStats system falls into the range of a couple of thousand to more than one hundred thousand; iv) the contents of the information sources are often carefully written and edited by professional writers, so the qualities are higher than that of less formal documents such as arbitrary web pages; and v) the contents of the information sources are often focused on several specified topics (e.g., agency reports of FedStats or legal, financial and news of West system), which is different from the much general contents of an open-

[5] http://search.fedstats.gov

[6] http://www.westlaw.com

Table 1. Testbed statistics of Trec123_100Col and Trec4_kmeans.

Name	Query Count	Size (GB)	Number of Docs			Megabytes (MB)		
			Min	Avg	Max	Min	Avg	Max
Trec123_100Col	100	3.2	752	10782	39713	28.1	32	41.8
Trec4_kmeans	50	2.0	301	5675	82727	3.9	20	248.6

Table 2. Query set statistics.

Name	TREC Topic Set	TREC Topic Field	Average Length (Words)
Trec123_100Col	51-150	Title	3.1
Trec4_kmeans	201-250	Description	7.2

domain Web search engine. These similarities of the FedStats and West systems can be seen not only as the common characteristics of federated search solutions in large organization or large company environments but also for federated search systems of domain-specific hidden Web.

The scale of federated search systems for open domain hidden Web can be much larger than the scale of federated search system in large organizations or companies. Furthermore, it can be imagined that the contents of general purpose hidden Web federated search systems are much more diverse. This characteristic should serve as a guidance to build testbeds of simulating the federated search environments of open domain hidden Web.

Conducting experiments with real world applications is the most desired way to evaluate federated search algorithms. However, it is hard to conduct user studies in very large scale or to obtain full control of these systems. An alternative way is to take advantage of existing large text collections with thorough relevance judgments and simulate the operational environments of federated search systems.

2.2 Simulating real world applications with TREC Data

It is discussed in Section 2.1 that evaluation of federated tasks requires sufficient number of information sources, enough queries and thorough relevance judgments. It is often hard to meet all these requirements in user studies of real world applications. In contrast, existing large text collections such as TREC provide us the opportunity to simulate real world federated search environments.

The TREC conference[7] (Text REtrieval Conference) (Harman, 1995) is co-sponsored by the National Institute of Standards and Technology (NIST) and the Information Technology Office of the Defense Advanced Research Projects Agency (DARPA). The goal of this conference is to encourage research in information retrieval for text applications by providing a large test collection including large corpora with sufficient queries and relevance judgments (Harman, 1995). The creation of TREC data provided the first opportunity of conducting federated search experiments to simulate the environments with large numbers of rather diverse information resources, distributed geographically and maintained by many parties. For example, the database merging track of the TREC-4 conference was one of earliest exploration of federated search task (Harman, 1995).

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

2.2.1 Simulation with TREC news/government data

TREC news/government documents data is concentrated on relatively narrow topics. The contents of these documents are well written and organized by professional writers, which resemble the contents of federated search systems such as FedStats or West system. This type of data provides a good opportunity to simulate the environments of large organizations or domain-specific hidden Web.

Partitioning TREC news/government documents is the most commonly used strategy to create federated search testbeds (Lu et al., 1996)(Xu & Callan, 1998)(French et al., 1999)(Hawking & Thistlewaite, 1999)(Callan, 2000)(Si & Callan, 2002a)(Ipeirotis & Gravano, 2004). This approach has several advantages: i) news/government documents are much more similar to the contents provided by a topic-oriented information source than an arbitrary web page; ii) the above testbeds are composed of O(100) information sources which contain thousands of documents by average; this is more realistic compared with testbeds of only several information sources (Fox et al., 1992)(Yuwono & Lee, 1997) or testbeds of about one thousand small information sources containing only 250 documents by average (Craswell, 2000); and iii) a large body of previous research has reported experiment results on the testbeds of TREC news/government collections (Xu & Callan, 1998)(French et al., 1999)(Hawking & Thistlewaite, 1999)(Callan, 2000)(Si & Callan, 2002a), which provided baselines to compare the effectiveness of the new proposed algorithms. The relatively specific and narrow topics in TREC news/government data, the relatively large number of information sources containing thousands of documents and the thorough baseline results make the TREC news/government data a good candidate to simulate the federated search environments of large organizations or domain specific hidden Web. However, it has disadvantages to simulate the federated search environments such as the open domain hidden Web. More specifically, i) the federated search environments of open domain hidden Web are expected to have more diverse contents; and ii) the federated search environments of open domain hidden Web tend to have larger number of hidden information sources.

A large number of previous federated research split the large TREC collections containing news/government documents into smaller information sources and evaluated their work with these testbeds (Lu et al., 1996)(Xu & Callan, 1998)(French et al., 1999)(Hawking & Thistlewaite, 1999)(Callan, 2000)(Si & Callan, 2002a)(Ipeirotis & Gravano, 2004).

Two types of testbeds used in this dissertation were created to organize the information sources by different criteria:

(i) Trec123_100Col: One hundred information sources were created from TREC CDs 1,2,3. They were organized by source and publication date, and are some somewhat heterogeneous. 100 short queries extracted from the title fields of TREC topics 51-150 were associated with this testbed (French et al, 1999)(Callan, 2000)(Si & Callan, 2002a).

(ii) Trec4_kmeans: One hundred information sources were created from TREC 4 data. A k-means clustering algorithm was used to automatically cluster the information sources by topic. The contents of the information sources are homogenous and the word distributions are rather skewed. 50 longer queries were created from the description fields of the TREC topics 201-250 (Xu & Croft, 1999)(Si & Callan, 2002a).

The characteristics of these two testbeds are shown in Table 1 and the characteristics of their corresponding queries are shown in Table 2. Many other testbeds constructed out of the TREC news/government data are also available. For examples, Fox et al. (Fox et al., 1992) used 5 collections from TREC CD 1 and French et al. (French et al., 1998) partitioned TREC CDs 1,2,3 with finer granularity into 236 collections. Among these testbeds, Trec123_100Col is the most widely used (French et al, 1999)(Callan, 2000)(Si & Callan, 2002a)(Nottelmann & Fuhr, 2003b) and there are many baseline results available for this testbed. So it is used in this dissertation. As Trec4_kmeans is organized by topics which provides another interesting type of partition instead of the source and date partition of TREC CDs 1,2,3, it is also utilized in this research.

Table 3. Statistics for the large databases.

Database	LDB1	LDB2	APall	WSJall	FRall	DOEall
Num of docs (x 1,000)	231.3	199.7	242.9	173.3	45.8	226.1
Size (MB)	665	667	764	533	492	194

It can be noted that Trec123-100Col testbed has a relatively uniform information source size distribution and trec4-kmeans has modest skewed source size distribution. Prior research in federated search often ignored the source size issues (Callan, 1995b)(Yuwono & Lee, 1997)(Xu & Croft, 1999). However, it is important as skewed source size distribution is possible in real federated search environments. So it is necessary to study the behavior of different algorithms with testbeds of skewed source size distributions. Furthermore, another interesting direction is to vary the relevant document distribution and study the effectiveness of different algorithms with this manipulation. Following these ideas, three more testbeds were created based on the Trec123-100Col testbed. Each of them contains many "small" information sources and two large information sources as described below (Si & Callan, 2003a).

Trec123-2ldb-60Col ("representative"): One hundred information sources in the Trec123-100Col testbed were sorted alphabetically. Every fifth source from the first one was merged into one large source called LDB1. Every fifth source from the second one was merged into another large source called LDB2. The other 60 sources were left unchanged. This testbed simulates the environment of bimodal source size distribution where large sources have about the same densities of relevant documents as small ones (as the two large information sources have much more documents, they still contain more relevant documents).

Trec123-AP-WSJ-60Col ("relevant"): The twenty-four Associated Press information sources in the Trec123-100Col testbed were collapsed into a large APall information source, while sixteen Wall Street Journal collections were merged into a large WSJall source. The other 60 small sources were not changed. This testbed simulates the environment of bimodal source size distribution where large sources have higher densities of relevant documents than small ones.

Trec123-FR-DOE_81Col ("nonrelevant"): The thirteen Federal Register information sources in the Trec123-100Col testbed were combined into a large FRall information source, while the 6 Department of Energy information sources were collapsed into a large DOEall information source. The left 81 information sources were unchanged. This testbed simulates the environment of bimodal source size distribution where large sources have lower densities of relevant documents than small ones.

The three representative, relevant and nonrelevant testbeds have both large information sources and small information sources. In contrast, a Tec123-10Col testbed was created which contains only 10 large information sources. The information sources in Trec123-100Col testbed were sorted alphabetically. Every tenth source from the first one was merged into the first new source. Every tenth source from the second one was combined into the second new source, and so on. (Si & Callan, 2003a)

2.2.2 Simulation with TREC web data

TREC also offers large collections of Web documents. The Web data can often be naturally divided into much larger number of information sources by more diverse information providers than the news/government data. Therefore, it has the advantage to simulate the federated search environments with large amount of information sources such as open domain hidden Web.

WT2g (Voorhees & Harman, 1999)(Craswell, 2000) collection contains about two gigabytes of crawled Web documents. These documents are from 956 Web sites such as http://migration.ucdavis.edu or http://learnathome.com. Therefore, the documents can be naturally partitioned into 956 information sources with respect to the Web sites. A larger Web collection is also available as WT10g (Craswell, 2000)(Lu & Callan, 2003), which contains about 10 gigabytes of crawled Web documents from 11,486 Web sites. This

Table 4. Statistics for TREC WT10g data.

Name	Query Count	Size (GB)	Number of Docs			Megabytes (MB)		
			Min	Avg	Max	Min	Avg	Max
WT10g	250	10.8	1	146	26505	0.004	0.91	98

larger Web collection provides better opportunity to simulate realistic experiments with larger number of hidden information sources. The characteristic of this testbed is shown in Table 4.

The TREC Web collections provide a natural way to partition the documents into information sources with respect to different Web sites(Craswell, 2000)(Lu & Callan, 2003). More information sources with more diverse contents could be created from the TREC Web collections than the TREC news/government data to better simulate operational environments such as open domain hidden Web. However, some important issues must be addressed before utilizing the TREC Web data. For example, most web sites contain very small amount of Web documents. It can be seen from Table 4 that the each Web site of WT10g contains only 146 documents by average, which is strongly against the intuition that most online information sources contain a large number of valuable topic-oriented documents.

Another general weakness of utilizing TREC Web data for federated search is that the crawled Web documents generally do not reflect the hidden Web contents. The contents of hidden information sources such as federal agency information sources of FedStats are often carefully written and edited by professional writers. However, the TREC Web corpus contains very noisy data such as personal homepages or even Web pages indicating that the pages are under construction and there is no useful information at all.

In summary, TREC Web collections have both advantages and disadvantages to be used for federated search evaluation. As they have been utilized in some previous research work (Craswell, 2000), experiments will be designed and conducted with this type of testbeds in the future work. More specifically, we would create testbeds with the TREC Web data to better simulate federated search environments such as open domain hidden Web.

2.3 Simulating multiple types of search engines

It is most likely that multiple types of search engines are utilized in uncooperative federated search environments. Three retrieval algorithms of a vector-space algorithm similar to SMART (Buckley et al., 1995), INQUERY (Turtle 1990) and Language Model (Lafferty & Zhai, 2001) were chosen in the experiments reported here as they are widely used in ad-hoc retrieval systems. It is possible to further vary the behavior of these search engines by choosing different stemming algorithms and different stopword lists. Note all these systems are effective search engines, and this is a hidden assumption in most part of this research. We do not include any bad search engines, but inaccurate search engines are common in real world environments. For example, the well-known PubMed [8] system uses unranked retrieval. This assumption of effective search engines is reexamined and relaxed in Chapter 7.

The vector-space retrieval algorithm uses SMART "lnc.ltc" weighting scheme (Buckley et al., 1995), where logarithmic version of term frequency and cosine normalization is used by both query and document representations while query representation utilizes logarithmic idf weight and document representation does not. The document scores fall into the range of [0.0 , 1.0].

The INQUERY algorithm (Turtle 1990)(Callan et al, 1992)(Callan, Croft & Broglio, 1995) is a retrieval model that adapts an Okapi term frequency normalization formula (Robertson & Walker, 1994) to rank the

[8] http://www.ncbi.nlm.nih.gov/entrez/query.fcgi

information sources. Formally, the belief of the j^{th} document according to the word q is expressed as follows:

$$T=\frac{tf}{tf+0.5+1.5*doclen_j/avg\_doclen} \tag{2.1}$$

$$I=\frac{\log(\frac{|d|+0.5}{df})}{\log(|d|+1.0)} \tag{2.2}$$

$$p(q|d_j)=b+(1-b)*T*I \tag{2.3}$$

where: tf is the number of occurrence of q in this document;
df is the number of documents that contain q;
|d| is the number of documents to be ranked;
$doclen_j$ is the number of words in this document;
avg_doclen is the average document length; and
b is the default belief, usually set to 0.4.

The belief P (Q | d_j) is obtained by combining scores P(q | d_j) from different term q. This can be achieved by different probabilistic operators modeled in the Bayesian inference network model. The INQUERY operators cover a wide range of Boolean, proximity and synonym operators. The detail can be found in (Turtle, 1990)(Callan, 2000). In this research, the belief P (Q | d_j) is calculated by a simple operator of averaging the belief of individual query words.

Language Model retrieval algorithm treats each document as a multinomial distribution of the words in the vocabulary. It measures the relevance of each document to a query as how likely the document can generate this query. Formally, the generation probabilities of documents are calculated as:

$$P(Q|d_j)=\prod_{q\in Q}(\lambda P(q|d_j)+(1-\lambda)P(q|G)) \tag{2.4}$$

where P (q | d_j) is the unigram language model for the j^{th} document. P (q | G) is the probability of the query term q in a global language model. The global language model can be obtained by combining all the documents together and calculating a single language model. P (Q | d_j) is usually a very small positive number and the logarithm of this value gives us a final score usually in the range of [-60 , -30].

All these three algorithms were implemented with the Lemur toolkit (Ogilvie & Callan, 2001b), and they were usually assigned to the information sources in a round-robin manner.

2.4 Federated search experimental metrics

Different federated search applications are associated with a variety of evaluation metrics. The evaluation in this dissertation mainly follow procedures established by previous research (Liu et al., 2001)(French et al., 1999)(Callan, 2000). For information source size estimation, the average value of absolute error ratio (AER) is used to measure the deviation of estimated information source size to the actual information source size (Liu et al., 2001)(Si & Callan, 2003a). For resource selection of information source recommendation system, the information source ranking of a specific resource selection algorithm is compared with that of a desired source ranking, which ranks information sources by the amount of relevant documents they contain for a user query. The smaller is the difference between the ranking of the specific

algorithm and that of the desired ranking, the more effective is the algorithm (French et al., 1999)(Callan, 2000). For federated document retrieval, the system searches the selected information sources and generates a final ranking list. The final ranking list is evaluated with the Precision at the top ranks (Xu & Croft, 1999)(Callan, 2000). More detail of these specific metrics is covered in later chapters. It can be noted that relevance judgment is required for the evaluation of information source recommendation system and federated document retrieval system. Especially, in information source recommendation system, all the relevant documents among the information sources for a specific query should be identified for the evaluation, which causes difficulties of evaluating real world applications as discussed in Section 2.1.

Chapter 3: Resource Representation

To acquire accurate resource descriptions of information sources is the first step for every federated search system. There are three problems that need to be addressed to get good resource descriptions: i) what types of resource descriptions are required in order to well accomplish our tasks; ii) for each type of resource description, how can it be obtained efficiently; and iii) in what kind of suitable form should the description be represented. Previous research on resource description was mainly focused on representing each information source by a description of its words and the word frequencies. This chapter introduces information size estimates as another type of resource description. Specifically, this chapter discusses why information size estimation is important in federated search task, proposes a new algorithm to calculate the information source size estimates more efficiently and evaluates this new method to prove its effectiveness and efficiency.

3.1 Previous research on resource description

This section shows the previous research on resource description from three aspects: resource description constructed by words and their occurrences, the centralized sample database constructed by sampled documents and the size estimates of information sources.

3.1.1 Resource description of words and their occurances

Many prior research on resource description represented each information source by a description of the words that occur in the information source, and the word frequencies (Gravano, 1994)(Gravano & García-Molina, 1995)(Callan et al, 1995b) or other statistics derived from the word frequencies such as the term weights (Gravano & García-Molina, 1995). This approach is possible to be extended for other indicative text features such as phrases or proper names.

The most desirable scenario to acquire this type resource description of information sources is when every information source exposes its corpus statistics in a cooperative manner such as providing the required information via the protocol of STARTS (Gravano et al, 1997). STARTS (Stanford Protocol Proposal for Internet Retrieval and Search) is a complete protocol of federated search in cooperative environments. It covers many topics from resource description acquisition to results merging. Source metadata acquisition part of the STARTS protocol obtains the information about information sources' contents and other features. More specifically, each information source is required to provide both the content summary containing information such as vocabulary and word frequencies and also other metadata indicating the properties of the information source such as stopwords, document score range and the type of retrieval algorithm.

Unfortunately, cooperative protocols are not valid in uncooperative environments like large organizations or the Web. In these environments, even the information sources are willing to share their information, it is not easy to judge whether the information they provide is accurate or not. Furthermore, it is not easy to coordinate the information sources to provide resource representations that are compatible with each other.

An alternative to the cooperative protocols is the query-based sampling approach. The only assumption made by query-based sampling (Callan & Connell, 2001) is that all the information sources run queries and return documents. It does not require information sources to cooperate to provide content information nor use a particular type of search engine. This solution generates and submits single word queries to each

information source and downloads some documents in the returned document ranked lists (Callan et al., 1999)(Callan & Connell, 2001). Experiments have shown that under a variety of conditions the query-based sampling method can acquire rather accurate content description for each hidden information source by using about 80 queries to download a relatively small number of documents (300 documents) (Callan, 2000)(Callan & Connell, 2001)(Craswell et al., 2000). More specifically, after obtained 250 sampled documents, about 80 percent of term occurrences in an information source can be covered by the term in the sampled data; and the spearman rank correlation coefficient (Press et al., 1992) which measures the similarities between two rankings of the sampled df values and the actual df values is about 0.7 with the possible optimal value as 1. Some variants of query-based sampling techniques are the focused probing (Ipeirotis & Gravano, 2002) which utilizes query probes pre-derived from rule-based classifier of a hierarchy of topics, and probe queries (Craswell et al., 2000) which are multi-term queries chosen from available query log.

3.1.2 Resource description of centralized sample database

The above resource description only keeps the information source vocabulary and the corresponding term statistics. Many federated search systems only utilize the sampled documents to obtain this type of resource description and then discard the sampled documents. In contrast, these sampled documents can be combined into a single searchable database as the *centralized sample database*. Ogilvie and Callan's work (Ogilvie & Callan, 2001a) was one of the earliest research to utilize centralized sample database. Although their attempt of using centralized sample database for query expansion was not successful, centralized sample database may serve as a type of important information for many other problems. In this work, the centralized sample database is constructed and included as another type of useful information in resource descriptions.

3.1.3 Resource description of information source size

Information source size is another important type of information of a hidden information source. The main reason is that information source size estimates are very important to further sub-problems such as resource selection and results merging. For example, resource selection algorithms need source size estimates to adjust (normalize) the information source selection scores when they estimate the number of relevant documents across the information sources (Si & Callan, 2003b). A minor reason to acquire information size estimates is that it is very desirable to present information size estimates together with the information source recommendation results or the final document retrieval ranked lists so that the users may get broader information. Information source size can be defined in many different ways such as the size of vocabulary, number of words or the number of documents. In this work, we define information source size as the number of documents; other statistics such as the number of words or the size of vocabulary can be derived from the number of documents (Zipf, 1949)(Mandelbrot, 1988).

Very limited research has been conducted to estimate information source size. Liu and Yu (Liu et al., 2001) proposed a Capture-Recapture algorithm to estimate the information source size statistics. This method follows previous work in the statistic community of estimating population size of wild animals. Specifically, this algorithm assumes that we can get two independent documents id lists from a particular information source. Let N denote the actual information source size, A be the event that a document id was included (captured) in the first sample, which contained altogether n1 documents ids, B be the event that a document id was in the second sample, whose size was n2, and m2 is the number of document ids that were in both samples. The probabilities of events A and B can be calculated as follows:

$$P(A)=\frac{n_1}{N} \qquad P(B)=\frac{n_2}{N} \tag{3.1}$$

The conditional probabilities that a document id appeared in the second sample given it was observed in the first sample is:

$$P(B|A) = \frac{m_2}{n_1} \tag{3.2}$$

As these two samples are assumed to be independent as follows:

$$P(B|A) = P(B) \tag{3.3}$$

Finally, if the $\widehat{N}$ denotes the size estimate for this information source, it can be obtained as:

$$\frac{n_2}{\widehat{N}} = \frac{m_2}{n_1} \quad \Rightarrow \quad \widehat{N} = \frac{n_1 n_2}{m_2} \tag{3.4}$$

With the basic Capture-Recapture method, Liu and Yu reported rather accurate information source size estimates (the error rate is about 5% in the size estimate of an information source with 300,000 documents) (Liu et al., 2001). However, their method was not efficient. The basic Capture-Recapture method used a large number of sampled queries (about 1,000 queries) and large sample sizes (retrieved 1,000 document ids for each query) to estimate the size of an information source with 300,000 documents.

3.2 New information source size estimation methods

The basic Capture-Recapture algorithm is shown to obtain accurate information source size estimates. However, it is based on an important assumption that long ranked list (1,000 document ids for a query) can be acquired by one interaction with information source. This is often not true in real world applications. Whenever only short ranked lists are avaible by a single interaction, the basic Capture-Repcature algorithm may require excessive communication costs. Our goal is an effective and much more efficient source size estimation method.

In this section, other variants of Capture-Recapture algorithm are first proposed to utilize different possible methods of accessing ranked lists. Furthermore, a new source size estimation algorithm based on different estimation strategy is introduced.

3.2.1 New variants of Capture-Recpature method

The effectiveness of the Capture-Recapture algorithm can be strongly influenced by the method of accessing the ranked document lists provided by the individual hidden information sources. Generally, long document ranked lists can not be obtained by a single interaction with information sources in real world applications. On the contrary, only much shorter ranked lists (e.g., 10 or 20) may be available. Assuming 20 document ids can be returned in a single result page (some information sources allow more document ids to be returned while some allow even fewer; for simplicity, a fixed number of 20 is used), the Capture-Recapture algorithm is allowed to choose document ids from a id pool of 20 by a single interaction with an information source. If the ranked lists can only be obtained sequentially, Capture-Recapture algorithm can only access the top 20 document ids for each query with one interaction. On the other side, if the information source provides the service of accessing the ranked list at any specified section, it is possible for the Capture-Recapture algorithm to acquire more random document ids. These two variants of the Capture-Recapture algorithms are called the “Top” approach and the “Direct” approach respectively.

Another direction of extending the basic Capture-Recapture algorithm is to vary the amount of document ids included in the samples from each returned ranked list page. The basic Capture-Recapture algorithm only randomly chooses a single document id from a returned ranked list page (“1” approach), while it is possible to take advantage of all the 20 document ids in the returned list page and the corresponding variant of Capture-Recapture algorithm is denoted as the “All” approach.

3.2.2 Sample-Resample method

Sample-Resample method is a new information source size estimation algorithm, which makes different assumptions than the variants of the Capture-Recapture algorithm. There are two assumptions made by this new information source size algorithm. First, it assumes the resource representations are created by query-based sampling algorithm. Furthermore, the sampled documents from all the information sources are collected in a single *centralized sample database* with the information source identities to indicate which information source a specific document is sampled from. Therefore, the document boundaries of sampled documents are preserved and each term's document frequency in the sampled documents of an information source can be easily acquired. This is our first utilization of the centralized sample database, and more applications of the centralized sample database with resource selection and results merging are shown in later chapters.

The second assumption of the Sample-Resample method is that each information source provides the information of how many documents in this information source match a single word query. This type of statistics is often presented with the returned document ranked lists of many information sources even in uncooperative environments. For example, both Google and AltaVista indicate the approximate number of documents matching a single word query.

The new method estimates the information source size with a sample and resample process. Therefore, it is called Sample-Resample information source size estimation algorithm. The basic procedure of this algorithm is: First, query-based sampling method is used to build the resource description and create the centralized sample database for each information source; Second, a single term is randomly selected in the resource description of a specific information source. This term is submitted to the information source as a single word query. Assume N_{samp} documents from this information source have been sampled and collected in the centralized sample database. Let df_{q_samp} be the number of sampled documents that contain this query term. N denotes the information source size and df_q denotes the actual document frequency of the query word in the whole information source.

Let A denote the event that a sampled document contains this query term in the sample process and B denote the event that an arbitrary document in the whole information source contains this term in the resample step. The probabilities of these two events can be calculated as follows:

$$P(A)=\frac{df_{q\_samp}}{N_{samp}} \qquad P(B)=\frac{df_q}{N} \tag{3.5}$$

As long as the sampled documents acquired by query-based sampling can be assumed as a good representation of the whole information source, the above two events should have equal probabilities or formally as P (A) = P (B). Then, the corresponding information source size is estimated as:

$$\widehat{N}=\frac{df_q * N_{samp}}{df_{q\_samp}} \tag{3.6}$$

A refinement of the Sample-Resample algorithm is to submit multiple resample queries and calculate the average of individual information source size estimates in order to reduce the variance. For example, if altogether K resample queries are submitted, the final information source size estimate is calculated as follows:

$$\widehat{N}=\frac{1}{K}\sum_{k=1}^{K}\frac{df_{q_k} * N_{samp}}{df_{q_k\_samp}} \tag{3.7}$$

3.3 Evaluation methodology and experimental results

Very little prior work has been done to study the evaluation methods of information source size estimation algorithms. This section discusses how to define evaluation metrics and the associated costs of the information source size estimation algorithms. Furthermore, experiments are conducted to study the effectiveness and efficiency of both the Capture-Recapture algorithm and the Sample-Resample algorithm.

3.3.1 Evaluation methodology

Information source size estimation algorithms are associated with different types of costs. To give accurate evaluation of these algorithms, we need to compare their effectiveness with the same amount of costs. In this work, the costs of different information source size estimation algorithms are measured by the number of interactions with the hidden information sources that they require in order to obtain the size estimates. Therefore, as both the action of acquiring a page of ranked document ids and the action of downloading a document need a single interaction with the information source, these two actions are associated with the same amount of cost.

Both the Capture-Recapture algorithm and the Sample-Resample algorithm submit queries to the hidden information sources to collect some information. For the Capture-Recapture algorithm, it sends out a query and extracts document ids from the returned document ranked lists. If it is assumed that 20 document ids can be returned in a single result page (some information sources allow more document ids to be returned while some allow even fewer; for simplicity, a fixed number of 20 is used), the Capture-Recapture algorithm is allowed to choose document ids from a id pool of 20 by a single interaction with a particular information source. The Sample-Resample algorithm requires submitting several queries to the hidden information source in the resample step to collect document frequencies of the whole information source.

There are two evaluation scenarios to compare the Capture-Recapture algorithm and the Sample-Resample algorithm. In the first scenario information source estimation is combined with other components of a federated search system, so both of the Capture-Recapture algorithm and the Sample-Resample algorithm can take advantage of the information acquired by query-based sampling method. Therefore, the Capture-Recapture algorithm can use the document ids in the result pages of sampled queries and the Sample-Resample algorithm can access the downloaded documents to calculate the sampled document frequency statistics. More specifically, let query-based sampling method obtain the resource description for each information source by submitting 80 queries and downloading 300 documents. The Sample-Resample algorithm can take advantage of the 300 downloaded documents (80 queries are implicitly used for acquiring the documents) while the Capture-Recapture algorithm can only utilize the 80 pages of ranked document ids from the sample queries. In the resample process the Sample-Resample method needs several extra queries. The number of the resample queries is set to 5 in the experiments of this chapter. To investigate the influence of the number of resample queries on the accuracy of information source size estimates is an interesting future topic, which is discussed in more detail in Chapter 7. Therefore, the Capture-Recapture algorithm of this scenario is only allowed to acquire altogether 85 pages of ranked document ids as shown in Figure 1.

In the second scenario, the information source size estimation algorithms are independent applications of other federated search components, where the information from query-based sampling can not be accessed for free and the costs such as downloading documents must be included in the evaluation. Therefore, the cost of Sample-Resample algorithm is increased and thus the Capture-Recapture algorithm of Scenario 2 is allowed to obtain more pages of ranked document ids as shown in Figure 1. There is a favor bias towards the Capture-Recapture algorithm for the second evaluation scenario where it is allowed to send 385 queries to obtain pages of ranked document ids instead of only 85 in the first scenario.

Scenario 1 is a better representation of operational environments. However, in order to conduct component level study and to evaluate the effectiveness of the new proposed Sample-Resample algorithm in a stricter manner, the second evaluation scenario is chosen in this work.

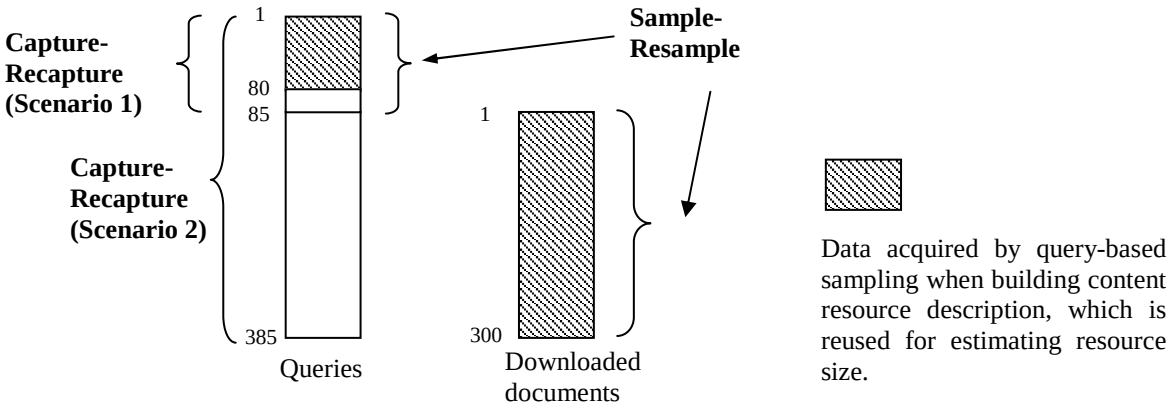

Figure 1. The data utilized by the Capture-Recapture algorithm and the Sample-Resample algorithm. (the shadow part of the data may be obtained from query-based sampling.)

Table 5. Testbed statistics of Trec123_100Col and Trec123_10Col.

Name	Size (GB)	Number of Docs (x 1000)			Megabytes (MB)		
		Min	Avg	Max	Min	Avg	Max
Trec123_100Col	3.2	0.7	10.8	38.7	28	32	41.8
Trec123_10Col	3.2	17	107	263	300	320	378

To summarize, the Capture-Recapture method is allowed to send 385 queries to a hidden information source, where the document ids in the first half of the queries are used as the first sample and the document ids in the second half of the queries are collected in the second sample. Different variants of the Capture-Recapture algorithm have different methods of accessing the ranked lists and utilizing different number of document ids per returned list page. By the same amount of interactions with the information source, the Sample-Resample method uses 80 queries to download 300 documents from the information source in the sample process and submits additional 5 queries in the resample step.

We use the absolute error ratio (AER) measure to evaluate the information source size estimation algorithms, which is consistent with prior work (Liu et al., 2001). Formally, let N_{dbi} denote the actual information source size of the i^{th} information source and $\widehat{N}_{db_i}$ denote the corresponding estimate. Then, the AER is calculated as follows:

$$AER = \frac{\left|\widehat{N}_{db_i} - N_{db_i}\right|}{N_{db_i}} \tag{3.8}$$

The mean absolute error ratio (MAER) is used by averaging the individual AER values in the case when a set of information source size estimates are evaluated.

Two testbeds were used in our empirical study, namely the Trec123-100Col testbed and the Trec123-10Col testbed. These two testbeds contain information sources of different sizes (Table 5), and they were chosen to evaluate the effectiveness of the Capture-Recapture algorithm and the Sample-Resample algorithm for small information sources (about 10,000 documents by average) and large information sources (about 100,000 documents by average) respectively.

Table 6. Experiment results of the information source size estimation methods on Trec123-100Col and Trec123-10Col testbeds with mean absolute error ratio (MAER) metric. (the lower the value, the better the result.)

Testbed	Cap-Rep Top1	Cap-Rep TopAll	Cap-Rep Direct1	Cap-Rep DirectAll	Sample-Resample
Trec123-100Col	0.729	0.377	0.566	**0.182**	0.232
Trec123-10Col	0.943	0.849	0.845	0.404	**0.299**

3.3.2 Experiment results

The experiment results of all the variants of the Capture-Recapture algorithm and the Sample-Resample algorithm are shown in Table 6. It can be seen that the basic Capture-Recapture algorithm ("Top1") was the worst among all the algorithms. The two extensions proposed in Section 3.3.1 of randomly accessing the document ranked lists and including all document ids in the pages of returned results have substantially improved the accuracy of the Capture-Recapture algorithm. The improvement can be attributed to the reasons that the random access of the document ranked lists makes the two document id samples more independent and the larger number of documents included in the samples is also helpful.

The Sample-Resample algorithm was more accurate than all the variants of the Capture-Recapture algorithm except for the DirectAll method on the Trec123-100Col testbed. Its advantage over the Capture-Recapture algorithm was more remarkable on the Trec123-10Col testbed with larger information sources. The Capture-Recapture algorithm was only comparable with the Sample-Resample algorithm in the small information source case with the support of random access of the ranked lists, which is a very strong assumption and may not be provided by many information sources. Therefore, we tend to draw the conclusion that Sample-Resample algorithm is more robust than the Capture-Recapture algorithm especially for large information sources. We are not arguing that the Sample-Resample algorithm is better than the Capture-Recapture algorithm in all cases. In the environments where the information sources do not provide document frequency information, where the information source can provide a large amount of document ids (larger than 20) per page of ranked list, or where the random access of ranked list can be guaranteed, special variants of Capture-Recapture algorithm may have their advantages.

Another issue observed from the experiment results is that both the Capture-Recapture algorithm and the Sample-Resample algorithm tended to underestimate the information source sizes. This can be explained by the assumptions made by these two approaches. The Capture-Recapture algorithm assumes that the two samples of document ids are independent. This is not perfect as some documents with more words and more diverse contents are more likely to be retrieved for different queries and thus more likely to appear in both the first sample and the second sample. On the other side, the Sample-Resample algorithm assumes that the sampled documents are representative of the whole information source. However, as the complete information source contains a large proportion of unseen words, the percentage of documents containing a sampled word is usually overestimated based on the sampled documents.

3.4 Future research

This chapter has shown the proposed Sample-Resample algorithm to be more effective than the Capture-Recapture algorithm in some cases. However, the behavior of the Sample-Resample algorithm should be studied with more thorough experiments such as for even larger information sources (information source of several thousand hundred documents) or with various number of resample queries. Furthermore, the Sample-Resample algorithm described in this chapter depends on document frequency information provided by individual search engines, which is not true in many operational environments. New variant of the Sample-Resample algorithm is possible to release the constraint. All the new research is discussed in Chapter 7.

Chapter 4: Resource Selection

After the resource descriptions are acquired, the decision of which information sources to search must be made in both the information source recommendation application and the federated document retrieval application. This chapter discusses previous research on resource selection and shows the attempts to incorporate information source size estimates with two well-known resource selection algorithms: CORI and KL divergence. Furthermore, the deficiency of the "big document" resource selection approach is discussed. Based on this discussion, a Relevant Document Distribution Estimation (ReDDE) resource selection algorithm is introduced, which tries to explicitly optimize the high-recall goal of information source recommendation application. Experiment results are shown to evaluate the accuracy of these resource selection algorithms.

4.1 Previous research on resource selection

A large number of resource selection algorithms have been proposed in the literature, which includes bGlOSS/gGlOSS/vGlOSS (Gravano et al., 1994)(Gravano et al., 1999), query clustering/RDD (Voorhees et al., 1995), decision-theoretic framework (DTF) (Fuhr, 1999)(Nottelmann & Fuhr, 2003b), lightweight probes (Hawking & Thistlewaite, 1999), CVV(Yuwono & Lee, 1997), CORI (Callan, 1995b)(Callan, 2000), KL-divergence algorithm (Xu & Croft, 1999), and a hierarchical database sampling and selection algorithm (Ipeirotis & Gravano, 2002).

The three algorithms of Query clustering/RDD, DTF and lightweight probes take advantage of training data in different ways. Query clustering/RDD and DTF methods require human relevance judgments while lightweight probes method obtains necessary statistics in an online manner. CVV, CORI and KL-divergence algorithms follow the strategy of "big document" approach by treating information sources as big documents and ranking information sources by their similarity scores with user queries. On the contrary, bGlOSS/gGlOSS/vGlOSS algorithms turn away from the "big document" approach by considering goodness/utilities of individual documents. Finally, the hierarchical database sampling and selection algorithm builds information source hierarchy and utilizes other base resource selection algorithms (e.g., CORI or KL-divergence) to rank the information sources. More detail of these algorithms is described in the rest part of this section.

Query clustering/RDD (Voorhees et al., 1995) rely on query log that is composed of a set of training queries and relevance judgments. These algorithms detect similar training queries of a user query with methods such as k nearest neighbor (Yang, 1999)(Duda, Hart & Stork, 2000), and rank the information sources by the distribution of the relevant documents of these training queries. They may work well when enough training queries and corresponding relevance judgments are available. However, the main problems in applying these methods are: i) the relevance judgments require large amount of human efforts, which grow linearly with the number of information sources; and ii) when individual information sources update their contents, it needs large amount of work to make the query log synchronous with the change.

The DTF method (Fuhr, 1999)(Nottelmann & Fuhr, 2003b) yields an information source selection that minimizes a cost function of overall costs (e.g., retrieval accuracy, query processing cost and communication cost) for a federated document retrieval application. It is based on a solid theoretic decision framework but may suffer large amount of human judgment efforts as it requires building a model for each individual information source. Chapter 6 shows a more detail analysis of this algorithm.

Lightweight probes method (Hawking & Thistlewaite, 1999) broadcasts two-word subsets of user queries to the information sources to obtain query term statistics. These term statistics are used to rank the information sources. This method needs very little amount of knowledge about each information source and calculates the information source ranking in an online version, so it is better in recognizing the content change of the information sources. However, sending query probes can cause significant communication costs in a large federated search system with many information sources.

A variety of resource selection algorithms share the same property of treating information sources as big documents and calculating similarities between these "big documents" and user queries to make the selection decision. These big document resource selection algorithms include the CVV (Yuwono & Lee, 1997), the CORI (Callan, 1995b) and the Kullback-Leibler (KL) divergence algorithm (Xu & Croft, 1999) etc. They choose different representation of the "big documents" to calculate different types of similarity scores (information source selection scores) and do not directly consider whether individual documents within an information sources are relevant or not. Therefore, for the high-recall goal of information source recommendation application, the actual goodness of each information source as the amount of relevant documents it contains can not be directly estimated from these big document methods. Different methods try to normalize or adjust their similarity scores to more directly reflect the information source utilities, but this is a difficult job for the big document approach.

The CVV (Cue Validity Variance) resource selection algorithm (Yuwono & Lee, 1997) assigns different weights (CVV) to the words. The words that better discriminate information sources are assigned higher weights than the words that distribute more evenly across the information sources. More specifically, a term which appears in most documents of a specific information source and appears very rarely in other information sources is assigned a higher CVV value, while a term appears in about the same proportions of documents among different information sources is assigned a lower CVV value. The CVV algorithm ranks the information sources by the sum of the weighted document frequencies of query words as follows:

$$s_i = \sum_{q \in Q} CVV_q * df_q \tag{4.1}$$

As large information sources tend to have large document frequency values, it has been indicated that the CVV resource selection algorithm tends to favor large information sources and information source size normalization may be necessary to improve the selection accuracy (Craswell, 2000).

The CORI resource selection algorithm (Callan, 1995b)(Callan, 2000) is a Bayesian inference network model that adapts an Okapi term frequency normalization formula (Robertson & Walker, 1994) to rank the information sources. CORI is related with the INQUERY ad-hoc retrieval algorithm. Formally, the belief of the i^{th} information source according the word q is calculated by:

$$T = \frac{df}{df + 50 + 150 * cw_i / avg\_cw} \tag{4.2}$$

$$I = \frac{\log(\frac{|DB| + 0.5}{cf})}{\log(|DB| + 1.0)} \tag{4.3}$$

$$p(q|db_i) = b + (1-b) * T * I \tag{4.4}$$

where: df is the number of documents in the i^{th} information source that contain q;

cf is the number of information sources that contain q;

|DB| is the number of information sources to be ranked;

cw_i is the number of words in the i^{th} information source;

avg_cw is the average cw of the information sources to be ranked; and

b is the default belief, usually set to 0.4.

The belief P (Q | db_i) denotes the probability that query Q is satisfied with the observation of the i^{th} information source. The most common way to calculate the belief P (Q | db_i) is to use the average of belief of each query word while some more complex operators are also available (Turtle, 1990)(Callan, 2000).

Kullback-Leibler (KL) divergence resource selection algorithm was proposed by Xu and Croft (Xu & Croft, 1999). In this method, the content descriptions of all the information sources and user queries are treated as multinomial distributions, and the Kullback-Leibler divergence between the distributions of the query and the information sources are used to measure how well the contents of these information sources match the query. Formally, the KL divergence between the query Q and the i^{th} information source is computed as:

$$KL(Q,db_i)=\sum_{q\in Q} P(q|Q)\log\left\{\frac{P(q|Q)}{\lambda P(q|db_i)+(1-\lambda)P(q|G)}\right\} \quad (4.5)$$

P (q | db_i) is the probability of query term q in the unigram language model (multinomial distribution) of the content descriptions of the i^{th} information source. P (q | Q) and P (q | G) are the probabilities of the query term q in the query language model and a global language model respectively. The global language model can be obtained by combining the individual unigram language models of the information sources together (weighted by their information source sizes). Linear interpolation constant λ smoothes the information source language model with the global language model and is set to a numerical value between 0 and 1 such as 0.5.

CORI and KL-divergence resource selection algorithms have been shown in previous study to be more robust and effective than several alternatives in different experiment environments (French et al., 1999)(Craswell et al., 2000)(Xu & Croft, 1999). They are computational efficient and easy to be incorporated with the query-based sampling method in uncooperative environments. However, they belong to the big document resource selection approach and do not normalize the lengths of hidden information sources well. Section 4.4 shows experiments that both CORI and KL-divergence algorithms are biased against large information sources and thus miss a large amount of relevant documents.

Gravano and García-Molina proposed the resource selection algorithm bGlOSS (Gravano et al., 1994) and the gGlOSS/vGlOSS algorithms in (Gravano & García-Molina, 1995)(Gravano et al., 1999). The bGlOSS algorithm is based on Boolean retrieval algorithm. It assumes that the query term distributions are independent and estimates the number of documents containing query terms to rank the information sources. The vGlOSS algorithm is a more sophisticated model based on vector space model. It represents a document in the vector space of m distinct words as $<w_{1,...,}w_m>$ where w_k is the weight (such as the idf value) assigned to the k^{th} word in the document, and a query in the same space as $<q_{1,...,}q_m>$ where q_k is typically a function of the number of occurrences the k^{th} word appears in the query. With these representations, the vGlOSS method calculates the similarity sim(Q , d) between a query Q and a document d as follows:

$$sim(Q,d)=\sum_{k=1}^{m} q_k * w_k \quad (4.6)$$

vGlOSS calculates the *goodness* of the information sources with respect to the query and ranks them accordingly:

$$Goodness(l,Q,db_i)=\sum_{\{d\in db_i,\ sim(Q,d)>l\}} sim(Q,d) \quad (4.7)$$

The ideal rank of information sources Ideal(l) is then determined by sorting the information sources according to their goodness for the query. However, to calculate the exact values of the goodness in Equation 4.7, the information about all the documents in the hidden information sources must be acquired,

which is not available because otherwise we can directly apply the ad-hoc retrieval approach to all the documents crawled in a single centralized database. Therefore, vGlOSS tries to approximate the Ideal(l) function by another two functions Max(l) and Sum(l). These two functions calculate the estimated goodness for an information source based on a high correlation scenario and a disjoint scenario of the query word concurrences respectively. vGlOSS needs two vectors of information as the document frequency and the sum of weights of each word from a particular information source to calculate Max(l) and Sum(l). Information sources can provide these statistics in cooperative environments or sampling queries can be sent to learn the information in uncooperative environments.

The vGlOSS method does not treat information sources as big documents and does consider individual documents to judge whether they are relevant or not. This strategy makes vGlOSS algorithm different from big document resource selection approach and gives it a better opportunity to model the utility of each information source as the amount of relevant documents it contains. However, two important issues limit its power. First, the two approximations Max(l) and Sum(l) make too strong assumptions about the query word distribution within the information sources. Max(l) assumes query words always occur together in the documents while Sum(l) assumes that query words do not occur together. These two strong assumptions are not valid and tend to introduce large errors. Second, the word weight w_k in Equation 4.6 is information source specific and thus the same document may be judged as relevant in one information source and as irrelevant in another information source. This arrangement may be valid when the retrieval accuracy issue of each information source is considered. However, as here the utility of a particular information source is measured by the number of relevant documents it contains, the information source specific weighting scheme is not appropriate (the issue of retrieval effectiveness of information sources is discussed in Chapter 7). These two issues can be used to explain why vGlOSS algorithm is less accurate than other algorithms such as CORI or KL-divergence in empirically studies (French et al., 1999)(Craswell, 2000).

The hierarchical database sampling and selection algorithm (Ipeirotis & Gravano, 2002) derives information source descriptions by using focused query probes (query on specific topics) and builds hierarchical structure for hidden information sources. It iteratively uses base resource selection algorithm in the hierarchy to do the resource selection. This hierarchy provides a better way to smooth the word distributions in the resource representations of information sources. However, as the resource selection is still conducted with base algorithms such as CORI, this resource selection algorithm still suffers from the weakness of the base algorithms.

4.2 Incorporate information source size effects with resource selection algorithms

The goal of resource selection algorithms in information source recommendation system is to select a small number of information sources with the largest number of relevant documents. Therefore, they need to estimate the number of relevant documents each hidden information source contains and thus the information source size factor is very important. However, very little research has been conducted to study the effect of information source size on resource selection. One reason is that information source size estimation has been a major unsolved problem until now. Chapter 3 presents the Sample-Resample method as a promising solution, and it provides an opportunity to better adjust or normalize the resource selection algorithms for information source size.

With the estimated information source size, an information source size factor is associated with a particular i^{th} database and defined as the ratio of its estimated information source size and the number of sampled documents from this information source. Formally as:

$$SF_{db_i} = \frac{\hat{N}_{db_i}}{N_{db_i\_samp}} \tag{4.8}$$

where $\hat{N}_{db_i}$ denotes the source size estimate for the i^{th} information source and N_{dbi_samp} denotes the number of sampled documents from this information source.

Previous research has shown that the CORI and KL-divergence algorithms are more robust and effective than several other alternatives in different experiment environments (French et al., 1999)(Craswell et al., 2000)(Xu & Croft, 1999). They were chosen as the baseline algorithms in this work and new variants of these two algorithms that adjust for information source sizes are also proposed in this section.

The CORI information source selection scores are calculated as Equations 4.2, 4.3 and 4.4. Callan indicated that the CORI formula of Equation 4.2 is a variation of Roberson's term frequency (tf) (Robertson & Walker, 1994) weight, where the term frequency is substituted by document frequencies in the sampled documents and the constants are scaled 100 times to accommodate the large document frequency values (Callan, 1995b)(Callan, 2000). Therefore, Equation 4.2 can be generalized and reformulated as follows:

$$T=\frac{df}{df+df\_base+df\_factor*cw_i/avg\_cw} \tag{4.9}$$

where df is the document frequency, df_base and df_factor are the two constants, cw_i and avg_cw represent the number of words in the i^{th} information source and the average number of words respectively. In uncooperative environments, CORI is combined with query-based sampling method and the above statistics are calculated on the sampled data. df is the document frequency in the sampled documents; cw_i and avg_cw are calculated based on the sampled documents in the same way and the df_base and df_factor constants are set to 50 and 150. This configuration has been shown to be effective on several testbeds with rather uniform source size distributions such as Trec123-100Col, where the average information source size is 10,000 and 300 hundreds documents are sampled from each information source.

One strategy of incorporating the information source size factor in the CORI resource selection algorithm is to simulate its behavior as if the complete resource descriptions are available (so the statistics in Equations 4.2, 4.3 and 4.4 are calculated from all the documents across the information sources). To accomplish that, there are at least three issues that should be addressed in Equation 4.9 (Si & Callan, 2004).

First, the document frequency represented by df in Equation 4.2 is the document frequency of a specific term q in the sampled documents. To calculate the document frequency in the whole information source, the sampled document frequency should be scaled as follows:

$$df' = df * SF_{db_i} \tag{4.10}$$

Second, cw_i in Equation 4.2 indicates the number of words contained in the sampled documents from the i^{th} information source, while avg_cw represents the average number of words in sampled documents across all the information sources. To incorporate the information source size factor, these two values should also be scaled as:

$$cw_i' = cw_i * SF_{db_i} \tag{4.11}$$

$$avg\_cw' = \frac{1}{|DB|}\sum_i cw_i * SF_{db_i} \tag{4.12}$$

where |DB| represents the number of information sources.

The last issue to be addressed is the two constants of df_base and df_factor. It was indicated in (Callan, 2000) that large df_base and df_factor should be used to accommodate larger value of the document frequency. However, how to set the values of df_base and df_factor is still not clear. While 0.5 and 1.5 are

chosen for ad-hoc document retrieval in the Okapi formula (Robertson & Walker, 1994), 50 and 150 are chosen for information source selection with complete resource descriptions and these values also work for sampled resource descriptions (about 300 documents for each information source). We do not try to solve the optimal setting of df_base and df_factor in this work. However, the effects of larger values of df_base and df_factore are investigated. More specifically, these two values are scaled as in the following Equations.

$$df\_base' = 50 * SF_{db_i} \tag{4.13}$$

$$df\_factor' = 150 * SF_{db_i} \tag{4.14}$$

Finally, all the new item values are plugged into Equation 4.9 to calculate an updated T value that considers the information source size factor. Furthermore, the information source beliefs P ($Q \mid db_i$) are calculated and the information sources can be ranked according to these beliefs.

Two variants of CORI algorithm are proposed based on the above extensions. CORI/Ext1 algorithm uses the updated document frequency as Equation 4.10 and the updated number of words in information sources as Equations 4.11 and 4.12. The second extension, which we call CORI/Ext2, takes advantage of all the updated document frequency, the updated number of words and the two new df_base and df_factor constants from Equations 4.13 to 4.14. The difference of CORI/Ext1 and CORI/Ext2 is the choice of relatively small values of df_base and df_factor (CORI/Ext1) and the choice of relatively large values (CORI/Ext2). The comparison between CORI/Ext1 and CORI/Ext2 helps us investigate these different settings.

KL-divergence resource selection algorithm was proposed by Xu and Croft (Xu & Croft, 1999). The basic KL-divergence algorithm views the resource representations of the information sources and the user queries as probability distributions and calculates the KL-divergence distance between the probability distributions to rank the information sources. Actually, the basic KL-divergence resource selection algorithm can be given an interpretation in the language-modeling framework. In this framework, all the sampled documents from a specific information source are collapsed into a single large document and the unigram language model (multinomial distribution) is calculated for each of the large documents. More specifically, the information sources are sorted by the probabilities of $P(db_i \mid Q)$, which are the generation probabilities of predicting different information sources based on the observation of query Q. By the Bayesian rule, the probabilities of $P(db_i \mid Q)$ can be further expressed as follows:

$$P(db_i|Q) = \frac{P(Q|db_i) * P(db_i)}{P(Q)} \tag{4.15}$$

where $P(Q \mid db_i)$ is the likelihood of generating query Q from the i^{th} information source; $P(db_i)$ is the prior probability, which represents our preference of this specific information source before observing the query. P (Q) is the generation probability of the query and acts as the normalization factor. Since it is not related with the information source ranking, it is ignored in the calculation. Following the Naïve-Bayes principle of ad-hoc language model approach, the query generation likelihood of $P(Q \mid db_i)$ is calculated as:

$$P(Q|db_i) = \prod_{q \in Q} (\lambda P(q|db_i) + (1-\lambda) P(q|G)) \tag{4.16}$$

$P(q \mid db_i)$ is the probability of generating a specific term q from the content descriptions of the i^{th} information source. $P(q \mid G)$ is the probability of generating this term by a global unigram language model, which is obtained by collapsing resource descriptions of all the information sources together and building a single unigram language model. The linear interpolation constant λ is introduced to smooth the

information source language model with the global language model and is usually adjusted in the range of 0 to 1 (It is set to 0.5 in this work).

To rank the information sources by the probabilities of P (db_i | Q), it is necessary to estimate the information source prior probabilities of P (db_i) as indicated in Equation 4.15. If we assign a simple uniform distribution on the information source prior distribution, the probabilities of P (db_i | Q) will be totally determined by the query likelihood P (Q | db_i). In this case, it is not difficult to show that the extended language model resource selection algorithm and the KL-divergence algorithm in Equation 4.5 are actually equivalent by simply taking the logarithm of Equation 4.15 and noticing that the term of $\sum_{q \in Q} P(q|Q)\log(q|Q)$ in Equation 4.5 is an information source independent constant.

Our strategy to incorporate the information source size factor into this extended language model resource selection algorithm is to assign the information source prior probabilities according to the information source sizes. Formally, the prior distribution is calculated as:

$$P(db_i) = \frac{\widehat{N}_{db_i}}{\sum_i \widehat{N}_{db_i}} \tag{4.17}$$

This is a natural idea as we prefer large information sources to small information sources when we are ignorant of the information need. More specifically, in a federated search system which contains two information sources A and B. A contains 10 documents and B contains 5. Without knowing the information need, all the documents are treated equally. Therefore, source A can be estimated to have two times more useful information than source B. Finally, Equations 4.16 and 4.17 are plugged into the language model resource selection framework in Equation 4.15, and the information sources can be selected according to the conditional probabilities of P (db_i | Q) as follows:

$$P(db_i|Q) = \frac{\prod_{q \in Q} (\lambda P(q|db_i) + (1-\lambda) P(q|G)) * \frac{\widehat{N}_{db_i}}{\sum_i \widehat{N}_{db_i}}}{P(Q)} \tag{4.18}$$

The new extended language model resource selection algorithm is denoted as the LM/Ext resource selection algorithm in this work.

The above extensions of the CORI and KL-divergence resource selection algorithms incorporate information source size factor in different ways, which improve the original big document approach. However, these extensions do not directly address the high-recall goal of information source recommendation application as the number of relevant documents contained in information sources. For example, the normalization approach of the extended CORI algorithm utilizes the source size factor to estimate the actual corpus statistics such as document frequency. However, each information source is still treated as a large document and the source size statistics are not directly used to estimate the number of relevant documents.

The extended language model resource selection algorithm goes a step further than the extended CORI algorithm. The likelihood probability P (Q | db_i) in Equation 4.16 can be seen as the averaged query generation probability from a single document in the i^{th} information source, and the source prior probability P (db_i) is introduced to reflect the effect of the information source sizes. If P (Q | db_i) is indicative of whether by average a single document in the i^{th} information source is relevant or not, the value P (db_i | Q) can be related with the number of relevant documents this information source contains. Although the extended language model resource selection algorithm goes a step further than the extended CORI algorithm to approximate the goal of information source recommendation system, it still does not explicitly deal with the individual documents, which may be a serious problem. For example, an information source A contains one very long relevant document and another nine irrelevant documents, while another source B

contains nine short relevant documents and another one long irrelevant document, the information source A and information source B may have the same unigram language models such that P (Q | db_A) equals to P (Q | db_B). Furthermore, since these two information sources contain the same number of documents, P (db_A) is also equivalent to P (db_B) and therefore P (db_A | Q) equals P (db_B | Q). The above derivation suggests that the two information sources are equally valuable to us. However, this is not consistent with our expectation as selecting source B gives us nine relevant documents while selecting source A only returns one.

The above observation tells us that we should treat the individual documents in an information source more explicitly. In other words, the probability of relevance for each document in the information sources should be estimated one by one to achieve better resource selection results.

4.3 Relevant document distribution estimation (ReDDE) resource selection approach

The relevant document distribution estimation (ReDDE) resource selection algorithm was proposed to turn away from the "big document" resource selection approach (Si & Callan, 2003a). It was pointed out that the goal of an information source recommendation system is to select a small number of information sources that contain the largest number of relevant documents. The ReDDE algorithm explicitly estimates the distribution of relevant documents across all the information sources and ranks the information sources according to this distribution.

Formally, the number of documents relevant to query Q in the i^{th} information source db_i is estimated as follows:

$$\text{Rel\_Q(i)} = \sum_{d \in db_i} P(\text{rel}|d) * P(d|db_i) * N_{db_i} \tag{4.19}$$

where N_{dbi} denotes the number of documents in the i^{th} information source and the probability P (d | db_i) is the generation probability of an arbitrary document in this information source. If all the documents in this information source can be downloaded, this probability will be 1/ N_{dbi}. However, in the real uncooperative environments, it is only possible to access the sampled documents, and the actual information source size N_{dbi} is replaced by its corresponding estimate $\widehat{N}_{db_i}$. As long as the sampled documents are assumed to be representative, Equation 4.19 can be approximated as:

$$\text{Rel\_Q(i)} \approx \sum_{d \in db_i\_samp} P(\text{rel}|d) * SF_{db_i} \tag{4.20}$$

The idea behind this equation is that when one sampled document from the information source is relevant to the user's query, we believe that there are about SF_{dbi} similar documents that are also relevant to the query among all the documents of this information source.

The only item left to be estimated in Equation 4.20 is P (rel | d), which denotes the probability of relevance of an arbitrary sampled document. Calculating this probability is a fundamental problem of information retrieval research. Many retrieval algorithms such as Bayesian belief network and Language Model have been proposed to address this problem. This problem is not solved in the general case here. Instead, the probability of relevance is approximated in a simple way as described below.

To approximate the probability of relevance, we need to take advantage of the available information. Query-based sampling generates the resource descriptions for the ReDDE resource selection algorithm. The sampled documents are combined to build the centralized sample database (CSDB), which is not only used

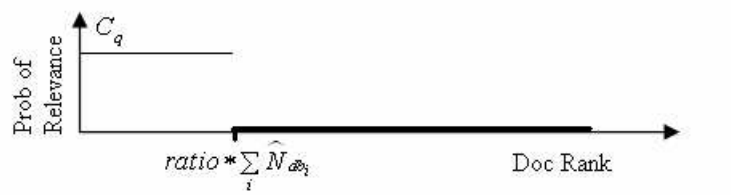

Figure 2. The curve of probability of relevance as a step function.

to estimate the information source sizes but also plays an important role to approximate the probabilities of relevance for all the documents.

To estimate the probability of relevance, we refer to a complete version of the centralized sample database, which is called the *centralized complete database* (CCDB). The centralized complete database is the union of all the individual documents available in the information sources. Of course, the centralized complete database is not accessible; otherwise, the ad-hoc retrieval method can be directly applied on the complete database instead of using the federated search solution. However, the centralized sample database is a good representative subset of the centralized complete database and the statistics on the centralized complete database can be estimated by the statistics on the centralized sample database. An example in this section shows how to simulate the retrieval ranked list on the centralized complete database by the ranked list on the centralized sample database.

The probability of relevance is modeled as a step function with respect to the retrieval result on the centralized complete database. More specifically, an effective retrieval algorithm is applied on the centralized complete database. The documents that rank at the top part of the centralized complete database have the probabilities of relevance as positive constants, while all the other documents have zero probabilities of relevance. This idea can be formalized as:

$$P(rel|d) = \begin{cases} C_Q & \text{if } Rank_{CCDB}(Q,d) < ratio * \sum_i \widehat{N}_{db_i} \\ 0 & \text{otherwise} \end{cases} \quad (4.21)$$

where $Rank_{CCDB}(Q, d)$ indicates the rank of document d in the retrieval results of the centralized complete database with respect to query Q. Ratio is a percentage threshold, which separates relevant documents form irrelevant documents. This formula is also visualized in Figure 2. The strategy of treating the curve of probability of relevance as a step function is a very rough approximation. However, this is a common approach in information retrieval research when the available information is very limited. For example, the pseudo relevance query expansion method uses the top documents in the initial retrieval as relevant documents to extract expanded query terms (Xu & Croft, 1996). Note that they chose a rank based threshold while a different ratio based threshold is used here to accommodate the large variation of the information source sizes.

The retrieval results on the centralized complete database are not directly accessible, but they can be approximated by the retrieval results on the centralized sample database. More specifically, the user query is submitted to the centralized sample database and searched by an effective ad-hoc retrieval algorithm, which is INQUERY (Callan, Croft & Broglio, 1995) in this work. The ranked list of the centralized complete database can be constructed with the ranked list of the centralized sample database and the information source size estimates. Formally, the rank of a document in the centralized complete database is calculated by:

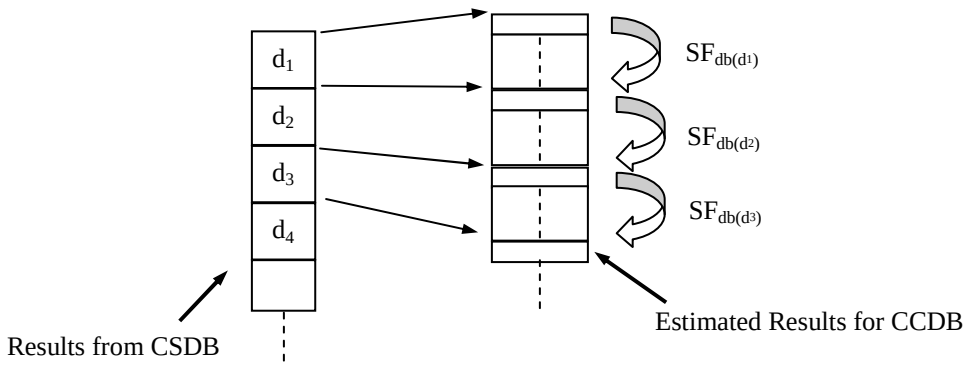

Figure 3. The approximation of the ranked list in centralized complete database by the ranked list in centralized sample database.

$$\mathrm{Rank}_{CCDB}(Q,d) = \sum_{\substack{d_j: \\ \mathrm{Rank}_{CSDB}(Q,d_j) < \\ \mathrm{Rank}_{CSDB}(Q,d)}} SF_{db(d_j)} \tag{4.22}$$

where $\mathrm{Rank}_{CSDB}(Q, d)$ denotes the rank of a document in the centralized sample database, $db(d_j)$ indicates which information source the document of d_j is from. The approximation of ranked list in centralized complete database by the ranked list of centralized sample database is also shown in Figure 3.

Given Equations 4.21 and 4.22, the number of relevant documents in each information source can be calculated as Equation 4.20. However, note that there still exists a query dependent constant C_Q in Equation 4.21, which makes the calculation of the exact number of relevant documents intractable. Therefore, the number of relevant documents across the information sources is further normalized to compute the distribution of the relevant documents as follows:

$$\mathrm{Dist\_Rel\_Q(i)} = \frac{\mathrm{Rel\_Q(i)}}{\sum_i \mathrm{Rel\_Q(i)}} \tag{4.23}$$

The numerator and the denominator in Equation 4.23 both contain the query specific constant C_Q, which is cancelled during the calculation. Therefore, Equation 4.23 provides us the computable distribution of relevant documents. It serves as the criterion to rank the information sources and the information sources with the largest number of relevant documents are selected as they cover larger fractions of relevant documents calculated by Equation 4.23. So this strategy explicitly meets the high-recall goal of information source recommendation system, which is to select information sources that contain the largest number of relevant documents.

4.4 Evaluation methodology and experimental results

This section discusses experimental methodology to evaluate resource selection algorithms. Furthermore, experiment results on testbeds with different characteristics are shown to compare the effectiveness of the basic version of CORI algorithm, the basic version of KL divergence algorithm, the extended CORI algorithms, the extended language model resource selection algorithm and the ReDDE algorithm.

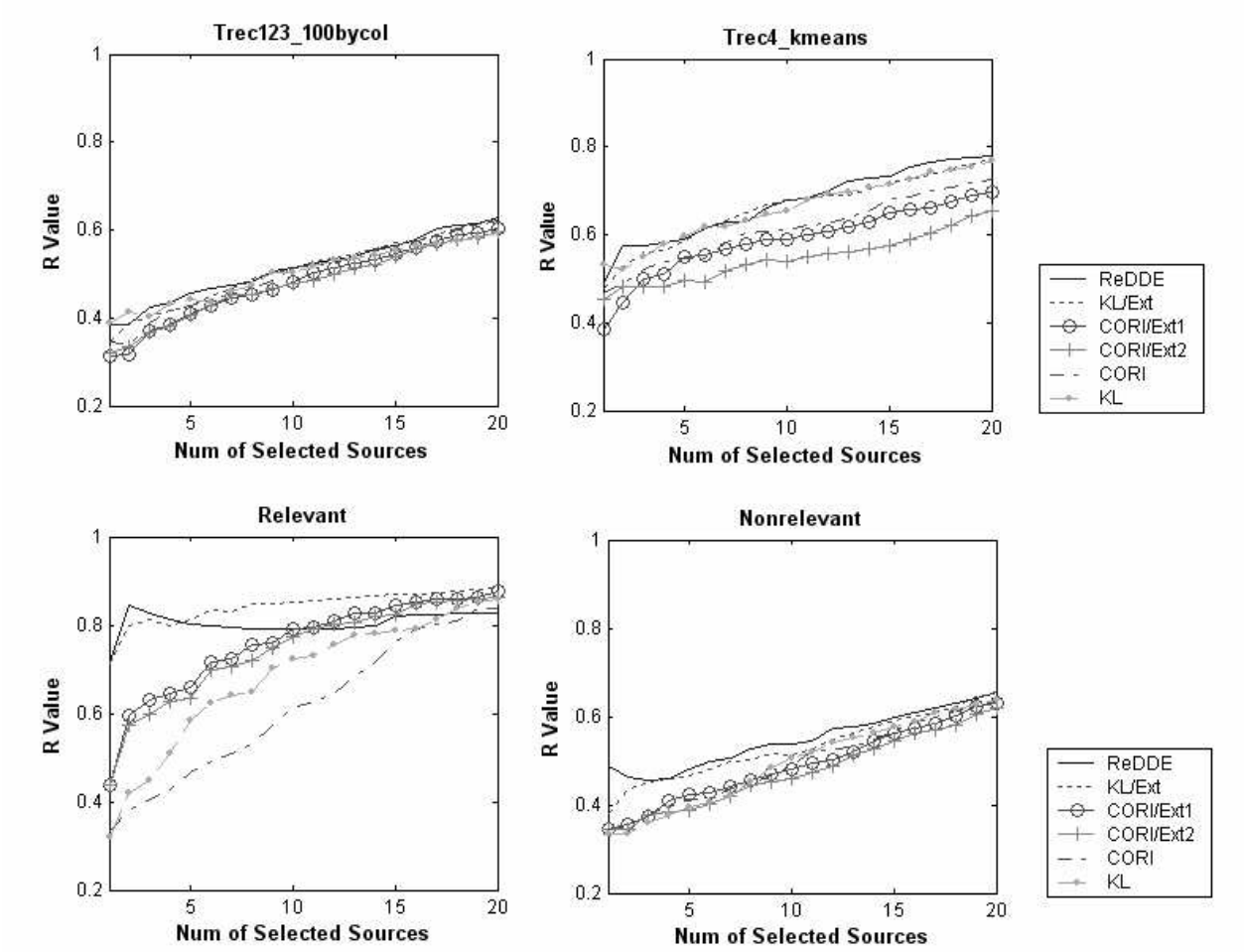

Figure 4. Resource selection experiments on four testbeds (with database size estimates).

4.4.1 Evaluation methodology

Resource selection algorithms of information source recommendation system select some information sources with the largest proportion of relevant documents. They are compared with a desired information source ranking called *Relevance-Based Ranking*, where the information sources are ranked by the actual number of relevant documents they contain. Let E be the ranking provided by the resource selection algorithm being evaluated, and B_i and E_i represent the number of relevant documents in the i^{th} ranked information source of B and E respectively. The recall metric R_n for comparison is defined as follows:

$$R_k = \frac{\sum_{i=1}^{k} E_i}{\sum_{i=1}^{k} B_i} \tag{4.24}$$

This comparison metric measures the percentage difference between the ranking algorithm in evaluation and the relevance-based ranking algorithm. Therefore, at a fixed k, a larger value of R_k indicates a better information source ranking result.

The experiments were conducted on four testbeds with six resource selection algorithms. The four testbeds cover a wide range of environments: relatively uniform information source size and content distributions (Trec123-100Col), moderately skewed information source size and moderately skewed content distributions (Trec4-kmeans), bimodal information source size distribution where large proportion of relevant documents in the large information sources (Relevant) and bimodal information source size distribution where small proportion of relevant documents are in the large information sources (Nonrelevant). More detail of these testbeds can be found in Chapter 2. The six resource selection algorithms are the basic CORI algorithm, the CORI/Ext1 algorithm (Section 4.2), the CORI/Ext2 algorithm (Section 4.2), the basic KL-divergence algorithm, the extended language model resource selection algorithm (LM/Ext) (Section 4.2) and the ReDDE algorithm (Section 4.3).

4.4.2 Experiment results

All the resource selection algorithms in the experiments used resource representations built by query-based sampling method, which submitted about 80 queries to each information source and downloaded 300 documents. The information source sizes were estimated by the Sample-Resample method as described in Chapter 3.

The experiment results are shown in Figure 4. The basic versions of the CORI and KL-divergence algorithms, which do not consider information source size factor, did reasonably well on the testbeds of Trec123-100Col with relatively uniform information source size distribution and Trec4-kmeans with moderately skewed source size distribution. However, their accuracy on the relevant and nonrelevant testbeds with more skewed information source size distributions was not very satisfactory. This indicates that information source size factor plays a very important role in resource selection. Therefore, robust resource selection algorithms designed to work in a wide range of environments should incorporate the information source size factors.

The two extensions of the CORI algorithm showed inconsistent behavior on these four testbeds. They were better than the basic CORI algorithm on the relevant testbed, about the same as CORI on the Trec123-100Col and nonrelevant testbeds, but even worse on the Trec4-kmeans testbed. This does not mean other more sophisticated modifications of the basic CORI algorithm can not work well in the environments with skewed information source size distributions, it only indicates that our two extensions of the CORI/Ext1 and CORI/Ext2 are not consistently successful.

In contrast to the extensions of CORI algorithm, the extended language modeling resource selection algorithm was at least as effective as the basic KL-divergence algorithm and all the variants of CORI algorithms on the Trec123-100Col testbed, or was even better on the Trec4-kmeans, relevant and irrelevant testbeds. The advantage of the extended language modeling resource selection algorithm can be attributed to the better treatment with individual documents in the information sources than the simple normalization method of the extended CORI algorithms, which still follow the “big document” approach (Section 4.2). The ReDDE resource selection algorithm calculates the probability of relevance of each individual document, and explicitly approximates the distribution of relevant documents. It was about as accurate as the extended language model resource selection algorithm on all the four testbeds and was more robust than all other methods. This success suggests that our discussion of the deficiency of “big document” resource selection algorithms is correct. However, no evidence shows that ReDDE algorithm had a large advantage over the extended language model resource selection algorithm on these four testbeds. The disadvantage of the extended language model resource selection algorithm is discussed in Section 4.2 and an example is proposed to demonstrate this weakness. We believe a simulated testbed with the characteristics of that example can help us demonstrate the advantage of the ReDDE algorithm against the extended language model resource selection algorithm. This will be considered in the dissertation research.

4.5 Future research

There are several interesting research topics to be investigated in the future research such as new testbed to show the advantage of ReDDE algorithm over the extended language model algorithm. Another interesting topic is to investigate resource selection goals other than high-recall. High-recall is adopted in this chapter to evaluate the effectiveness of information source recommendation system by the amount of relevant documents contained in the selected information sources. However, this may not be the case for a real operational application, where most users may only browse reasonable amount of documents (e.g., 100 or lower) retrieved from a particular selected information source when they visit this information source. Therefore, an alternative use scenario of information source recommendation system is to select a small number of information sources that can return the largest number of relevant documents in the top retrieved documents of selected information sources. The detail of this use scenario is discussed in Section 7.3.

Chapter 5: Results Merging

In a federated document retrieval system, user queries are forwarded to the selected information sources and returned ranked lists can be acquired from these information sources. There are two choices to organize these returned results. One approach is to display these results side by side in the same page or in different result pages. This method makes sense when the number of selected information sources is very limited (e.g., less than 3) and the results from these selected information sources are of about the same quality. However, when a larger number of information sources (e.g., more than 5) are selected or the qualities of returned documents from the selected information sources are quite different, it might be distracting to display the results separately and users often prefer a single merged ranked list by results merging. In the latter case, a results merging component is required in the federated document retrieval system.

Results merging is a hard problem especially in uncooperative environments. There are at least two issues that make individual ranked lists incomparable: i) different information sources may use different retrieval algorithms so that the range of relevant scores in the ranked lists may be totally different; and ii) even when two information sources use the same type of retrieval algorithms, they may return incomparable document scores due to different resource representations or different corpus statistics (e.g., vocabulary size, inverse document frequency, average document length, etc) .

This chapter first discusses previous research on results merging, and then proposes a *semi-supervised learning approach*, which utilizes the documents in the centralized sample database as training data and builds models to transform all the information source specific scores into final comparable document scores. Empirical studies are conducted to show the effectiveness of the new results merging approach.

5.1 Prior research on results merging

Results merging has received relatively less attention than resource selection in prior research of federated search, partially because of the misunderstanding that it is similar with the meta-search problem. However, the results merging sub-problem of federated document retrieval is different from meta-search (Lee, 1997)(Aslam & Montague, 2001)(Manmatha, Rath & Feng, 2001) in that, in meta-search, the individual ranked lists are retrieved from the same or very similar information sources with different retrieval algorithms so that there is a large proportion of overlap information among the individual lists; On the other side, in federated document retrieval, the contents of information sources are usually independent and we can not obtain much overlap information among the individual ranked lists.

One approach for results merging in the cooperative environments is to make every information source use the same type of retrieval algorithm and the same set of corpus statistics (Viles & French, 1995)(Xu and Callan, 1998)(Xu & Croft, 1999). A related approach is for each information source to return term frequency information for each retrieved document and each query term so that the search client can compute a consistent set of document scores using global corpus statistics (Kirsch, 1997). These methods can be quite accurate but all of them require information sources to use the same type of retrieval algorithm or provide term frequency information upon request, which is not the case in uncooperative environments.

Round Robin method (Voorhees et al., 1995) is a simple method that can be easily applied in uncooperative environments. It only utilizes the document rank information in individual ranked lists and the information source rank information of resource selection algorithms. It chooses the first document returned by the first selected information source as the first document in the merged list and the first document from the second

selected information source as the second one, and so on. A finer version, the Weighted Round Robin, which merges documents from information sources with consideration of their expected contribution, was also proposed. These methods are simple and easy to apply in uncooperative environments but are not very effective (Savoy & Rasolofo, 2000).

CORI results merging formula (Callan, 2000)(Callan et al., 1995b) follows the idea that both the documents from information sources with high selection scores and the high-scoring documents from information sources with lower selection scores should be favored. It is a heuristic formula with linear combination of the information source selection scores and the document scores from individual ranked lists. First, the information source selection scores and the document scores are normalized as:

$$S(d_i)' = \frac{S(d_i)-S(d_{min})}{S(d_{max})-S(d_{min})} \tag{5.1}$$

$$S(d_{ij})' = \frac{S(d_{ij})-S(d_{i\_min})}{S(d_{i\_max})-S(d_{i\_min})} \tag{5.2}$$

Associated with the CORI resource selection algorithm, Equation 5.1 normalizes information source selection score of the i^{th} information source to the range of [0, 1]. The raw score $S(d_i)$ of the i^{th} information source is the information source belief $P(Q \mid db_i)$ in the CORI resource selection algorithm. $S(d_{min})$ and $S(d_{max})$ are calculated by setting the T component in Equation 4.4 to 0 and 1 respectively. $S(d_{ij})'$ is the normalized document score for the j^{th} document from the i^{th} information source. In cooperative environments, the maximum document score $S(d_{i\_max})$ and the minimum score $S(d_{i\_min})$ are provided by the i^{th} information source, otherwise they are simply set to the maximum and minimum document scores returned by the information sources in uncooperative environments.

Eventually, the final comparable scores are calculated as:

$$S_C(d_{ij}) = \frac{S(d_{ij})'+0.4*S(d_{ij})'*S(d_i)'}{1.4} \tag{5.3}$$

The CORI results merging formula has been shown to be effective in previous research (Callan, 2000)(Callan et al., 1995b), and we make it the baseline algorithm to compare with our new proposed method.

A logistic transformation model for results merging was proposed in (Le Calv & Savoy, 2000), it takes the advantage of human-judged training data to build logistic models for different information sources to transform information source specific scores into information source independent scores. This method is more theoretically solid than the Round-Robin methods or the CORI formula, and has been shown to be effective in previous research (Le Calv & Savoy, 2000). However, it utilizes human-judged relevance information to build separate models for the information sources, so the corresponding human efforts may not be affordable with a large number of information sources. Another weakness of this approach is that the same model was used for all the queries of a given information source, but it can be imagined that the retrieval behavior of the same information source varies with respect to different queries and thus query-specific model would be more favorable than query-independent model.

The brief review of the existing results merging algorithms tells us that: i) most accurate algorithms require information sources to calculate consistent document scores or recalculate document scores in the central search client by making assumptions which are not practical in uncooperative environments; and ii) the CORI and Round Robin methods, which only utilize information from individual ranked lists and resource selection results are simple and efficient, but they are not very accurate. It can be noted that all these algorithms are trying to produce document ranked lists as if all the documents were stored in a single, global database. The CORI and Round Robin method only utilize the information from individual ranked lists and information source selection scores (or only ranks) but the results are not very accurate; while the methods based on exchanging corpus statistics or downloading returned documents are effective but rely on unpractical assumptions in uncooperative environments or require large amount of communicate costs.

5.2 Semi-Supervised Learning results merging approach

The Semi-Supervised Learning (SSL) (Si & Callan, 2002a)(Si & Callan, 2003b) method is a results merging algorithm to effectively and efficiently produce a single ranked list of documents that approximates the ranked list in the centralized environment. This method makes full use of the available information. Specifically, for each user query, the documents in the centralized sample database are searched with an effective centralized retrieval algorithm to acquire information source independent scores. Then, the sampled documents that have both information source independent scores and information source specific scores serve as training data to calculate models that transform information source specific document scores into the scores as if the they were calculated in the same environment of centralized sample database with the same type of retrieval algorithm. It can be noticed that the SSL method uses centralized sample database as an extra source of information, which is ignored by the CORI and Round Robin methods.

SSL algorithm is based on two assumptions: i) some documents retrieved from the selected information sources also exist in the centralized sample database; and ii) given both the information source independent scores and the information source specific document scores of these overlap documents, a linear function can be learned to transform the information source specific scores into the corresponding information source independent scores.

The first assumption indicates the availability of training data. It may be questionable, but one fact enhances the probability that enough overlap documents can be found: an information source is selected only if some of its sampled documents acquired by query-based sampling method appear to be relevant, thus these documents are likely to be retrieved from this information source. Experiments are presented below to study this problem later. Furthermore, we propose a variant of the SSL algorithm that downloads some retrieved documents on the fly to create extra training data when there are not enough overlap documents.

We do not argue that it is the best choice to use linear functions as the transformation models in the second assumption. Other more sophisticated transformation models may be more accurate in some special cases. The linear transformation model is chosen in this work because: i) the linear transformation model can be computed very efficiently and only requires a very small number of training data (as few as two overlap documents), and ii) the CORI results merging algorithm described in Equation 5.3 can be generalized into a linear model, which suggests the effectiveness of using a linear function.

Given some amount of overlap documents that have both information source specific document scores and information source independent scores, the optimal linear model which best transforms the information source specific document scores to the information source independent scores can be calculated. Formally, the linear transformation model can be calculated as follows:

$$(a\ ,\ b)^{*} = \underset{(a,b)}{\operatorname{argmin}} \sum_{j} (a * x_j + b - y_j)^2 \qquad (5.4)$$

where $(x_j\ ,\ y_j)$ is the information source specific score and information source independent score of the j^{th} overlap document respectively. $(a\ ,\ b)^*$ denotes the parameters of the optimal linear model, which are used to transform other documents with only information source specific scores into the corresponding information source independent scores. Finally, the merged ranked list is created by sorting the documents with their information source independent scores.

In uncooperative environments such as large organizations or the Web, it is most likely that the information sources are searched by multiple types of search engines (the multiple engine-types case). Different retrieval algorithms may generate document-specific scores with quite different score ranges, and it is not likely that a single linear model can transform these scores into information source independent scores. Therefore, multiple linear transformation models are learned for the multiple engine-types case. As our

research is focused on uncooperative federated search environments, we focus on multiple engine-types case in this work.

The first step is to identify the overlap documents for each information source that appear in both the centralized sample database and the corresponding ranked list. Formally, for each overlap document d_{ij} that comes from the i^{th} information source, the information source independent score is denoted as $S_c(d_{ij})$ and the normalized information source specific score is denoted as $S_i(d_{ij})$ (with the range of [0,1]). The goal is to estimate the linear transformation model for this information source that transforms all its information source specific document scores into the information source impendent scores as $S_c(d_{ij}) = a_i * S_i(d_{ij}) + b_i$. The regression problem over all the training data from the i^{th} information source can be formulated in the matrix representation as follows:

$$\begin{bmatrix} S_i(d_{i,1}) & 1 \\ S_i(d_{i,2}) & 1 \\ ..... & 1 \\ S_i(d_{i,m}) & 1 \end{bmatrix} * \begin{bmatrix} a_i & b_i \end{bmatrix} = \begin{bmatrix} S_c(d_{i,1}) \\ S_c(d_{i,2}) \\ ..... \\ S_c(d_{i,m}) \end{bmatrix} \tag{5.5}$$

where a_i and b_i are the two parameters of the linear model for the i^{th} database. The first matrix on the left hand side of the above equation can be denoted as X (constructed by information source specific document scores and constants), the second item on the left hand side (the linear transformation model parameters) as W and the item on the right hand side as Y (the information source independent document scores). Simple mathematic manipulation shows that the optimal solution can be expressed as follows:

$$W = (X^T X)^{-1}(Y^T X) \tag{5.6}$$

This solution is acquired by minimizing the squared error criterion described in Equation 5.4.

Following the same procedure, all the linear models for the selected information sources can be calculated to transform the information source specific document scores into the corresponding information source independent scores and finally the merged ranked list is constructed by sorting the documents with the information source independent scores.

The main steps of building the linear models are described as above. Some adjustments are also made to make the models more robust. Theoretically, as few as two data points are needed to train a linear model. More training data usually generates more accurate models, so we require at least three overlap documents to calculate a linear model. When an information source does not have enough training data, it is called a "bad" information source. One possible way to treat the bad information sources is to simply ignore the returned results from them, as it is believed that a "good" information source that contains a lot of relevant documents tends to have more relevant documents in the centralized sample database and thus have more overlap documents as the training data. However, when there are too many information sources without enough training data, it may suggest that the SSL approach is not good for this query, so the algorithm backs off to the CORI merging formula, which does not require any training data at all. The back off threshold is set empirically as 40%. For example, when there are more than 4 information sources out of totally 10 selected information sources containing less than 3 overlap documents, the algorithm backs off to the CORI merging formula.

In contrast, when more than enough training data is available for a good information source, it may be selective to choose the important ones for accuracy and efficiency. As it is believed that it is more important to be accurate on the top part of a ranked list, we only choose up to top 10 overlap documents from each selected information source as the training data.

Another adjustment is to correct the bias problem of anomalous linear models. A learned linear transformation model is anomalous when it produces an information source independent document score

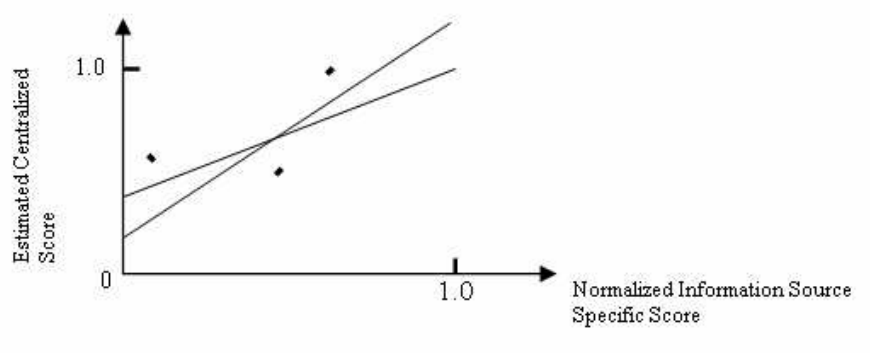

Figure 5. Adjust the bias problem of linear regression model.

great than 1, as INQUERY (Turtle, 1990)(Callan, Croft & Broglio 1995a) is used as the centralized retrieval algorithm and it is impossible to produce a document score larger than 1. A document with a score larger than 1 is not a serious problem by itself, but this indicates that the model may be too highly biased and the documents from this information source may have been given excessive advantages.

This bias problem is addressed by replacing the original linear model with another one that intersects the point of (1 , 1) and is closest to the original linear model (as shown in Figure 5). Formally, let y = ax + b be the original model and y = a'x + b' be the new model, the new linear model is obtained by solving the following problem:

$$(a', b') = \underset{a',b'}{\operatorname{argmin}} \int_0^1 [(a'-a)*x+(b'-b)]^2 dx \tag{5.7}$$

Simple mathematical manipulation shows the following update formula for the new model:

$$a' = \frac{3-a-3b}{2} \qquad b' = 1-a' \tag{5.8}$$

These small adjustments have been shown to slightly improve the results merging accuracy in empirical studies. Furthermore, this adjustment can be generalized to other cases when different centralized retrieval algorithms (They may produce different maximum retrieval scores) other than INQUERY are utilized.

5.3 Evaluation methodology and experimental results

A set of experiments was conducted to study the effectiveness of the Semi-Supervising Learning results merging algorithm under a variety of conditions. Specifically, this section first describes experimental methodology for results merging algorithms. Then, the sufficiency problem of overlap documents is studied in the case when long ranked lists can be acquired from the selected information sources, followed by the experiment results to compare the CORI results merging formula and the SSL algorithm. Finally, the scenario when only short ranked lists are available from selected information sources is addressed and a variant of the SSL algorithm is proposed to download a minimum number of documents as additional training data. Experiments are conducted to study the effectiveness of the new SSL algorithm.

5.3.1 Evaluation methodology

Experiments on two testbeds with three retrieval algorithms were conducted to compare the effectiveness of the SSL result merging algorithm and the CORI merging formula in various environments.

Two types of testbeds were used in our experiments, namely the Trec123-100Col testbed and the Trec4-kmeans testbed (The corpus statistics can be found in Tables 1 and 2). Trec123-100Col is organized by source and publication date. The contents of the information sources are relatively heterogeneous. Trec4-kmeans is organized by topic, where the information sources are relatively homogenous and the word distribution is more skewed than the Trec123-100Col testbed.

The effectiveness of federated search systems is highly influenced by testbed characteristics. As the contents of information sources are more homogenous, the Trec4-kmeans testbed is generally considered to be easier for resource selection than the Trec123-100Col testbed where the contents are much more heterogeneous. In contrast, the Trec123-100Col testbed is believed to be easier for results merging than the Trec4-kmeans testbed, as the n^{th} returned documents of different selected information sources on the Trec123-100Col testbed are more comparable than the same set of documents on the Trec4-kmeans testbed. Actually, the difficulty of results merging for testbed with quite skewed word distribution has been reported in previous research (Larkey, Connell & Callan, 2000).

The three retrieval algorithms introduced in Chapter 2 as INQUERY (Callan, Croft & Broglio 1995), Language Model (Lafferty & Zhai, 2001) and vector-space algorithms are chose in the experiments. All these three algorithm were implemented with the Lemur Toolkit (Ogilvie & Callan, 2001b), and they were assigned to the information sources in a round-robin manner (more detail in Section 2.3).

In all of our experiments, the resource descriptions were acquired by query-based sampling method. About 80 queries were sent to each information source to download 300 hundred documents, as the top 4 (or fewer when there are not enough) documents from each ranked list were acquired. All the sampled documents were collected in the centralized sample database, which contains 30,000 (100 information sources time 300 documents per information source) documents.

CORI resource selection algorithm was used to rank the information sources for the experiments in this chapter. It was chosen for two reasons: i) CORI resource selection algorithm has been used in many federated document retrieval applications; and ii) information source selection scores from the CORI resource selection algorithm is required by the CORI results merging formula, which serves as the baseline in the experiments. The document retrieval experiments with other more effective resource selection algorithms are discussed in Chapter 6.

5.3.2 Experimental results: Overlap documents

The first assumption of the SSL algorithm says that there exist sufficient overlap documents. This hypothesis is crucial for the success of the SSL algorithm. Many factors such as the information source sizes, the number of sampled documents from the information sources, the lengths of the returned ranked lists and the query characteristics determine the number of overlap documents. Generally, it can be expected that the longer the query, the higher percentage coverage of the sampled documents in a specific information source, the longer the returned document ranked list, the more number of overlap documents can be acquired.

A set of experiments were conducted to investigate the number of overlap documents in the case where a small proportion of documents are sampled (300 out of 10,000 by average for Trec123-100Col, 300 out of 5,600 by average for Trec4-kmeans) and long returned ranked lists are provided (up to 1,000 documents per selected information source). Acquiring long returned ranked lists from information sources in uncooperative environments where only short ranked lists are available may cause substantial communication costs. So a variant of the SSL algorithm is introduced in Section 5.3.4 to work in the case of short ranked lists (e.g., 50). However, as the accuracy of the SSL algorithm in the case of long ranked

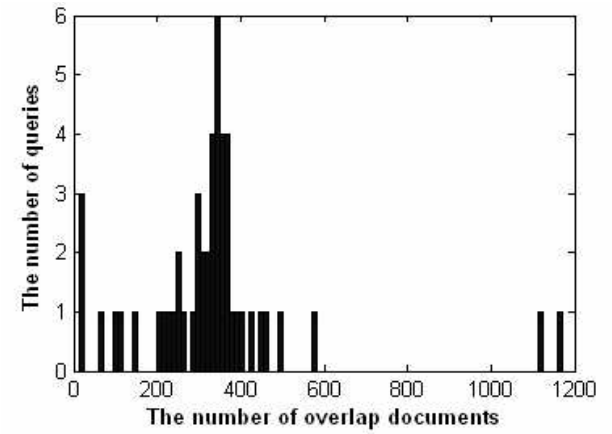

Figure 6. The histogram of overlap documents for 50 “title” queries on Trec123-100Col testbed.

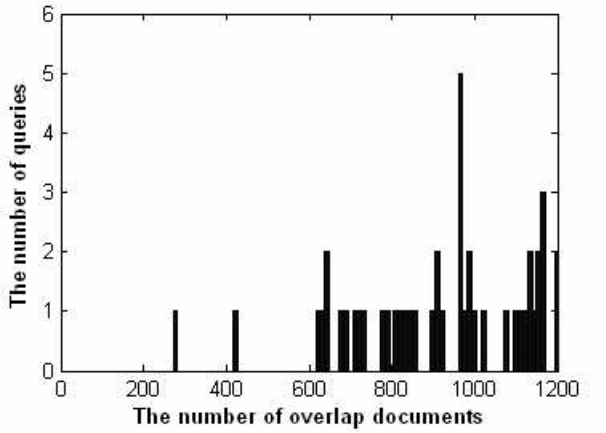

Figure 7. The histogram of overlap documents for 50 “description” queries on Trec4-kmeans testbed.

lists can serve as the baseline to compare its variant in the short ranked list case, the behavior of the SSL algorithm with long ranked lists is first studied.

Specifically, the experiments were conducted on two testbeds of Trec123-100Col and Trec4-kmeans. Both of them contain 100 information sources, and three retrieval algorithms of INQUERY, Language Model and Vector Space model were assigned to the information sources in a round-robin manner. Resource descriptions were built by query-based sampling to acquire 300 sampled documents from each information source. CORI resource selection algorithm was used to rank the information sources, and result lists of up to 1,000 document ids with their information source specific scores were returned by the selected information sources to the central search agent.

Very few queries (3 out of 50) on the Trec123-100Col testbed and none queries (0 out of 50) on the Trec4-kmeans testbed were short of overlap documents (more than 4 among the 10 selected information sources had less than 3 overlap documents). For more detail, Figures 6 and 7 show the histograms of the number of overlap documents for the queries on the Trec123-100Col testbed and the Trec4-kmeans testbed respectively.

The experiment results show that our assumption of sufficient number of overlap documents is satisfied for the environments where a small amount of documents are sampled and long ranked results lists can be returned. Particularly, the queries on the Trec4-kmeans testbed tend to have more overlap documents than the queries on the Trec123-100Col testbed, which is consistent with our expectation that the higher coverage of the sampled documents (the information sources on the Trec4-kmeans testbed are smaller than the information sources on the Trec123-100Col testbed by average while all of them contribute 300

Table 7. Precision at different document ranks using the CORI and Semi-Supervised Learning approaches to merging retrieval results from INQUERY, Language Model and Vector Space search engines. 3 sources were selected to search for each query. Results are averaged over 50 queries.

Document Rank	Trec123-100Col Testbed		Trec4_kmeans Testbed	
	CORI Merge	SSL Merge	CORI Merge	SSL Merge
5	0.3240	0.3680 (+13.6%)	0.2440	0.3440 (+41.0%)
10	0.3260	0.3420 (+4.9%)	0.2320	0.2840 (+22.4%)
15	0.3147	0.3253 (+3.4%)	0.1987	0.2520 (+26.8%)
20	0.2930	0.3090 (+5.5%)	0.1820	0.2260 (+24.2%)
30	0.2627	0.2747 (+4.6%)	0.1573	0.1993 (+26.7%)

Table 8. Precision at different document ranks using the CORI and Semi-Supervised Learning approaches to merging retrieval results from INQUERY, Language Model and Vector Space search engines. 5 sources were selected to search for each query. Results are averaged over 50 queries.

Document Rank	Trec123-100Col Testbed		Trec4_kmeans Testbed	
	CORI Merge	SSL Merge	CORI Merge	SSL Merge
5	0.3520	0.4040 (+14.8%)	0.2360	0.3720 (+57.6%)
10	0.3360	0.3700 (+10.1%)	0.1980	0.3160 (+59.6%)
15	0.3333	0.3627 (+8.8%)	0.1893	0.2853 (+50.7%)
20	0.3210	0.3470 (+8.1%)	0.1710	0.2520 (+47.4%)
30	0.3067	0.3233 (+5.4%)	0.1480	0.2133 (+44.1%)

Table 9. Precision at different document ranks using the CORI and Semi-Supervised Learning approaches to merging retrieval results from INQUERY, Language Model and Vector Space search engines. 10 sources were selected to search for each query. Results are averaged over 50 queries.

Document Rank	Trec123-100Col Testbed		Trec4_kmeans Testbed	
	CORI Merge	SSL Merge	CORI Merge	SSL Merge
5	0.3520	0.4320 (+22.7%)	0.2360	0.3640 (+54.3%)
10	0.3500	0.4080 (+16.6%)	0.1860	0.3220 (+73.1%)
15	0.3453	0.4013 (+16.2%)	0.1733	0.2933 (+69.2%)
20	0.3390	0.3820 (+12.7%)	0.1630	0.2720 (+66.8%)
30	0.3287	0.3627 (+10.3%)	0.1460	0.2400 (+66.4%)

sampled documents), the longer the queries (the description queries on the Trec4-kmeans testbed are longer than the title queries on the Trec123-100Col testbed), the more number of overlap documents can be acquired.

5.3.3 Experimental results: Comparison with CORI results merging formula

Experiments were conducted to compare the accuracy of the CORI results merging formula and the SSL results merging algorithm with the multiple engine-types case on the two testbeds of Trec123-100Col and Trec4-kmeans. Three retrieval algorithms of INQUERY, Language Model or Vector Space models were assigned to the information sources in a round-robin manner. In order to report thorough results, the number of information sources to be searched for each query was varied by 3, 5 or 10. The detail experiments results are shown in Tables 7 to 9.

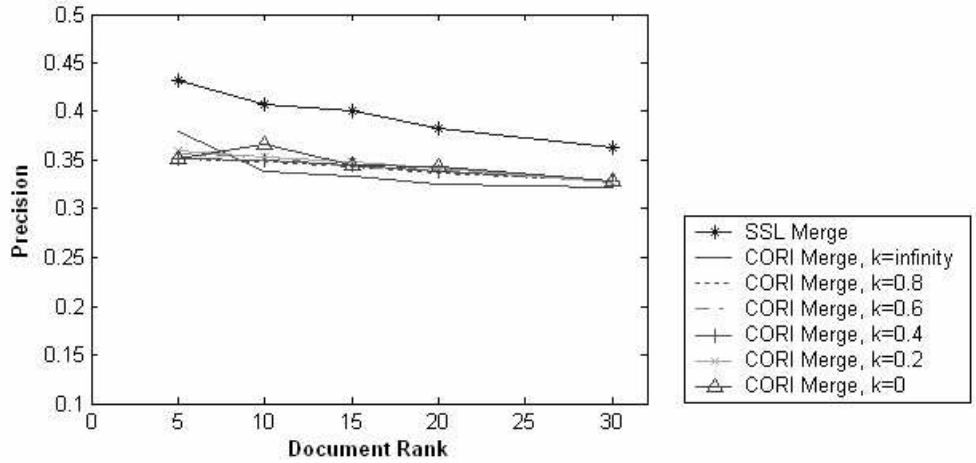

Figure 8. How varying the k parameter in the CORI result merging formula affects Precision on the Trec123-100Col testbed.

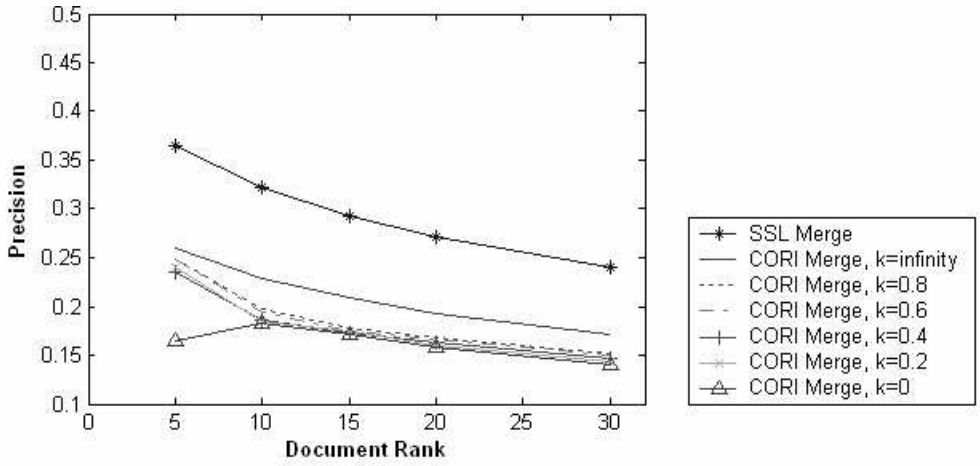

Figure 9. How varying the k parameter in the CORI result merging formula affects Precision on the Trec4-kmeans testbed.

Several interesting issues can be observed from these experiment results. First, the SSL result merging algorithm was more accurate than the CORI merging formula in all configurations on these two testbeds. Particularly, the advantage of the SSL algorithm against CORI formula was much more notable on the Trec4-kmeans testbed than on the Trec123-100Col testbed. It is known that the Trec4-kmeans testbed has more skewed word distribution than the Trec123-100Col testbed so that the information source statistics on the Trec4-kmeans testbed are more diverse than the Trec123-100Col testbed. This causes a serious problem for the CORI merging formula, which was also pointed out in previous research (Larkey, Connel & Callan, 2000). However, the SSL algorithm addresses this problem by building multiple models for different selected information sources. This strategy automatically corrects the skewed word distribution and has been shown to be effective in the experiments.

Second, on the Trec123-100Col testbed, both the SSL algorithm and the CORI formula tended to be more effective as more information sources were searched. In contrast, on Trec4-kmeans testbed, although the accuracy of the SSL algorithm has been improved by selecting more information sources, the accuracy of the CORI algorithm was deteriorated slightly. We attribute this to the fact that the contents of information sources on Trec4-kmeans testbed are much more homogenous than that on the Trec123-100Col testbed. Therefore, the relevant documents for a query are distributed in much fewer information sources on the Trec4-kmeans testbed than on the Trec123-100Col testbed. Relatively more irrelevant documents from

"bad" information sources were introduced by selecting more information sources on the Trec4-kmeans testbed. CORI result merging formula suffered from those irrelevant documents while the SSL algorithm was still able to gain advantage by distinguishing the relatively few relevant documents from much more irrelevant documents in the information sources of lower qualities.

Third, the advantage of the SSL algorithm against the CORI merging formula was generally larger at the top part of the ranked list (5 or 10) than the lower part (15, 20 or 30). This is exactly what we favor and is consistent with our strategy in the SSL algorithm to put more weights on the top returned documents (e.g., select up to top 10 overlap documents as training data for each information source).

Another set of experiments was conducted to study the effectiveness of the CORI results merging formula more carefully. The CORI results merging formula in Equation 5.3 can be rewritten in another way as a linear function of the information source specific document scores:

$$S_C(d_{ij}) = \frac{S(d_{ij})'(1+k*S(d_i)')}{1+k} \tag{5.12}$$

where k measures the importance of information source selection score, and it is set to 0.4 by default in the CORI results merging formula. The linear function only has the slope as $(1 + k*S(d_i)')/(1 + k)$ and has no intercept. Therefore, Equation 5.12 can be seen as a special case of the linear transformation model in the SSL algorithm.

The experiments were done to study the accuracy of the CORI merging formula with different k values (when k is set to infinity, the formula is $S_c(d_{ij}) = S(d_{ij})' * S(d_i)'$). The results are shown in Figures 8 and 9.

It can be seen from Figure 8 that the adjustment of parameter k did not cause much difference to the effectiveness of the CORI merging algorithm on the Trec123-100Col testbed. On the Trec4-kmeans testbed, the CORI merge formula with an infinity value of k had a slight advantage against other values as shown in Figure 9, which can be explained by the fact that a much larger proportion of relevant documents are in the top few selected information sources on the Trec4-kmeans testbed than on the Trec123-100Col testbed. Therefore, there was no particular choice of k that can improve the accuracy consistently in the experiments. Although the parameter k in the CORI merging formula was tuned in the experiments, essentially linear models with only one parameter of slope is applied to the selected information sources, and thus the linear transformation models across the information sources were associated with each other by the value k and the information source selection scores. In contrast, the SSL algorithm does allow each selected information source to choose its own query-specific and information source specific linear transformation model to achieve better accuracy. The results of experiments in Figures 8 and 9 show that the SSL algorithm was more effective than all the variants of CORI merging formula. Therefore, we tend to conclude that the power of the SSL algorithm derives from its ability of adjusting the linear transformation models source-by-source and query-by-query.

5.3.4 Experimental results: The effect of returned ranked list length

In Section 5.3.1, experiments have shown that the SSL algorithm is very likely to have enough overlap documents when the information sources provide long returned ranked document lists. Let us assume an information source provides a ranked list of 1,000 document ids and their scores by a single interaction with the central search client. If the corresponding communication cost can be estimated by 80 bytes per document (60 bytes for document id and 20 bytes for document score) for each of the 1,000 documents, the total communication cost of this ranked list is about 80,000 bytes and it is not excessive for the network bandwidth nowadays.

However, in real uncooperative environments of federated document retrieval systems, information sources may only provide limited ranked list (top 10 or 20) for the initial search request and additional interactions are needed to obtain the results farther down the list. If an information source returns a page of ranked list containing 20 document ids, it requires a total number of 50 interactions to obtain the ranked list of 1,000

Table 10. The percentage of selected information sources that are short of overlap documents for various lengths of ranked lists on Trec123-100Col and Trec4-kmeans testbeds. 10 information sources were selected per query. Results are averaged over 50 queries.

Results List Length	Percentage of Selected Information Sources with Fewer than 3 Overlap Documents	
	Trec123-100Col	Trec4-kmeans
50	54.2%	17.4%
100	27.8%	5.6%
200	10.4%	2.6%
500	6.0%	0.0%
1000	5.2%	0.0%

Table 11. The average number of downloaded documents to meet the requirement of at least 3 overlap documents per selected information sources on Trec123-100Col and Trec4-kmeans testbeds. 10 Information sources were selected per query. Results are averaged over 50 queries.

Result List Length	Trec123-100Col		Trec4-kmeans testbed	
	Download Docs Per Source	Total Download Docs (10 Sources)	Download Docs Per Source	Total Download Docs (10 Sources)
50	1.0	10.2	0.3	3.3
100	0.4	4.4	0.1	1.1

documents. This is exactly our argument against the Capture-Recapture information source size estimation algorithm in Chapter 2, and we regard it as excessive communication cost.

A set of experiments was conducted to study how many information sources are short of overlap documents for training data with various lengths of ranked document lists. The results are shown in Table 10. It can be seen from Table 10 that most information sources have enough overlap documents with long ranked lists (500 or 1,000) on both these two testbeds, which is consistent with the experiments in Section 5.3.2. However, with shorter and shorter ranked list, the percentage of information sources without enough overlap documents is growing dramatically. This is especially severe for the cases on the Trec123-100Col testbed with ranked lists of 50 or 100 documents.

An alternative approach is to download a minimum amount of returned document on the fly and intentionally create the overlap documents as the training data. If we measure the communication costs by the number of interactions with information sources, this method may substantially reduce the cost without degrading the accuracy.

Specifically, when a selected information source does not provide enough overlap documents (fewer than 3), the rest of required overlap documents are downloaded on the fly and their centralized document scores are calculated with the corpus statistics of the centralized sample database to create the training data. Documents ranked at 1^{st}, 10^{th} and 20^{th} in the result lists are the candidates for downloading. These documents are chosen as they cover a relatively wide range of the top ranked documents in the ranked lists, which are most important for federated document retrieval systems.

Experiments were conducted to study the amount of downloaded documents with short ranked lists of 50 or 100 documents and the results are shown in Table 11. In all these experiments, 10 information sources were selected to search. It can be seen that only 0.3-1.0 documents were required to download per information source by average when information sources returned ranked lists of 50 documents, and even fewer 0.1-0.4 documents were required with ranked lists of 100 documents. In both these two configurations, the number of interactions for a selected information source to download and acquire enough overlap documents was much fewer than the approach of obtaining long ranked lists such as 500 or 1,000 documents.

Table 12. Precision at different document ranks for three methods of obtaining enough overlap documents for the SSL results merging algorithm. Three types of search engines were used. 3 sources were selected to search for each query. Results are averaged over 50 queries.

Document Rank	Trec123-100Col			Trec4-Kmeans		
	List of 1,000 Docs	List of 50 Docs + Downloads	List of 100 Docs + Downloads	List of 1,000 Docs	List of 50 Docs + Downloads	List of 100 Docs + Downloads
5	0.3680	0.3640 (-1.1%)	0.3800 (+3.3%)	0.3440	0.3440 (0.0%)	0.3440 (0.0%)
10	0.3420	0.3360 (-1.8%)	0.3440 (+0.6%)	0.2840	0.2940 (+3.5%)	0.2980 (+4.9%)
15	0.3253	0.3253 (0.0%)	0.3320 (+2.1%)	0.2520	0.2547 (+1.1%)	0.2520 (0.0%)
20	0.3090	0.3140 (+1.6%)	0.3060 (-1.0%)	0.2260	0.2220 (-1.8%)	0.2270 (+0.4%)
30	0.2747	0.2780 (+1.2%)	0.2733 (-0.5%)	0.1993	0.1913 (-4.0%)	0.2000 (+0.4%)

Table 13. Precision at different document ranks for three methods of obtaining enough overlap documents for the SSL results merging algorithm. Three types of search engines were used. 5 sources were selected to search for each query. Results are averaged over 50 queries.

Document Rank	Trec123-100Col			Trec4-Kmeans		
	List of 1,000 Docs	List of 50 Docs + Downloads	List of 100 Docs + Downloads	List of 1,000 Docs	List of 50 Docs + Downloads	List of 100 Docs + Downloads
5	0.4040	0.4000 (-1.0%)	0.4120 (+2.0%)	0.3720	0.3440 (-7.5%)	0.3640 (-2.2%)
10	0.3700	0.3800 (+2.7%)	0.3920 (+6.0%)	0.3160	0.3040 (-3.8%)	0.3240 (+2.5%)
15	0.3627	0.3560 (-1.9%)	0.3653 (+0.7%)	0.2853	0.2560 (-10.2%)	0.2760 (-3.3%)
20	0.3470	0.3430 (-1.2%)	0.3470 (0.0%)	0.2520	0.2260 (-10.3%)	0.2440 (-3.2%)
30	0.3233	0.3240 (+0.2%)	0.3213 (-0.6%)	0.2133	0.1940 (-9.1%)	0.2073 (-2.8%)

Table 14. Precision at different document ranks for three methods of obtaining enough overlap documents for the SSL results merging algorithm. Three types of search engines were used. 10 sources were selected to search for each query. Results are averaged over 50 queries.

Document Rank	Trec123-100Col			Trec4-Kmeans		
	List of 1,000 Docs	List of 50 Docs + Downloads	List of 100 Docs + Downloads	List of 1,000 Docs	List of 50 Docs + Downloads	List of 100 Docs + Downloads
5	0.4320	0.4440 (+2.8%)	0.4360 (+0.9%)	0.3640	0.3320 (-8.8%)	0.3560 (-2.2%)
10	0.4080	0.4300 (+5.4%)	0.4280 (+4.9%)	0.3220	0.3000 (-6.8%)	0.3420 (+6.2%)
15	0.4013	0.4187 (+4.3%)	0.4133 (+3.0%)	0.2933	0.2600 (-11.3%)	0.2960 (+0.9%)
20	0.3820	0.3980 (+4.2%)	0.3900 (+2.1%)	0.2720	0.2420 (-11.0%)	0.2650 (-2.6%)
30	0.3627	0.3653 (+0.7%)	0.3707 (+2.2%)	0.2400	0.2113 (-11.9%)	0.2380 (-0.8%)

The efficiency of downloading minimum amount of overlap documents has been proven in the last set of experiments. Another set of experiments was conducted to show its effectiveness. The experiment results in Tables 12-14 show the accuracy of several methods of obtaining enough overlap documents for the SSL results merging algorithm. These methods either requires long ranked lists (1,000 documents), which is associated with a large amount of communication cost whenever only a small piece of ranked list can be provided by a single interaction, or downloads the minimum number of documents to create overlap documents with much shorter ranked lists (50 or 100 documents). It can be seen that these methods were about the same effective on the Trec123-100Col testbed. Although, the method with long ranked lists had small advantage against the minimum downloading method with ranked lists of 50 documents on the Trec4-kmeans testbed where relatively more overlap documents can be obtained by the long ranked lists, the minimum downloading methods were still much more effective than the CORI merging formula

compared with the results shown in Tables 7-9. Furthermore, when the minimum downloading method is required to download more training documents so that each selected information source has at least 5 instead of 3 overlap documents the downloading method shows about the same accuracy as the SSL method with long ranked lists.

The above minimum downloading method of SSL algorithm uses the downloaded documents as training data to build the linear transformation models. Another utilization of these downloaded documents is to accumulate them in the centralized sample database as extra data for future queries. It can be imagined that when more and more downloaded documents have been accumulated in the centralized sample database, more training data is available for results merging and the requirement of downloading new document is reduced especially for queries on popular topics. Therefore, the long-term learning of accumulating downloaded documents with the minimum downloading method of SSL algorithm gives us an opportunity to increase results merging accuracy and reduce communication costs for future queries.

To summarize, generally the accuracy of the minimum downloading method of the SSL algorithm is comparable with that of the SSL algorithm with long ranked lists. Since the minimum downloading method could be much more efficient, this method is very practical to be applied in the environments where only short result lists are provided. Furthermore, it provides an opportunity to improve results merging accuracy and reduce communication costs in the long run by accumulating the downloaded documents as extra training data.

5.4 Future research

Several other issues should be addressed to investigate the effectiveness of the SSL results merging algorithm in real operational environments. First, in real operational environments, information sources may not return document scores with ranked document lists. Extension of the SSL results merging algorithm should be proposed to address this use scenario. Furthermore, in this chapter the minimum downloading method of Semi-Supervised Learning algorithm is evaluated with ranked lists that contain relatively small number of document ids such as 50 or 100 ids. However, in real operational environments, users may be interested in browsing ranked lists with even shorter sizes such as 10 or 20 documents. Experiments should be conducted to investigate the behavior of the SSL algorithm in this case. The detail of all these new research topics is discussed in Section 7.4.

Chapter 6: Preliminary Work on Unified Utility Maximization Framework

Federated search task is composed of three sub-problems, namely resource description, resource selection and results merging. The research problems of these three sub-problems were discussed seperately from Chapter 3 to Chapter 5. However, since these sub-problems are correlated with each other in federated search applications, exploring the relationship between these sub-problems is as imporant as proposing separate effective solutions. A unitifed probablistic framework is proposed in this chapter for federated search task in uncooperative environments. The new model integrates and adjusts individual solutions of different sub-problems to achieve effective results for different applications. Specifically, when used for information source recommendation system, the model targets the high-recall goal (select a small number of information sources with as many relevant documents as possible); when used for federated document retrieval, the model is optimized for the high-precision goal (high precision in the top part of final merged lists). The new research proposed in previous chapters for individual sub-problems of federated search task is utilized and integreated to the new framework. Emprically results also show the effectiveness of this framework.

The prior research most similar to the research proposed in this chapter is the decision-theoretic framework (DTF) (Fuhr, 1999)(Nottelmann & Fuhr, 2003b). This method yields an information source selection that minimizes a cost function of overall costs (e.g., retrieval accuracy, query processing cost and communication cost) for federated document retrieval application. When it is focused on retrieval accuracy, the DTF model estimates the probabilities of relevance for the documents among available information sources, and then it generates a resource selection decision for the high-precision goal. However, its empirical results have been shown to be at most as good as that of the CORI resource selection algorithm with the same experiment setting (Nottelmann & Fuhr, 2003b). Three issues limit the power of the DTF algorithm: i) the DTF model was proposed for federated document retrieval application and does not address the high-recall goal of information source recommendation application explicitly; ii) the DTF model assumes that the same type of retrieval algorithm is used by all the information sources, which is not true in uncooperative environments; and iii) the DTF model builds separate models for each information source to estimate the probabilities of relevance. This requires human relevance judgments for the results retrieved from each information source, which is excessive amount of human efforts if there are many information sources. In contrast, the unified utility maximization framework proposed in this chapter integrates the goals of high-recall and high-precision in a single probabilistic model. It works in uncooperative environments and is much more efficient than the DTF model.

6.1 High recall of information source recommendation vs. high precision of document retrieval

Information source recommendation and federated document retrieval are two important applications of federated search task, and there has been considerable research on both of these two applications.

Information source recommendation system is composed of resource description and resource selection. It recommends information sources for users' information need; this system is very useful when the users want to browse the selected information sources manually for broader contents instead of asking the system

to retrieve relevant documents automatically. Most current resource selection algorithms are evaluated and designed for effective results of the information source recommendation application, which is to recommend a small number of information sources that contain as many relevant document as possible (the goal of high-recall).

On the contrary, in federated document retrieval application the selected information sources are searched and the individual results are automatically merged into a final ranked list. Therefore, all the three sub-problems of federated search need to be addressed in this application. Since the user is presented with a single merged list for each query, the federated document retrieval application is often evaluated with the Precision at the top ranks in the final list (the goal of high-precision).

In order to achieve accurate results for federated document retrieval systems, most previous methods simply combine effective resource selection algorithms and results merging algorithms together. However, these simple approaches suffer from an important fact that resource selection algorithm optimized for the high-recall goal of the information source recommendation application is not necessarily optimal for the high-precision goal of the federated document retrieval application. This type of inconsistency has also been observed in previous research (Craswell, 2000).

6.2 Unified utility maximization framework

Our solution to address this inconsistency is a unified utility maximization framework that integrates the two applications of information source recommendation and federated document retrieval together by assigning them different optimization goals. Specifically, a logistic transformation model is learned off line with a small amount of training queries to map the centralized document scores in the centralized sample database to the corresponding probabilities of relevance. For each query, the probabilities of relevance of all the (mostly unseen) documents among the available information sources can be inferred from the probabilities of relevance of the sampled documents and the corresponding information source size estimates. Based on these probabilities, the information sources are selected according to either the high-recall goal or the high-precision goal with respect to different applications. Furthermore, for the application of federated document retrieval, the SSL (Semi-Supervised Learning) results merging algorithm is utilized to rank the returned documents by their estimated centralized document scores (thus also the probabilities of relevance as we assume the mapping between the centralized document scores and the probabilities of relevance is monotonically increasing).

In this section, we first discuss how to estimate the probabilities of relevance for all the documents, and then show how to apply the framework for the information source recommendation application and the federated document retrieval application respectively.

6.2.1 Estimate probabilities of relevance for all documents across the information sources

The key issue to meet either the goal of high-recall in information source recommendation application or the goal of high-precision in federated document retrieval application is to estimate the probabilities of relevance for all the documents across the information sources. This is a hard problem as we can only observe very limited proportion of all the contents in the information sources by query-based sampling. Our strategy is to make full use of all the available information to accomplish this hard task. Specifically, we build one centralized logistic transformation model on the centralized sample database to calculate the probabilities of relevance for the sampled documents, and then make the centralized sample database serve as the bridge to connect the centralized transformation model and the information sources to estimate the probabilities of relevance for all the documents across the information sources.

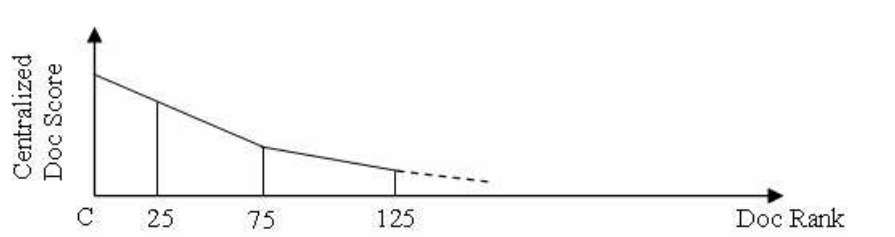

Figure 10. The complete centralized document score curve constructed by linear interpolation for a particular information source with scale factor of 50.

6.2.1.1 Estimate centralized document scores

The unified utility maximization framework uses a centralized logistic transformation model to map centralized retrieval scores to the corresponding probabilities of relevance for all the documents. Therefore, it is necessary to estimate the centralized document scores for all the (mostly unseen) documents and these scores are inferred from the centralized retrieval scores of the documents in the centralized sample database and the information source size estimates.

The concept of information source size factor is introduced in Chapter 4. It is associated with a particular i^{th} information source and defined as the ratio of its estimated information source size and the number of sampled documents from this information source. Formally as:

$$SF_{db_i} = \frac{\hat{N}_{db_i}}{N_{db_i\_samp}} \tag{6.1}$$

where $\hat{N}_{db_i}$ denotes the information source size estimate for the i^{th} information source and N_{dbi_samp} denotes the number of sampled documents from this information source.

The idea behind the information source size factor is that: for a particular information source with information source size factor of 50, if a sampled document from this information source has a centralized document score of 0.8, it can be roughly estimated that there are another 49 unseen documents in the information source that have the centralized document scores of 0.8 as long as we assume the sampled documents are representative. Instead of this simple histogram non-parametric estimator, we can choose a finer piecewise linear estimator. Formally, all the sampled documents from the particular i^{th} information sources are first ranked according to their centralized document scores to get the sampled centralized document score list as $\{S_c(ds_{i1}), S_c(ds_{i2}), S_c(ds_{i3}),.....\}$. Suppose that we can calculate the centralized document scores for all the documents from this information source to obtain the complete centralized document score list, we assume that the top document in the sampled list would rank at the position of $SF_{dbi}/2$ in the complete list, the second document in the sampled list would rank at the position of $SF_{dbi}3/2$ in the complete list, and so on. Therefore, a set of sampled data points can be obtained from the complete centralized document score curve as $\{(SF_{dbi}/2, S_c(ds_{i1})), (SF_{dbi}3/2, S_c(ds_{i2})), (SF_{dbi}5/2, S_c(ds_{i3})),.....\}$. Finally, the complete centralized document score curve can be estimated with linear interpolation approach as indicated in Figure 10, and the whole complete centralized document score list $\hat{S}_c(d_{ij}),\ j \in [1, \hat{N}_{db_i}]$ can be extracted from the curve accordingly.

It can be noted that the more sampled data points that are available the more accurate complete centralized document score list can be estimated. On the contrary, with very sparse sampled data points, the estimation of the complete list may be inaccurate especially for the top ranked documents (e.g., $[1, SF_{dbi}/2]$), on which we put more emphasis as they are more probable to be relevant. Based on this observation, an alternative approach is proposed to adjust the estimated centralized document scores of the top ranked documents for the information sources with large information source scale factors (e.g., larger than 100). Specifically, a

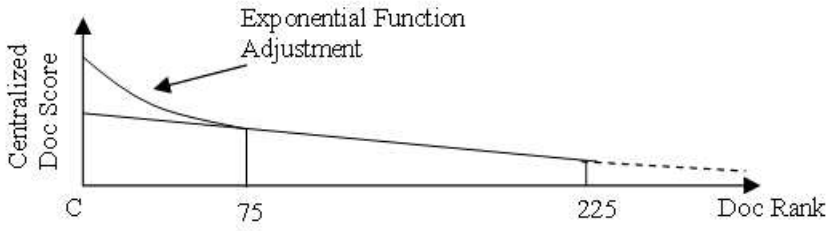

Figure 11. The adjusted centralized document score curve for a large information source with scale factor of 150.

logistic transformation model is learned for each information source with a large information source scale factor to estimate the centralized document score of the top 1 document by the centralized document scores of the top two sampled documents from this information source as:

$$S_c(\hat{d}_{i1}) = \frac{\exp(\alpha_{i0}+\alpha_{i1}S_c(ds_{i1})+\alpha_{i2}S_c(ds_{i2}))}{1+\exp(\alpha_{i0}+\alpha_{i1}S_c(ds_{i1})+\alpha_{i2}S_c(ds_{i2}))} \quad (6.2)$$

where $S_c(\hat{d}_{i1})$ is the estimated centralized document score of the top 1 document in the i^{th} information source. a_{i0}, $a_{i1}$and a_{i2} are the three parameters of the corresponding logistic model. For each of the information source with large information source scale factor, the top retrieved documents for all training queries are downloaded and their centralized document scores are calculated. With the centralized document scores of the top two sampled documents for the queries, the three parameters of the logistic model can be estimated.

After a logistic transformation model has been built for an information source with a large information source scale factor, an exponential function is fitted for the top part ([1, $SF_{dbi}/2$]) of the complete centralized document score curve as follow:

$$S_c(\hat{d}_{ij}) = \exp(\beta_{i0}+\beta_{i1}*j) \quad j \in [1, SF_{db_i}/2] \quad (6.3)$$

$$\beta_{i0} = \log(S_c(\hat{d}_{i1})) - \beta_{i1} \quad (6.4)$$

$$\beta_{i1} = \frac{(\log(S_c(ds_{i1}) - \log(S_c(\hat{d}_{i1}))}{(SF_{db_i}/2-1)} \quad (6.5)$$

The two parameters β_{i0} and β_{i1} are chosen to make the exponential function pass through the two points of $(1, S_c(\hat{d}_{i1}))$ and ($SF_{dbi}/2$, $S_c(ds_{i1})$). Only the top part ([1, $SF_{dbi}/2$]) of the curve is adjusted by the exponential function and all the other parts are still fitted with the linear interpolation approach. This adjustment is shown in Figure 11.

Empirical studies show that this type of score adjustment for large information sources produces more accurate results. Exploring more direct evidences such as plotting the fitting results of the centralized document score curves is an interesting topic for the future research.

By now, the problems of calculating the probabilities of relevance with centralized document scores and estimating the centralized document scores for all the documents across the available information sources

have been addressed. Therefore, the centralized document scores for all the documents can be obtained and their corresponding probabilities of relevance can be further estimated. Formally for a particular i^{th} information source, the complete list of probabilities of relevance can be expressed as: $\hat{R}(\hat{d}_{ij}),\ j \in [1, \hat{N}_{db_i}]$.

6.2.1.2 From centralized document scores to probablities of relevance

There are several alternative approaches that can map centralized retrieval document scores to the corresponding probabilities of relevance such as linear transformation, logistic transformation of raw centralized scores and logistic transformation of normalized centralized scores. Prior research (Nottelmann & Fuhr, 2003a)(Nottelmann & Fuhr, 2003b) has measured the error of these methods and has shown the transformation of normalized centralized scores to be the most effective one. A centralized logistic transformation model is used in this work to map the normalized centralized retrieval score of a document to its corresponding probability of relevance. Formally, the logistic transformation model is expressed as follows:

$$R(d) = P(rel|d) = \frac{\exp(a_c + b_c \bar{S}_c(d))}{1+\exp(a_c + b_c \bar{S}_c(d))} \tag{6.6}$$

where $\bar{S}_c(d)$ denotes the normalized centralized document score for a particular document d. a_c and b_c are the two parameters of the logistic model. These two parameters are estimated by maximizing the probabilities of relevance of all the training queries. More specifically, in order to create training data, a small amount of queries (e.g., 50) are utilized to search on the centralized sample database by the INQUERY retrieval algorithm (Callan, Croft & Broglio, 1995). The CORI resource selection algorithm is used to rank the information sources and 10 selected information sources are searched for each query (the choice of INQUERY and CORI is not unique, we believe other effective retrieval methods like language model algorithm and other resource selection algorithms like ReDDE are also good candidates here). The individual ranked lists for each query are merged into a single list by the SSL results merging algorithm, where the top 50 documents are downloaded and the corresponding centralized scores are calculated by the INQUERY algorithm with the corpus statistics of the centralized sample database. Then the centralized scores are normalized by dividing the maximum centralized score for each query, as this approach has been suggested to improve estimation accuracy in previous research (Nottelmann & Fuhr, 2003a). Human relevance judgments are further acquired for these documents.

Note that only a single centralized logistic transformation model is built and the centralized sample database serves as the bridge to connect the centralized transformation model and all the information sources to estimate the probabilities of relevance for all the documents across the information sources. The human efforts required to train the single centralized logistic model do NOT scale with the number of databases. This makes a big distinction between the unified utility maximization model and the prior research that required building separate models for different information sources such as the RDD (Voorhees et al., 1995) or the DTF (Fuhr, 1999)(Nottelmann & Fuhr, 2003b) methods. This advantage shows that unified utility maximization can be trained much more efficiently and is much easier to be applied in large federated research environments with many information sources.

6.2.2 Unified utility maximization model

In the application of information source recommendation, the systems only rank the information sources in resource selection, while the federated document retrieval systems also need to decide how many documents to retrieve from each selected information source. The resource selection action of the information source recommendation application can be generalized, which implicitly recommends all the documents in the selected information sources, as a special case of the resource selection action of the federated document retrieval application. Formally, let d_i denote the number of documents to select from a

particular i^{th} information source and the vector of $\vec{d}=\{d_1,d_2,.....\}$ denote the selection decision across the available information sources.

To obtain effective results of federated search applications, we refer to the complete lists of probabilities of relevance that are derived in the Section 6.2.1. Those lists are inferred from all the available information, namely $\overrightarrow{R_s}$ which stands for the resource descriptions acquired by query-based sample and information source size estimates obtained by the Sample-Resample method, and $\overrightarrow{S_c}$ which stands for the centralized document scores of all the sampled documents.

If both the methods of estimating the centralized document scores and the probabilities of relevance are acceptable, the most probable complete lists of probabilities of relevance can be derived and we denote them as $\theta^* = \{(R(\hat{d}_{1j}), j \in [1, \hat{N}_{db_1}]),\ (R(\hat{d}_{2j}), j \in [1, \hat{N}_{db_2}]), ...\}$. Random vector θ denotes an arbitrary set of the complete lists of probabilities of relevance and $P(\theta|\overrightarrow{R_s},\overrightarrow{S_c})$ denotes its corresponding generation probability. Furthermore, the utility function $U(\theta,\vec{d})$ is defined as the benefit that can be gained by making the selection action $\vec{d}$ when the complete lists of probabilities of relevance are θ. Finally, the desired selection action derived from the Bayesian framework can be expressed as:

$$\vec{d}^{\,*} = \underset{\vec{d}}{\operatorname{argmax}} \int_{\theta} U(\vec{d},\theta)\, P(\theta|\overrightarrow{R_s},\overrightarrow{S_c})\, d\theta \tag{6.7}$$

However, it is not easy to derive an accurate expression for $P(\theta|\overrightarrow{R_s},\overrightarrow{S_c})$; even when there exists one, the computation costs are generally not acceptable as there are infinite choices of θ. An alternative approach to simplify the computation in the Bayesian framework is to calculate the utility function $U(\theta^*,\vec{d})$ at the most probable parameter values instead of calculating the whole expectation. Formally, as:

$$\vec{d}^{\,*} = \underset{\vec{d}}{\operatorname{argmax}}\, U(\vec{d},\theta^*) \tag{6.8}$$

This equation serves as the basic model from which the desired selection actions can be derived for both the information source recommendation application and the federated document retrieval application.

6.2.2.1 Optimal resource selection for information source recommendation application

Similar to the ReDDE resource selection algorithm described in Chapter 4, the new resource selection algorithm proposed here are also optimized for the high-recall goal of the information source recommendation application. Furthermore, both of them turn away from the "big document" resource selection approach and explicitly estimate the probabilities of relevance for all documents across the available information sources to calculate the expected number of relevant documents. However, the key point that distinguishes these two methods lies in the different ways to estimate the probability of relevance for each document. Specifically, the ReDDE uses a heuristic method and treats the curve of probabilities of relevance as a step function, where only documents' ranks in the centralized complete database are considered; while the new algorithm takes advantage of the training data and builds logistic model to transfer the centralized document scores to their corresponding probabilities of relevance. More specifically, the selection criterion of this new selection algorithm is formalized as follows.

It was pointed out that the high-recall goal of the information source recommendation system is to select a small number of information sources (e.g., N_{sdb} information sources) that contain as many relevant documents as possible. This criterion can be formalized as follows:

$$U(\vec{d},\theta^*) = \sum_i I(d_i) \sum_{j=1}^{\hat{N}_{db_i}} R(\hat{d}_{ij}) \tag{6.9}$$

where $I(d_i)$ is a binary indication function and denotes whether a particular i^{th} information source is selected or not. Plug this utility function into the basic model in Equation 6.8 and associate it with the constraint of the number of selected information sources, the following optimization problem can be obtained:

$$\vec{d}^* = \arg\max_{\vec{d}} \sum_i I(d_i) \sum_{j=1}^{\hat{N}_{db_i}} R(\hat{d}_{ij}) \quad \text{Subject to: } \sum_i I(d_i) = N_{sdb} \tag{6.10}$$

As the contribution of the information source is not coupled with each other (the number of relevant documents an information source contains is not affected by another information source), the solution of the above optimization problem is very simple and the expected number of relevant documents in each information source can be calculated as follows:

$$Rel\_Q(i) = \sum_{j=1}^{\hat{N}_{db_i}} R(\hat{d}_{ij}) \tag{6.11}$$

The information sources are ranked with respect to the number of relevant documents they contain and the top N_{sdb} information sources can be selected to obtain the high-recall results. This method is called the UUM/HR algorithm (Unified Utility Maximization for High-Recall).

6.2.2.2 Optimal resource selection for federated document retrieval application

It has been discussed how to apply the complete lists of probabilities of relevance for the information source recommendation application to achieve the high-recall goal. However, for the federated document retrieval application, the situation is more complex and we need to address two additional problems: i) documents with high probabilities of relevance may not be retrieved by the search engines of individual information sources; and ii) when they are retrieved, they may not be ranked highly in the final merged result list. First, it is assumed that all the individual search engines are effective to address the first problem (the case when this assumption is relaxed will be discussed later). Second, the SSL (Semi-Supervised Learning) results merging algorithm can automatically transform information source specific scores to the centralized document scores and rank all the returned documents accordingly. We assume that it can obtain the centralized document scores with high accuracy, so the final result list created by SSL algorithm is actually sorted by their centralized document scores (thus the probabilities of relevance). Therefore, the complete lists of probabilities of relevance can also be applied for the federated document retrieval application.

The accuracy of a federated document retrieval system is measured by the high-precision criterion as the Precision at the top part of the final merged document list. Thus, the utility function for federated document retrieval should reflect this high-precision goal as follows:

$$U(\vec{d},\theta^*) = \sum_i I(d_i) \sum_{j=1}^{d_i} R(\hat{d}_{ij}) \tag{6.12}$$

Note that the difference between this utility function and the utility function of information source recommendation application in Equation 6.9 is that for information source recommendation application, the probabilities of relevance of all the documents in an information source is summed together, while here a much smaller proportion of the documents are considered.

By combining the basic model in Equation 6.8 and the above utility function, the resource selection decision of federated document retrieval application can be obtained as follows:

$$\vec{d}^{*} = \arg\max_{\vec{d}} \sum_{i} I(d_i) \sum_{j=1}^{d_i} \hat{R}(d_{ij}) \tag{6.13}$$

With the same optimization goal, different constraints caused by applications with different characteristics are proposed. One simple configuration, which is consistent with the settings of most previous federated document retrieval research, is to select a fixed number (e.g., N_{sdb}) of information sources and retrieve a fixed number (e.g., N_{rdoc}) of documents from each selected information source as follows:

$$\begin{aligned} &\vec{d}^{*} = \arg\max_{\vec{d}} \sum_{i} I(d_i) \sum_{j=1}^{d_i} \hat{R}(d_{ij}) \\ &\text{Subject to: } \sum_{i} I(d_i) = N_{sdb} \\ &d_i = N_{rdoc}, \text{ if } I(d_i) \neq 0 \end{aligned} \tag{6.14}$$

This optimization problem can be easily solved by estimating the number of relevant documents in the top part of each information source's complete list of probabilities of relevance as follows:

$$Rel\_Q(i) = \sum_{j=1}^{N_{rdoc}} \hat{R}(d_{ij}) \tag{6.15}$$

Finally, the N_{sdb} information sources with the highest $Rel\_Q(i)$ values are selected and searched. As this algorithm is optimized for the high-precision goal and retrieves fixed lengths of document rank lists from the selected information sources, it is called the UUM/HP-FL algorithm.

The above configuration retrieves equal number of documents from the selected information sources. It can be imagined that information sources of high qualities are able to contribute more relevant documents than the information sources of relatively low qualities. Based on this observation, another configuration is proposed for federated document retrieval application to vary the number of retrieved documents from the selected information sources. Specifically, the selected information sources are allowed to return ranked document lists of different lengths. The lengths of the ranked lists are required to be multiples of a baseline number (which is set to 10 in this work for simplification to reduce the computation cost of the dynamic programming algorithm described in Figure 12). For further simplification, each selected information source is allowed to return at most 100 documents. Finally, the optimization problem of this configuration can be formalized as follows:

$$\begin{aligned} &\vec{d}^{*} = \arg\max_{\vec{d}} \sum_{i} I(d_i) \sum_{j=1}^{d_i} \hat{R}(d_{ij}) \\ &\text{Subject to: } \sum_{i} I(d_i) = N_{sdb} \\ &\sum_{i} d_i = N_{Total\_rdoc} \\ &d_i = 10 * k, \; k \in [0, 1, 2, .., 10] \end{aligned} \tag{6.16}$$

where N_{Total_rdoc} denotes the total number of documents to be retrieved.

This optimization problem is more complex and no simple solutions exist such as for the algorithms of UUM/HR or UUM/HP-FL. However, a dynamic programming solution is proposed to solve this optimization problem. The basic steps of the dynamic programming method are described in Figure 12.

Input: Complete lists of probabilities of relevance for all the |DB| information sources.

Output: Optimal selection solution for the optimization problem in Equation 6.16.

i) Create the data structure of a three-dimensional array:
Sel (1..|DB|, 1..$N_{Total\_rdoc/10}$, 1..N_{sdb})
Each Sel (x, y, z) is associated with a selection decision $\vec{d}_{xyz}$, which represents the best selection decision in the condition: only information sources from number 1 to number x are considered for selection; totally y*10 documents will be retrieved; only z information sources are actually chosen out of the x source candidates. And Sel (x, y, z) is the corresponding utility value by choosing the best selection.

ii) Initialize Sel (1, $N_{Total\_rdoc/10}$, 1..N_{sdb}) with only the complete list of probabilities of relevance for the 1st information source.

iii) Iterate the current database candidate i from 2 to |DB|
For each entry Sel (i, y, z):
Find k such that:

$$k^* = \arg\max_k \left(Sel(i-1,y-k,z-1) + \sum_{j \le 10*k} \hat{R}(d_{ij})\right)$$

subject to: $1 \le k \le \min(y,10)$

If $(Sel(i-1,y-k^*,z-1) + \sum_{j \le 10*k^*} \hat{R}(d_{ij})) > Sel(i-1,y,z)$
We should retrieve $10 * k^*$ documents from the i^{th} information source, then update the previous values of Sel (i-1, y, z) and set $\vec{d}_{iyz}$ accordingly.

If $(Sel(i-1,y-k^*,z-1) + \sum_{j \le 10*k^*} \hat{R}(d_{ij})) \le Sel(i-1,y,z)$
We should not select this information source and the previous best solution Sel (i-1, y, z) should be kept. Set $\vec{d}_{iyz}$ accordingly.

iv) The best selection solution is given by $\vec{d}_{|DB|N_{Toral\_rdoc/10}N_{sdb}}$ and the corresponding utility value is Sel (|DB|, $N_{Total\_rdoc/10}$, N_{sdb}).

Figure 12. The dynamic programming optimization procedure for Equation 6.16.

This selection approach is called UUM/HP-VL algorithm (Unified Utility Maximization for High-Precision with Variable Length ranked lists).

The optimization problems in Equations 6.14 and 6.16 solve the resource selection problems of federated document retrieval in two different configurations. Furthermore, the user queries are sent to search these selected information sources and the individual ranked lists are merged into a single list by the SSL algorithm. The SSL algorithm ranks the returned documents by their estimated probabilities of relevance as described before, which is consistent with our assumptions in the resource selection phrase.

6.3 Evaluation methodology and experimental results

In contrast to the previous algorithms described in this work, the utility maximization framework needs a small amount of training data with human relevance judgments to train its model. This section first explains the experimental methodology, and then shows the empirical results of the information source recommendation application and the federated document retrieval application respectively.

6.3.1 Experimental methodology

It was pointed out before that the utility maximization framework needs some amount of queries with relevance judgments as training data. The Trec123-100Col testbed was chosen as there are 100 TREC queries on this testbed, where 50 served as the training data and another 50 served as the test data in our experiments. Furthermore, the three testbeds of representative, relevant and nonrelevant were built based on the Trec123-100Col testbed (more detail in Chapter 2). All the four testbeds provided a wide range of corpus characteristics to conduct thorough experiments.

100 queries were created from the title fields of TREC topics 51-150. 50 queries from topics 101-150 served as the training queries and another 50 queries from topics 51-100 were used as the test queries.

To simulate the characteristics of uncooperative federated search environments, three types of search engines as INQUERY, Language Model and Vector Space model introduced in Chapter 2 were implemented with Lemur toolkit and assigned to the information sources in a round-robin manner.

Query-based sampling was used to acquire the resource descriptions and the centralized sample database. About 80 queries were sent to each information source to download 300 documents. The Sample-Resample method was used to obtain the information source size estimates.

6.3.2 Experimental results for information source recommendation application

The information source recommendation system suggests relevant information sources to users and it is typically evaluated using the recall metric R_n, which compares a particular algorithm with the relevance-based ranking strategy (more detail is Chapter 4).

Three resource selection algorithm for information source recommendation application, namely CORI, ReDDE (Relevant Document Distribution Estimation method) and UUM/HR, were evaluated on the four testbeds. The experiment results are shown in Figure 13.

It can be seen from this figure that the UUM/HR resource selection algorithm and the ReDDE algorithm were more effective (on the representative, relevant and nonrelevant testbeds) or as accurate as (on Trec123-100Col testbed) the CORI resource selection algorithm. The advantages of the UUM/HR algorithm and the ReDDE algorithm were more notable on the representative and relevant testbeds, where large information sources contain a large proportion of relevant documents and the CORI algorithm suffered substantially from the “big document” assumption without considering the information source size factors. This indicates that the power of the UUM/HR algorithm and the ReDDE algorithm comes from introducing information source size factors and explicitly estimating the probabilities of relevance for all documents across the available information sources to optimize the goal of high-recall.

Another observation can be drawn from Figure 13 is that the UUM/HR resource selection algorithm was more accurate than the ReDDE algorithm on the representative testbed and the relevant testbed, and it was about as effective as the ReDDE algorithm on the Trec123-100Col testbed and the nonrelevant testbed. This suggests that the UUM/HR algorithm is more robust than the ReDDE algorithm by introducing the centralized logistic model to estimate the probabilities of relevance and taking the advantage of the training data. However, careful analysis shows that when only a few information sources were selected on the Trec123-100Col testbed or on the nonrelevant testbed, the ReDDE algorithm had a small advantage over the UUM/HR algorithm. Two reasons can be used to explain this minor puzzle: i) the ReDDE algorithm was tuned on the Trec123-100Col testbed (set the optimal ratio value); and ii) although the difference is small, this may indicate that our centralized logistic model was not effective enough. More sophisticated model or more training data may help to solve this problem.

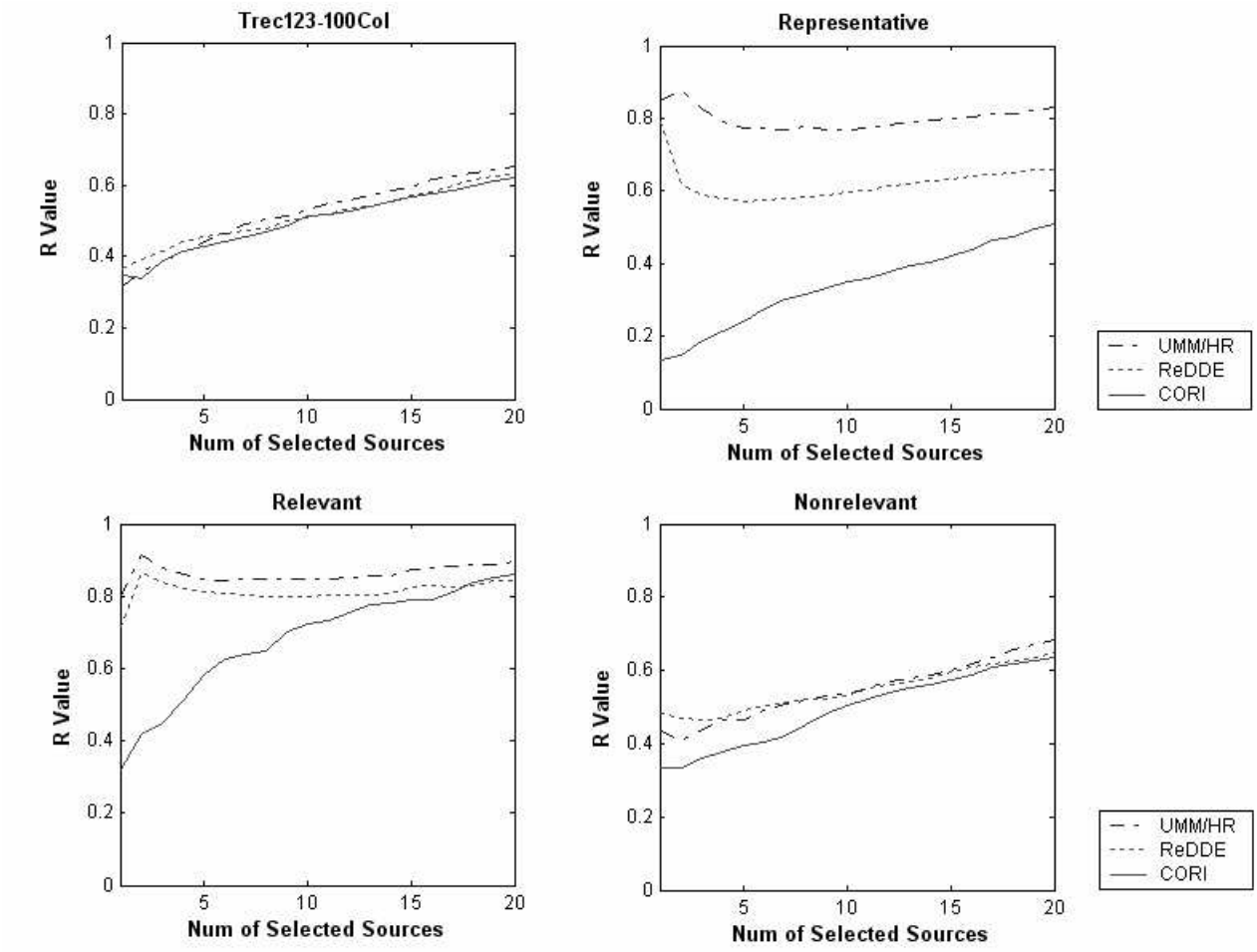

Figure 13. Resource selection experiments on four testbeds.

6.3.3 Experimental results for federated document retrieval application

For federated document retrieval, user queries are sent to search the selected information sources and the individual ranked lists are merged into a single list by the minimum downloading variant of the SSL algorithm which is described in Chapter 5. This type of SSL results merging algorithm downloads a small number of returned documents "on the fly" to create enough training data.

Experiments have been conducted to compare the five algorithms of CORI, ReDDE, UUM/HR, UUM/HP-FL and UUM/HP-VL. The Trec123-100Col and representative testbeds were selected as they represent two extreme cases, where the CORI resource selection algorithm is about as effective as the ReDDE algorithm and the UUM/HR algorithm for high-recall goal on the Trec123-100Col testbed and is much worse than the ReDDE and HMM/HR algorithms on the representative testbed. Three configurations were conducted on both the two testbeds to select 3, 5 or 10 information sources. The four algorithms of CORI, ReDDE, UUM/HR and UUM/HP-FL retrieved fix length of 50 documents from each selected information source, while the UUM/HP-VL algorithm was allowed to choose various lengths of documents from 10 to 100 as multiples of 10.

The experiments on the Trec123-100Col testbed and the representative testbed are shown in Tables 15-17 and Tables 18-20 respectively.

Table 15. Precision on the Trec123-100Col testbed when 3 sources were selected. (CORI is the baseline.)

Precision at Doc Rank	CORI	ReDDE	UMM/HR	UMM/HP-FL	UMM/HP-VL
5 docs	0.3640	0.3480 (-4.4%)	0.3960 (+8.8%)	0.4680 (+28.6%)	0.4640 (+27.5%)
10 docs	0.3360	0.3200 (-4.8%)	0.3520 (+4.8%)	0.4240 (+26.2%)	0.4220 (+25.6%)
15 docs	0.3253	0.3187 (-2.0%)	0.3347 (+2.9%)	0.3973 (+22.2%)	0.3920 (+20.5%)
20 docs	0.3140	0.2980 (-5.1%)	0.3270 (+4.1%)	0.3720 (+18.5%)	0.3700 (+17.8%)
30 docs	0.2780	0.2660 (-4.3%)	0.2973 (+6.9%)	0.3413 (+22.8%)	0.3400 (+22.3%)

Table 16. Precision on the Trec123-100Col testbed when 5 sources were selected. (CORI is the baseline.)

Precision at Doc Rank	CORI	ReDDE	UMM/HR	UMM/HP-FL	UMM/HP-VL
5 docs	0.4000	0.3920 (-2.0%)	0.4280 (+7.0%)	0.4680 (+17.0%)	0.4600 (+15.0%)
10 docs	0.3800	0.3760 (-1.1%)	0.3800 (0.0%)	0.4180 (+10.0%)	0.4320 (+13.7%)
15 docs	0.3560	0.3560 (0.0%)	0.3720 (+4.5%)	0.3920 (+10.1%)	0.4080 (+14.6%)
20 docs	0.3430	0.3390 (-1.2%)	0.3550 (+3.5%)	0.3710 (+8.2%)	0.3830 (+11.7%)
30 docs	0.3240	0.3140 (-3.1%)	0.3313 (+2.3%)	0.3500 (+8.0%)	0.3487 (+7.6%)

Table 17. Precision on the Trec123-100Col testbed when 10 sources were selected. (CORI is the baseline.)

Precision at Doc Rank	CORI	ReDDE	UMM/HR	UMM/HP-FL	UMM/HP-VL
5 docs	0.4400	0.4400 (0.0%)	0.4800 (+9.1%)	0.4680 (+6.4%)	0.4800 (+9.1%)
10 docs	0.4300	0.4080 (-5.1%)	0.4400 (+2.3%)	0.4520 (+5.1%)	0.4540 (+5.6%)
15 docs	0.4187	0.3840 (-8.3%)	0.4187 (+0.0%)	0.4320 (+3.2%)	0.4333 (+3.5%)
20 docs	0.3980	0.3750 (-5.8%)	0.3980 (+0.0%)	0.4040 (+1.5%)	0.4120 (+3.5%)
30 docs	0.3653	0.3513 (-3.8%)	0.3720 (+1.8%)	0.3820 (+4.8%)	0.3793 (+3.8%)

It can be observed from both the two sets of experiments that the difference between the accuracy of the algorithms got smaller when more information sources were selected. This is consistent with our expectation that the overlap in the selected information sources among different algorithms is much larger in the case of selecting many information sources than the case of selecting a small number of information sources.

More specifically, on the Trec123-100Col testbed, the document retrieval accuracy with the CORI selection algorithm was about the same or a little bit better than the ReDDE selection algorithm, while the UUM/HR algorithm had a small advantage over both of them. The main difference between the UUM/HR algorithm and the ReDDE algorithm is that the ReDDE uses a heuristic method to estimate the curve of the probabilities of relevance as a step function, while UUM/HR takes advantage of training data and builds a finer logistic model to transform the centralized document scores to the corresponding probabilities of relevance. This difference makes the UUM/HR better than the ReDDE algorithm at distinguishing the documents with high probabilities of relevance from documents with low probabilities of relevance to reflect the high-precision goal. Therefore, the UUM/HR is more robust and more accurate than the ReDDE algorithm. However, the advantage of the UUM/HR was small since it does not explicitly optimize the selection action according to the high-precision goal as what the UUM/HP-FL and UUM/HP-VL algorithms are designed to do. It can be noted from Tables 15-17 that the UUM/HP-FL and UUM/HP-VL algorithms were much more effective than the other algorithms when a small number of information sources were selected (3 or 5) and still had a small advantage with a large number of selected information

Table 18. Precision on the representative testbed when 3 sources were selected. (CORI is the baseline.)

Precision at Doc Rank	CORI	ReDDE	UMM/HR	UMM/HP-FL	UMM/HP-VL
5 docs	0.3720	0.4080 (+9.7%)	0.4640 (+24.7%)	0.4600 (+23.7%)	0.5000 (+34.4%)
10 docs	0.3400	0.4060 (+19.4%)	0.4600 (+35.3%)	0.4540 (+33.5%)	0.4640 (+36.5%)
15 docs	0.3120	0.3880 (+24.4%)	0.4320 (+38.5%)	0.4240 (+35.9%)	0.4413 (+41.4%)
20 docs	0.3000	0.3750 (+25.0%)	0.4080 (+36.0%)	0.4040 (+34.7%)	0.4240 (+41.3%)
30 docs	0.2533	0.3440 (+35.8%)	0.3847 (+51.9%)	0.3747 (+47.9%)	0.3887 (+53.5%)

Table 19. Precision on the representative testbed when 5 sources were selected. (CORI is the baseline.)

Precision at Doc Rank	CORI	ReDDE	UMM/HR	UMM/HP-FL	UMM/HP-VL
5 docs	0.3960	0.4080 (+3.0%)	0.4560 (+15.2%)	0.4280 (+8.1%)	0.4520 (+14.1%)
10 docs	0.3880	0.4060 (+4.6%)	0.4280 (+10.3%)	0.4460 (+15.0%)	0.4560 (+17.5%)
15 docs	0.3533	0.3987 (+12.9%)	0.4227 (+19.6%)	0.4440 (+25.7%)	0.4453 (+26.0%)
20 docs	0.3330	0.3960 (+18.9%)	0.4140 (+24.3%)	0.4290 (+28.8%)	0.4350 (+30.6%)
30 docs	0.2967	0.3740 (+26.1%)	0.4013 (+35.3%)	0.3987 (+34.4%)	0.4060 (+36.8%)

Table 20. Precision on the representative testbed when 10 sources were selected. (CORI is the baseline.)

Precision at Doc Rank	CORI	ReDDE	UMM/HR	UMM/HP-FL	UMM/HP-VL
5 docs	0.3920	0.3720 (-5.1%)	0.4480 (+14.3%)	0.4360 (+11.2%)	0.4520 (+15.3%)
10 docs	0.3860	0.3840 (-0.5%)	0.4520 (+17.1%)	0.4440 (+15.0%)	0.4560 (+18.1%)
15 docs	0.3813	0.3680 (-3.5%)	0.4373 (+14.7%)	0.4387 (+15.1%)	0.4453 (+16.8%)
20 docs	0.3710	0.3710 (+0.0%)	0.4250 (+14.5%)	0.4300 (+15.8%)	0.4370 (+17.7%)
30 docs	0.3513	0.3640 (+3.6%)	0.4140 (+17.9%)	0.4247 (+20.9%)	0.4227 (+20.3%)

sources (10). This suggests that the power of these two algorithms comes from the explicit optimization of the high-precision goal of the federated document retrieval application.

On the representative testbed, CORI algorithm was not as effective as the other algorithms especially when a small number of information sources (3 and 5) were selected. This can be explained by the fact that CORI does not consider information source size factor and the two large information sources with more relevant documents were not ranked highly in its database ranking. The document retrieval results with ReDDE algorithm were more accurate (when 3 or 5 information sources were selected) or at least as good as (when 10 information sources were selected) than those with the CORI algorithm, but consistently worse than the results with the UUM/HR algorithm. It can be noticed from Tables 18-20 that all the three variants of the UUM algorithms (UUM/HR, UUM/HP-FL and UUM/HP-VL) produced document retrieval results of roughly the same qualities. Carefully analysis shows that the overlap of the selected information sources among these three algorithms was much larger on the representative testbed than on the Trec123-100Col testbed as all of them tended to select the two large information sources with a lot of relevant documents. Therefore, the document retrieval results produced by the three UUM algorithms were roughly the same accurate.

To summarize, the strategy of explicitly optimizing the high-precision goal is very important to obtain accurate results of the federated document retrieval application. The algorithms designed according to this criterion have been shown to be very effective in different environments.

6.4 Future research

This chapter proposes the unified utility maximization framework to integrate individual solutions of different sub-problems of federated search together. More thorough experiments should be conducted to study the effectiveness of this integration such as how accurate the resource descriptions and the results merging should be in order to obtain effective results for the information source recommendation application or for the federated document retrieval application. Another research direction is to extend the framework to better simulate applications in real operational environments. In one scenario, the users are assumed to spend more efforts to browse the top retrieved documents and thus more weights are assigned on the top ranked documents. In another use scenario, the retrieval effectiveness of individual information sources is considered and incorporated into the unified utility maximization. The detail of the new research is discussed in Section 7.5.

Chapter 7: Dissertation Research

The previous chapters introduce new research to address main sub-problems of federated search separately and also propose a unified framework to integrate the separate solutions of individual sub-problems together to optimize various goals of different federate search applications. The new research provides more solid theoretical basis for federated search task and is also more promising to be utilized in the real operational environments. To more thoroughly study the effectiveness of the proposed algorithms and to better model our federated search solutions for real operational applications, a large body of dissertation research is proposed in this chapter.

7.1 Experiments with more data

Chapter 2 introduces several approaches to evaluate the effectiveness of federated search systems. The most desired evaluation approach is to study the effectiveness of different algorithms with real world applications. FedStats is an on-going project in Carnegie Mellon University. This project develops a federated Web search portal to connect federal agency information sources and provide federated search solution. FedStats is a real world application and it is an ideal candidate to evaluate federated search algorithms. However, some constraints limit the choice with this evaluation approach. For example, it is hard for us to acquire enough relevance judgments to tune the behavior of the federated search systems or to evaluate the effectiveness in full aspects. For example, evaluating the resource selection effectiveness of information source recommendation application described in Chapters 4 and 6 requires knowing the distribution of all relevant documents across the information sources, which is very difficult in the FedStats project. Although it is hard to conduct complete and thorough experiments with the FedStats project, user studies will be arranged to report possible evaluations with this real federated search application.

On the other side, TREC text collections provide a good opportunity for us to simulate the federated search environments. TREC data is composed of large corpora with sufficient queries and relevance judgments (Harman, 1995), which enables us to better tune our system and thoroughly evaluate the effectiveness. TREC news/government data has been utilized for evaluation throughout this dissertation to simulate the federated search environments of large organizations (companies) and topic-oriented hidden Web contents. As it is discussed in Chapter 2, the TREC news/government data has some weakness to simulate the environments of open domain hidden Web where the TREC Web data may be a better alternative. In the dissertation research, more detailed analysis will be made for the TREC Web collections WT10g. More specifically, a partition method will be proposed to better simulate the information source size distribution and relevant document distribution of open domain hidden Web environments. Experiments will be conducted and reported on the new testbed based on the TREC Web data.

7.2 Information source size estimation

The experiments in Chapter 4 and Chapter 6 indicate the importance of considering information source size factors in federated search task. To obtain the information source size estimates, the Sample-Resample algorithm is proposed and has been shown to be more effective than the Capture-Recapture algorithm in some cases. However, more experiments are necessary to study the behavior of the Sample-Resample algorithm in more diverse environments such as for larger information sources or to study the robustness of this algorithm by choosing various number of resample queries. Furthermore, the basic Sample-Resample algorithm depends on document frequency information provided by individual search engines, which is a serious problem in operational environments where the information is not available. A new variant of the Sample-Resample method is proposed to release the constraint in this section.

7.2.1 More experiments to study Sample-Resample algorithm

A set of experiments are conducted in Chapter 3 to study the effectiveness of the Sample-Resample algorithm. These experiments utilize a single experimental configuration to choose a fixed number of 5 resample queries and randomly choose the resample query terms. One direction for new research is to vary the number of resample queries. It can be imagined that more number of resample queries may reduce the variance of the estimation by the Sample-Resample algorithm. New experiments will be scheduled to study the effects of using more resample queries. However, an interesting question would be: whether more number of resample queries can substantially improve the source size estimation accuracy. As the Sample-Resample algorithm has been shown to underestimate the source size as indicated in Chapter 2, it is not clear whether more number of resample queries can improve the estimation accuracy as the reduced variance may result in a more stable underestimation. This question will be answered with empirical studies. Another direction for new research of the Sample-Resample algorithm is to study the selection method of the resample queries. Choosing resample query terms with low document frequencies in the sampled documents may suffer high variance of the document frequency statistics, while choosing terms with high document frequencies in the sampled documents may show bias favor the common words in the sampled documents and thus suffer size underestimations. A set of experiments will be conducted to compare the effects of choosing resample query terms with various types of document frequencies in the sampled documents.

Another limitation of the experiments in Chapter 3 is that only testbeds with small information sources (about 10,000 documents by average) and testbeds with relatively large information sources (about 100,000 documents by average) are chosen for evaluation. However, in real operational environments, even larger information sources may exist. Information sources with 300,000 documents have been used in the evaluation of prior research (Liu et al., 2001). Therefore, new experiments with large size information sources (e.g., 500,000 or larger) will be conducted in the dissertation research.

7.2.2 Sample-Resample database size estimation algorithm without available document frequency information

The basic Sample-Resample information source size estimation algorithm described in Chapter 3 can be easily combined with the query-based sampling method and has been shown to be quite effective when the document frequency information is provided by the information sources. However, in real operational environments (especially in large uncooperative environments like open domain hidden Web), it is very likely that some information sources do not provide the document frequency information. How to extend the basic Sample-Resample algorithm to work for these information sources is an interesting research problem.

In the basic Sample-Resample algorithm, the information source size estimates are calculated by the following formula as:

$$\widehat{N} = \frac{df_q * N_{samp}}{df_{q\_samp}} \tag{7.1}$$

where N_{samp} denotes the number of sampled documents from a particular information source, and df_{q_samp} stands for the number of sampled documents from this information source that contain the query term q (sampled document frequency). df_q is the number of documents in the whole information source that contain this term (document frequency).

Therefore, the key problem is to estimate the number of documents in the whole information source that contain a specific query term without requiring this statistic from the information source. The solution could be rather simple. Our approach is to send the single-term query to the information source and then analyze the document ids in the returned ranked lists. The overlap between the returned list and centralized sample database can be explored and the corresponding document frequency in the whole information source can be estimated.

Specifically, the single-term query is sent to the information source and a ranked list of N_{rtop} document ids is acquired. Let df_{q_samp} denote the number of sampled documents from this information source that contain the query term q , where df_{qtop_samp} of them can also be found among the returned ranked list from the complete information source. As long as we assume the sampled documents are representative, the document frequency of this term can be derived in the whole information source as follows:

$$df_q = \frac{N_{rtop} * df_{q\_samp}}{df_{qtop\_samp}} \tag{7.2}$$

Combining this formula together with Equation 7.1, the algorithm is able to estimate the corresponding information source size without requiring the document frequency from the information source.

Experiments will be designed to study the effectiveness of this new Sample-Resample algorithm with information sources of different characteristics (relatively small or very large). Especially, the accuracy of the new Sample-Resample algorithm and the basic Sample-Resample algorithm with additional information of actual document frequency will be compared to study how well the new Sample-Resample can work with limited evidence.

7.3 Resource selection for information source recommendation application

Information source recommendation system suggests most relevant information sources for given text queries. This system is very useful when the users want to browse the selected information sources by themselves instead of asking the system to retrieve relevant documents automatically.

Prior resource selection algorithms of information source recommendation system suggest all the documents in the selected information sources and are measured by the total number of relevant documents among the selected information sources. However, this may not be the case for real operational environments, where it can be imagined that most users only browse reasonable amount of documents (e.g., 100 or lower) retrieved from a particular selected information source when they visit this information source. This observation is consistent with the “high-precision” case of the rank-based ad-hoc information retrieval (Zhai & Lafferty, 2003) which assumes that users browse the retrieved documents in order and may stop wherever is appropriate. Therefore, an alternative use scenario of information source recommendation application is to select a small number of information sources that can return the largest

number of relevant documents in top retrieved documents of the selected information sources. Note that the retrieval effectiveness of each information source must be considered for this use scenario. Otherwise an information source with a very ineffective search engine that ranks all the documents from the most irrelevant ones to the most relevant ones is considered as valuable as another information source whose search engine is effective and ranks the documents in a good manner. In this use scenario, the evaluation measure of the information source recommendation system can be modified to compare the number of relevant documents in the top part of retrieved documents from the selected information sources.

One candidate of the resource selection algorithm for this new use scenario is the UUM/HP algorithm proposed in Chapter 6. However, as discussed above, the retrieval effectiveness issue must be addressed. More discussion about how to incorporate the factor of retrieval effectiveness is shown in Section 7.5. In a word, a new evaluation method for information source recommendation application is proposed to better simulate user behaviors in real operational environments and empirical studies will be conducted to evaluate the effectiveness of different algorithms for this new use scenario.

7.4 Result merging

Chapter 5 proposes the Semi-Supervised Learning (SSL) algorithm. The SSL algorithm builds separate linear models for selected information sources to transform information source specific scores into comparable information source independent scores. However, in real operational environments, information sources may not return document scores with the ranked document lists. A simple solution of dealing with the ranked lists without document scores is to normalize the ranks for pseudo document scores. It is suspected that this type of pseudo document scores calculated with only rank information should be inferior to actual document scores and may cause decrease of the merging accuracy. Experiments will be designed to study the effectiveness of different results merging algorithms for ranked lists without document scores and study the accuracy difference between the merging results of ranked lists without document scores and those of ranked lists with document scores.

Another set of experiments scheduled for the dissertation research is to study the behavior of minimum downloading variant of the Semi-Supervised Learning algorithm when only very short ranked lists are available. In Chapter 5, the minimum downloading method of Semi-Supervised Learning algorithm is evaluated with ranked lists that contain relatively small number of document ids such as 50 or 100 ids. However, in real operational environments, users may be interested in browsing the very top documents and thus only retrieve 10 or 20 documents from each selected information source. Furthermore, this is also the case when the UUM/HP-FL resource selection algorithm is utilized and the number of documents to retrieve from each selected database may be 10 or 20. It can be expected that most information sources are short of training data with this type of very short ranked lists and experiments will be conducted to study the behavior of different results merging algorithm in this case.

7.5 Unified utility maximization framework

Unified utility maximization framework proposed in Chapter 6 is a key contribution of this disseration. It integrates the individual solutions of different sub-problems of federated research together. Most prior research of different sub-problems of federated research is focused on improving the effectiveness of individual sub-problems. However, the effectiveness of individual solutions of the sub-problems may or may not be crucial for the overal system effectivesness. For example, it is not clear yet how accurate the resource descriptions and the results merging should be in order to obtain effective results for information

source recommendation system or for federated document retrieval system. Studying the influence of individual sub-problem solutions within the unified utility maximization framework provides us a better way to evaluate the individual solutions and thus to identify the bottleneck for further research.

Other than refine the individual solutions within the unified utility maximization framework, another research direction is to remodel the framework for simulating more applications in the real operational environments. The unified utility maximization framekwork is very flexible and many types of algorithms can be designed to achieve effective results for different applications. This section proposes two types of algorithms within the unified utility maximization framework to address two issues that may be important in the real operational environments. More speificially, the first algorithm assumes users spend more efforts to browse the top retrieved documents by putting more weights on the top ranked documents; and the second algorithm tries to model the retrieval effectiveness of the individual information sources and incorporates their retrieval effectiveness into the resource selection decision.

7.5.1 Study the resource representation accuracy and the results merging accuracy within the unified utility maximization framework

Two types of resource descriptions are utilized in the unified utility maximization framework, namely the content representations and the information source size estimates. In uncooperative environments, the content representations are often acquired by query-based sampling. Prior research (Callan, 2000) has shown more sampled documents from a specific information source can improve the accuracy of content representation and can be used by a big document resource selection algorithm to improve the results of the information source recommendation system and the federated document retrieval system. The UUM/HR and UUM/HP algorithms have been analyzed and shown to be superior to the big document resource selection approach in Chapter 6. It can be imagined that more sampled documents may result in better approximation of the centralized document scores and thus better approximation of the probabilities of relevance. Empirical studies will be conducted to study the influence of the amount of sampled documents on the overall results for the UUM/HR and UUM/HP algorithms.

Information source size estimate is another type of resource descriptions utilized by the new research in this dissertation. Chapter2 shows that current source size estimation algorithms such as Sample-Resample and Capture-Recapture give relatively rough estimation especially when only a small amount of communication costs are allowed. To investigate the influence of this rough source size estimation, experiments will be conducted to compare the results of information source recommendation system and the results of federated document retrieval system with actual information source sizes and the estimated information source sizes.

As well as resource descriptions, results merging step also introduces errors into the overall results of federated document retrieval system. SSL algorithm proposed in Chapter 5 has been shown to be more effective than prior research. However, it suffers from several problems such as the linear transformation assumption and the scarcity of training data. Empirical study will be scheduled to learn the difference in the accuracy of federated document retrieval results with the SSL algorithm and with an “optimal” merging algorithm that downloads and calculates the centralized scores for retrieved documents. This shows the upper bound of the results that the SSL merging algorithm can achieve within the unified utility maximization framework.

7.5.2 Associate different weights with top ranked documents in unified utility maximization framework

In federated document retrieval, the accuracy is usually measured by the high-precision at the top retrieved documents. Specifically, the Precision at top 5, 10, 15, 20, 30, 100, 500 and 1000 documents is evaluated. One example is shown as follows:

Precision:

At 5 docs:	0.3640
At 10 docs:	0.3360
At 15 docs:	0.3253
At 20 docs:	0.3140
At 30 docs:	0.2780
At 100 docs:	0.1666
At 200 docs:	0.0833
At 500 docs:	0.0333
At 1000 docs:	0.0167

It can be noted that these 1,000 documents have different weights in the evaluation. The top 1 document contributes to all the Precision at different ranks while the last document only influences the precision at the top 1,000 documents. Actually, the utility functions of the UUM/HP-FL algorithm and the UUM/HP-VL algorithm put equal weights on all the N_{rdoc} retrieved documents. On the other side, it is possible to put more emphasis on the top-ranked documents like:

$10*R(d_{ij})$ (For the d_{ij} at top 5) + $5*R(d_{ij})$ (For the d_{ij} at top 10) + $2*R(d_{ij})$ (For the d_{ij} at top 30)...

This approach of putting more emphasis on top-ranked documents is similar with the "high-precision" case of the rank-based ad-hoc information retrieval (Zhai & Lafferty, 2003) which assumes that users read the retrieved documents in order and may stop wherever is appropriate. That research uses a "stop distribution" to model the user reading behavior and puts more emphasis on the top ranked documents with the "stop distribution".

New research will be conducted to study how to select the best criterion to reflect the characteristics of this application, which should be easy to optimize. One concern about this type of new optimization goal is that it requires more accurate estimated probabilities of relevance of the documents. It can be imagined that a little variation on the probabilities of relevance of the top-ranked (1-5) documents will cause a large difference in the utility value of the new goal as it puts a large amount of emphasis on these documents. On the other side, the original uniform weight goal of the UUM/HP-FL algorithm and the UUM/HP-VL algorithm alleviates the influence by smoothing these weights among all the retrieved documents. Effects of this problem will also be studied.

7.5.3 Incorporate the impact of information source retrieval effectiveness into the unified utility maximization framework

Federated search is the task to search and retrieve useful information among individual information sources. Therefore, it is important to consider the effectiveness of individual search engines in the information sources. Very limited previous research has been conducted in this direction. This Section first analyzes the importance of incorporating the factors of individual information source retrieval effectiveness, then defines the new research problem and briefly discusses the expected solution and the evaluation methodology.

In a federated search system, the documents are maintained and searched by individual information sources. Therefore, the retrieval effectiveness of the information sources should be an important factor to be considered during the design of the algorithms. Most current resource selection algorithms (both for information source recommendation application and federated document retrieval application) only select information sources with the largest amount of relevant documents and do not consider whether these relevant documents can be retrieved or not due to the search engine effectiveness.

For an information source recommendation system, the users are provided with the recommendation of several relevant information sources and will visit these information sources themselves to find out the relevant documents. An information source which contains many relevant documents is a good candidate

for the resource selection algorithm. However, if the search engine of this information source is of very low quality, it may retrieve very few relevant documents when the users visit the information source and the utility of selecting this information source may be even lower than that of selecting an information source which contains fewer relevant documents but is much more effective to retrieve all of them.

For a federated document retrieval system, the Semi-Supervised Learning results merging algorithm helps correct the ineffectiveness problem of the search engines. When the returned documents from a selected information sources are less relevant than they were expected in the resource selection phrase, the SSL algorithm detects this inconsistency and automatically generates a linear transformation model to recover roughly the true centralized document scores (thus the probabilities of relevance) for those documents so that these "bad" documents do not outrank other "good" documents from other selected information sources. However, the key point is that the failure to consider the effectiveness of individual search engines in the resource selection phrase has already damaged the overall accuracy. If better method has been utilized in the resource selection phrase, we may choose information sources that are able to contribute more relevant documents, and avoid simply choosing the information sources with a lot of relevant documents that may not be retrieved at all. Therefore, a more sophisticated resource selection algorithm incorporated with the effectiveness of individual search engines is also helpful for the federated document retrieval application.

The basic idea of evaluating the retrieval effectiveness of individual information sources is to compare their retrieval results with the centralized retrieval algorithm (such as INQUERY). Hence, a similar assumption can be made as the one made for results merging that the more similar retrieval results an information source produces with the results of the centralized retrieval algorithm the more effective is the search engine of this information source.

The model to measure the effectiveness of a particular search engine can be built by collecting information source specific retrieval results and the corresponding centralized retrieval results for a set of training queries. There is no requirement for human relevance judgment. For each training query, the information source specific scores can be extracted from the returned rank list, and the retrieved documents are downloaded to calculate the corresponding centralized scores with the centralized retrieval algorithm.

In order to implement this idea within the unified utility maximization model, the model can be formalized as follows:

$$\hat{S_i}(d_{ij}) = a_i \hat{S_C}(d_{ij}) + b_i + \text{Noise\_Model}(i) \qquad (7.3)$$

$\hat{S_C}(d_{ij})$ denotes the estimated centralized document score for a document from the particular i^{th} information source, and $\hat{S_i}(d_{ij})$ denotes the corresponding estimated source specific score. a_i and b_i are the two parameters of the linear model and Noise_Model(i) is used to measure the effectiveness of the search engine.

It can be noted that here we are conducting the reverse transformation as what has been done for results merging. Specifically, the information source specific scores are estimated from the corresponding centralized document scores. It can be imagined that if the i^{th} information source uses the same kind of retrieval algorithm and also the same set of corpus statistics as the centralized retrieval algorithm, equal information source specific scores and the centralized scores will be obtained as $\hat{S_i}(d_{ij}) = \hat{S_C}(d_{ij})$. Then the Noise_Model is zero in this case. In contrast, the more inconsistent results that the particular search engine produces with the centralized retrieval algorithm, the larger is the Noise_Model. Therefore, the Noise _Model exactly measures the effectiveness of the particular search engine. We expect that the Normal distribution (or Gaussian distribution) can be used to build this Noise _Model.

For the resource selection algorithm of both the information source recommendation application and the federated document retrieval application, the selection decisions are made based on the probabilities of relevance of the documents among the information sources. It is assumed in Chapter 6 that documents with

the largest probabilities of relevance can always be retrieved. However, this is not correct when the search engines of the information sources are not effective for retrieval. Our strategy is to simulate the individual information source retrieval results by the transformation indicated by Equation 7.3. For example, in the case for a particular information source which is not very effective (with a large Noise_Model), two documents from this information source have been estimated to have centralized document scores as 0.8 and 0.7 respectively (this means the first document is more likely to be relevant than the second one). With the corresponding transformation model described in Equation 7.3, the first document may be assigned a information source specific score of 0.6 and the second may be assigned an information source specific score of 0.7 and therefore the document which is more likely to be relevant will be ranked after the other one.

To summarize, the retrieval behavior of the search engine in a particular information source can be obtained by simulating information source specific scores for its documents via the transformation model. The probabilities of relevance of those documents ranked at the top part of this simulated ranked lists can be used to make the selection decision. Therefore, the impact of information source retrieval effectiveness can be well incorporated into the formal framework of the unified utility framework.

Experiments will be conducted to study the impact of information source retrieval effectiveness for both the information source recommendation application and the federated document retrieval application. We plan to study the federated search environments with both effective and ineffective search engines. In the previous experiments, three effective search engines (INQUERY, Language Model and the Vector Space model) were utilized. It is planned to include ineffective search engines such as the Vector Space model with less effective weighting scheme in the further experiments. Multiple types of testbeds with different characteristics will be used. Thorough empirical studies for both the information source recommendation application and the federated document retrieval application will be conducted in different scenarios to study the effectiveness of the new algorithm and the old algorithms.

7.6 Expected contribution

The research discussed in this proposal and the scheduled work in the dissertation research is a significant improvement of federated search. Empirically effective solutions have been proposed to the full range of federated search problems: i) the Sample-Resample information source size estimation algorithm is more efficient than the Capture-Recapture algorithm to acquire accurate source size estimations; ii) the ReDDE resource selection algorithm turns away from the “big document” resource selection approach and explicitly estimates the distribution of relevant documents to produce better information source ranking; and iii) the Semi-Supervised Learning results merging algorithm explicitly approximates centralized document scores and generates more accurate merged result lists than heuristic algorithms such as the CORI results merging formula.

Furthermore, the unified utility maximization framework is proposed to combine the separate solutions together to construct effective systems of different federated search applications. As far as we know, this is the first probabilistic framework for integrating the different components of a federated search system. This more unified view of federated search task provides a new opportunity to utilize available information. The sampled documents have been combined as a centralized sample database to help produce more accurate resource selection and results merging results. By combining the components together, the unified framework also enables us to configure individual components globally to get desired overall results of different applications, which is superior to the simple choice of combining individual effective solutions together as what previous research adopted.

The new research goes beyond previous work in another respect of better modeling the real world applications such as: the Sample-Resample algorithm and its variant effectively acquire source size estimates in broader environments of the Capture-Recapture algorithm. The unified utility maximization

framework is extended to incorporate the retrieval effectiveness of individual information sources, which is important when some information sources provide bad retrieval results in real operational environments.

In summary, the research in this proposal and the scheduled work in the dissertation research is an important step forward of federated search. Federated search has become a hot research area for at least one decade. However, relatively less improvement has been made in the last four or five years. For example, the CORI resource selection algorithm has been the state-of-the-art for a long time. The new work advances the research of federated search in many aspects. It raises the bar to a new less ad-hoc level. It produces more effective results of different federated search applications. It addresses important problems of federated search research and especially considers how to better model this problems in the real world applications. The more theoretical foundation, the better empirically results and the better modeling of real world applications make this new research a bridge to turn the federated search from a cool research topic to a much more practical tool.

Chapter 8: Schedule

July. 2004 – Aug. 2004

Analysis and develop federated search testbed based on TREC Web data.

Sep. 2004 – Dec. 2004

Further study: i) more experiments to study the behavior of Sample-Resample information source size estimation algorithm; and ii) the new variant of Sample-Resample source size estimation without the access of actual document frequency information

Jan. 2005 – Apr. 2005

Further study: i) result merging algorithms with ranked lists without document scores; and ii) the influence of resource representation and results merging on the overall results of federated search applications; and iii) Propose and study new algorithm for unified utility maximization model when different weights are associated with top ranked documents.

May. 2005 – Aug. 2005

Further study: i) Propose and study new algorithm for unified utility maximization model when different weights are associated with top ranked documents; and ii) Propose new algorithms to incorporate information source retrieval effectiveness into the unified utility maximization framework and conduct empirical studies.

Sep. 2005 – Dec. 2005

Analyze the experiment results of the new research; summarize and write up the thesis.

References:

Aslam, J.A. & Montague, M. (2001). Models for metasearch. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

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

Bergman, M. (2000). The Deep Web: Surfacing the Hidden Value. http://www.completeplanet.com/Tutorials/DeepWeb/index.asp. BrightPlanet.

Buckley, C., Singhal, A., Mitra, M. & Salton, G. (1995). New retrieval approaches using SMART. In *Proceedings of 1995 Text REtrieval Conference (TREC-3).* National Institute of Standards and Technology, special publication.

Bergholz, A. & Chidlovskii, B. (2004). Using Query Probing to Identify Query Language Features on the Web. In *Distributed Multimedia Information Retrieval, LNCS 2924, Springer.*

Callan, J., Croft, W.B. & Harding, S. M. (1992). The INQUERY retrieval system. In *Proceedings of the Third International Conference on Database and Expert Systems Applications*, (pp. 78-83).

Callan, J., Croft, W. B. & Broglio, J. (1995a). TREC and TIPSTER experiments with INQUERY. *Information Processing and Management*, 31(3). (pp. 327-343).

Callan, J. Lu, Z. & Croft, W. B. (1995b). Searching distributed collection with inference networks. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.*

Callan, J., Connell, M. & Du, A. (1999). Automatic discovery of language models for text databases. In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*. ACM.

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

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

Conrad, J. G, Guo, X. S, Jackson, P. & Meziou, M. (2002). Database selection using actual physical and acquired logical collection resources in a massive domain-specific operational environment. Distributed search over the hidden web: Hierarchical database sampling and selection. In *Proceedings of the 28th International Conference on Very Large Databases (VLDB).*

Craswell, N., Hawking, D., & Thistlewaite, P. (1999). Merging Results from Isolated Search Engines. *In Proceedings of the 10th Australasian Database Conference.* (pp. 189-200).

Craswell, N. (2000). Methods for distributed information retrieval. Ph. D. thesis, The Australian Nation University.

Craswell, N., Bailey, P. & Hawking, D. (2000). Server selection on the World Wide Web. In *Proceedings of the 5th ACM Conference on Digital Libraries*. (pp. 37-46).

Duda, R., Hart, P. & Stork, D. (2000). Pattern Classification. Wiley-Interscience.

Fox, E. A., Koushik M. P., Shaw J., Modlin, R. & Rao, D. (1992) *Combining evidence from multiple searches*. In D.K. Harman, editor, The First Text REtrieval Conference (TREC-1).

French, J. C., Powell, A. L., Callan. (1998). Evaluating Database Selection Techniques: A Testbed and Experiment Comparing the performance of database selection algorithms. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

French, J. C., Powell, A. L., Callan, J., Viles, C. L., Emmitt, T., Prey, K. J., & 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*.

Fuhr, N. (1996). Optimum Database Selection in Networked IR. In Proceedings of the SIGIR'96 Workshop on Networked Information Retrieval.

Fuhr, N. (1999). A decision-theoretic approach to database selection in networked IR. *ACM Transactions on Information Systems,* 17(3) (pp. 229-249).

Gravano, L., Garcia-Molina, H., & Tomasic. A. (1994). The effectiveness of GlOSS for the text database discovery problem. In *Proceedings of the ACM-SIGMOD International Conference on Management of Data.*

Gravano, L. & García-Molina, H. (1995). Generalizing GLOSS to vector-space databases and broker hierarchies. In *Proceedings of the 21st International Conference on Very Large Databases (VLDB).*

Gravano, L., Chang, C., García-Molina, H., & Paepcke, A. (1997). STARTS: Stanford Proposal for Internet Meta-Searching. In *Proceedings of the ACM-SIGMOD International Conference on Management of Data.*

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

Harman, D. (1995). Overview of the fourth text retrieval conference (TREC-4). *Proceedings of the Third Text Retrieval Conference (TREC-3).* National Institute of Stadards ad Technology Special Publication.

Hawking., D., & Thistlewaite., P. (1999). Methods for information server selection. *ACM Transactions on Information Systems,* 17(1). (pp. 40-76).

Ipeirotis, P. & 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. & 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.*

Kirsch, S.T. (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.

Lafferty, J. & Zhai, C. G. (2001) Document language models, query models, and risk minimization for information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Larkey, L., Connell, M. & Callan, J. (2000). Collection selection and results merging with topically organized U.S. patents and TREC data. In *Proceedings of 9th ACM International Conference on Information and Knowledge Management (CIKM).*

Lee, J. H. (1997). Analyses of multiple evidence combination. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Le Calv, A. & Savoy, J. (2000). Database merging strategy based on logistic regression. *Information Processing & Management,* 36(3).

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

Lu, J. & Callan, J. (2003). Content-based retrieval in hybrid peer-to-peer networks. In *Proceedings of the Twelfth International Conference on Information and Knowledge Management*.

Lu, Z., Callan, J. & Croft, W. (1996). Measures in collection ranking evaluation. Technical Report, Department of Computer Science, University of Massachusetts.

Manmatha, R., Rath, T. & Feng, F. (2001). Modeling score distributions for combining the outputs of search engines. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Mandelbrot, B. (1988). Fractal Geometry of Nature. W.H. Freeman & Co.

Nottelmann, H. & Fuhr, N. (2003a). From uncertain inference to probability of relevance for advanced IR applications. In *Proceedings of the 25th* European Conference on Information Retrieval Research.

Nottelmann, H. & Fuhr, N. (2003b). Evaluating different method of estimating retrieval quality for resource selection. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.*

OCLC (Online Computer Library Center). (2001). Statistics. Online Computer Library Center, Inc. Retrieved August 15, 2001, from the World Wide Web: http://wcp.oclc.org/stats.html.

Ogilvie, P. & Callan, J. (2001a). The effectiveness of query expansion for distributed information retrieval. In *Proceedings of 10th ACM International Conference on Information and Knowledge Management (CIKM).*

Ogilvie, P. & Callan, J. (2001b). Experiments using the Lemur toolkit. In *Proceedings of 2001 Text REtrieval Conference (TREC 2001).* National Institute of Standards and Technology, special publication.

Press, W. H., Flannery, B. P., Teukolsky, S. A. & Vetterling, W. T. (1992). Numerical recipies in C: The art of scientific computing. Cambridge University Press.

Roberson, S. E. & Walker , S. (1994). Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval. In *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.*

Savoy, J. & Rasolofo, Y. (2000). Report on the TREC-9 experiment: Link-based retrieval and distributed collections. In *Proceedings of 9 th Text REtrieval Conference (TREC-9).* National Institute of Standards and Technology, special publication.

Sherman, C. (2001). Search for the invisible web. Guardian Unlimited.

Si, L. & Callan, J. (2002a). Using sampled data and regression to merge search engine results. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.*

Si, L., Jin, R., Callan, J. & Ogilvie, P. (2002b). A language model framework for resource selection and results merging. In *Proceedings of the 11th International Conference on Information and Knowledge Management, ACM.*

Si, L. & 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. & Callan, J. (2003b). A Semi-Supervised learning method to merge search engine results. *ACM Transactions on Information Systems*, 21(4). (pp. 457-491).

Si, L. & Callan, J. (2003c). Distributed information retrieval with skewed database size distributions. *In Proceedings of the NSF's National Conference on Digital Government Research (dg.o2003.)*

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

Turtle, H. (1990). Inference networks for document retrieval. Technical Report COINS Report 90-7, Computer and Information Science Department, University of Massachusetts, Amherst, MA 01003.

Xu, J. & Croft, W. (1996). Query expansion using local and global document analysis. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.*

Xu, J. & Callan, J. (1998). Effective retrieval with distributed collections. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Xu, J. & Croft., W. (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*.

Yuwono, B. & Lee, D. L. (1997). Server ranking for distributed text retrieval systems on the Internet. In R. 25POR and K. TANAKA Eds, DASFAA '97. (pp. 41-49)

Viles, C. L. & French, J. C. (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. K., & 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*.

Yang Y. (1999). An Evaluation of Statistical Approaches to Text Categorization. *Information Retrieval*. (1) (pp 69-90).

Zhai, C. X. & Lafferty, J. (2003). A risk minimization framework for information retrieval, In *Proceedings of the ACM SIGIR 2003 Workshop on Mathematical/Formal Methods in IR*.

Zipf, G. K. (1949). Human Behavior and the Principle of Least Effort: An Introduction to Human Ecology. Addison-Wesley, *Reading,* MA.