Distributed IR for Digital Libraries

Ray R. Larson

School of Information Management and Systems
University of California, Berkeley
Berkeley, California, USA, 94720-4600
`ray@sherlock.berkeley.edu`

Abstract. This paper examines technology developed to support large-scale distributed digital libraries. We describe the method used for harvesting collection information using standard information retrieval protocols and how this information is used in collection ranking and retrieval. The system that we have developed takes a probabilistic approach to distributed information retrieval using a Logistic regression algorithm for estimation of distributed collection relevance and fusion techniques to combine multiple sources of evidence. We discuss the harvesting method used and how it can be employed in building collection representatives using features of the Z39.50 protocol. The extracted collection representatives are ranked using a fusion of probabilistic retrieval methods. The effectiveness of our algorithm is compared to other distributed search methods using test collections developed for distributed search evaluation. We also describe how this system in currently being applied to operational systems in the U.K.

1 Introduction

Users of the World Wide Web (WWW) have become familiar with and, in most cases, dependent on the ability to conduct simple searches that rely on information in databases built from billions of web pages harvested from millions of HTTP servers around the world. But this "visible" web harvested by services such as Google and Inktomi is, for many of these servers, only a small fraction of the total information on a particular web site. Behind the myriad "search pages" on many web sites are the underlying databases that support queries on those pages and the software that constructs pages on demand from their content.

This huge set of databases make up the content of today's digital libraries and has been called collectively the "Deep Web". Estimates of the size of the Deep Web place it at over 7500 Terabytes of information [12]. As increasing numbers of digital libraries around the world make their databases available through protocols such as OAI or Z39.50 the problem arises of determining, for any given query, which of these databases are likely to contain information of interest to a world-wide population of potential users. Certainly one goal must be to aid information seekers in identifying the digital libraries that are pertinent to their needs regardless of whether the desired resources are part of the visible web or the deep web.

However, currently information seekers must rely on the search engines of the visible web to bring them to the search portals of these "Deep Web' databases, where they then must submit a new search that will (it is hoped) obtain results containing information that will satisfy their original need or desire. Today's searcher, therefore, must learn how to search and navigate not only the visible web search engines, but also the differing and often contradictory search mechanisms of the underlying Deep Web databases once those have been identified. The first challenge in exploiting the Deep Web is to decide which of these myriad databases is likely to contain the information that will meet the searcher's needs. Only then can come the challenge of how to mix, match, and combine one or more search engines for diverse digital libraries for any given inquiry, and also how to navigate through the complexities of largely incompatible protocols, metadata, and content structure and representation.

Buckland and Plaunt[1] have pointed out, searching for recorded knowledge in a distributed digital library environment involves three types of selection:

1. Selecting which library (repository) to look in;
2. Selecting which document(s) within a library to look at; and
3. Selecting fragments of data (text, numeric data, images) from within a document.

The research reported in this paper focuses on two aspects of the first type of selection, that is, creating a representative of the distributed databases and then, at search time, using these representatives to discover which of distributed databases are likely to contain information that the searcher desires. This problem, distributed information retrieval, has been an area of active research interest for many years. Distributed IR presents three central research problems that echo the selection problems noted by Buckland and Plaunt. These are:

1. How to select appropriate databases or collections for search from a large number of distributed databases;
2. How to perform parallel or sequential distributed search over the selected databases, possibly using different query structures or search formulations, in a networked environment where not all resources are always available; and
3. How to merge results from the different search engines and collections, with differing record contents and structures (sometimes referred to as the collection fusion problem).

Each of these research problems presents a number of challenges that must be addressed to provide effective and efficient solutions to the overall problem of distributed information retrieval. However, as noted above, the first challenge in exploiting the digital library databases of Deep Web is to identify, for any given query, those databases that are likely to contain information of interest to the user. This has been the central problem of distributed Information Retrieval (IR) and has been a focus of active research interest.

Some of the best known work in this area has been that of Gravano, et al. [6] on GlOSS and Callan's [2] application of inference networks to distributed IR

(CORI). French and Powell, along with a number of collaborators [5, 4, 11], have enabled comparative evaluation of distributed IR by defining test collections derived from TREC data, where the TREC databases are divided into sub-collections representing virtual distributed collections. In addition they defined a number of measures for evaluation of the performance of distributed IR[5]. We use one of these test collections and these metrics in this paper to compare previously published results on collection selection with our probabilistic model based on logistic regression.

In the remainder of paper we first describe the methods that we have been developing to harvest content descriptions, or "collection representatives" from distributed collections and examine its efficiency using the TREC-derived testbed databases. Then we examine the probabilistic algorithm used for searching and ranking these collection representatives and compare its performance with earlier approaches to distributed information retrieval. Finally we describe some actual applications of our approach.

2 Building A Distributed Collection Database

Research in distributed IR has focussed on the three major aspects of collection selection, distributed search, and merging results. Far more rarely have researchers discussed how the databases to be used in collection selection are to be built. In most of the experimental work in distributed IR the researchers have assumed that the collections would be accessible and that statistics normally derived during indexing of a single database would be available. Obviously this will not be the case in true distributed environment. Some have suggested sampling methods are sufficient for extracting adequate information for representation of distributed collection contents (see, for example [7, 10]. This section is based on a short description of this method that appeared in [8] and discusses how the collection representatives were derived.

Our approach to "harvesting" the collection representatives used for the subsequent retrieval and ranking of those collections exploits some features of the Z39.50 Information Retrieval protocol. However, other harvesting method are possible to create similar collection representations, including the sampling methods cited above.

We use only two functional aspects of the Z39.50 protocol in our harvesting method, the Explain Facility and the Browse Facility. In Z39.50 a "facility" is a logical group of services (or a single service) to perform various functions in the interaction between a client (origin) and a server (target).

2.1 The Explain Facility

The Z39.50 Explain facility was designed to enable clients to obtain information about servers. Servers can provide a large amount of information on themselves including names and descriptions of the databases that they support, the "attribute sets" and elements used (an attribute set specifies the allowable search

fields and semantics for a database), diagnostic or error information, record syntaxes and information on defined subsets of record elements that may be requested from the server (or "elementsets" in Z39.50 parlance). This explain database appears to the client as any other database on the server, and the client uses the Z39.50 Search and Retrieval facilities (with explain-specific attribute sets and record syntaxes) to query and retrieve information from it. These attribute sets, explain search keywords, and record syntaxes terms defined in the standard for the Explain database facilitate interoperability of explain among different server implementations.

2.2 Z39.50 Browse Facility

As the name of this facility implies, it was originally intended to support browsing of the server contents, specifically the items extracted for indexing the databases. The Browse facility consists of a single service called the Scan service. It can be used to scan an ordered list of terms (subject headings, titles, keyword, text terms, or other attribute set elements) drawn from the database. Typical implementations of the Scan service directly access the contents of the indexes on the server and return requested portions of those indexes as an ordered list of terms along with the document frequency for each term. This is exactly the kind of information required for information retrieval purposes, or at least for the type of information needed by most distributed search algorithms.

2.3 Implementation

Our harvesting method uses these two Z39.50 Facilities to derive information from Z39.50 servers (which including library catalogs, full-text search systems, and digital library systems) in order to build representatives of the distributed resources. The sequence of operations followed to build these representatives is:

1. Use the Z39.50 protocol to connect to the target server.
2. Search the Explain Database to obtain the list of (one or more) databases that are available on that server. Explain also will provide information about some of the collection level statistics useful in retrieval (such as the total number of records in each database).
3. We can then determine, for each database, which search attributes are supported using the Explain "AttributeDetails" query.
4. For each of the searchable attributes discovered, we send a sequence of Scan requests to the server and collect the resulting lists of index terms. As the lists are collected they are verified for uniqueness (since a server may allow multiple search attributes to be processed by the same index) so that duplication is avoided.
5. For each database an XML collection document is constructed to act as a surrogate for the database using the information obtained from the server Explain database and the Scans of the various indexes.

6. A database of collection documents is created and indexed using all of the terms and frequency information derived above.

The result of this process is a database of collection representatives that may be searched by any of the index terms that would be searchable via the various search attributes of the Z39.50 protocol, each grouped within the XML collection representatives. This permits, for example, terms that occur in titles of a particular collection to be searched separately. In the evaluation study described below the average time required to apply the above steps and create a collection representative from a remote database over the network was about 24 seconds. This time is quite reasonable considering that the underlying 236 collections used in this evaluation contained the full contents of TREC disks 1, 2, and 3, with individual collection sizes ranging from less than ten to over 8000 full-text documents.

Although the Z39.50 protocol has been used previously to construct collection representative databases[10], in that work random samples of the records in the collection were used to build the indexes. An advantage of the method described here is its ability to exploit the server's work in processing its records while extracting the terms to be matched in the collection representative index.

In the collection representative each harvested index of the source databases can become a separately searchable element in the collection database. This permits us to apply the same index restrictions to distributed searching as are applied in searching the individual databases. For example, if the harvested databases maintain an author index, this can be used to limit the possible matching collections to that do support an author search.

3 Probabilistic Distributed IR

Once the collection representatives have been constructed they must retrieved and ranked to attempt to predict which of the distributed collections are likely to contain the documents that would match the searcher's query. For this retrieval, we use a probabilistic collection ranking algorithm.

The probabilistic retrieval algorithms derived for this work are based on the *logistical regression* algorithms developed by researchers at U.C. Berkeley and tested in TREC evaluations [3]. Since the "collection documents" used for this evaluation represent collections of documents and not individual documents, a number of differences from the usual logistic regression measures were used. In addition, analysis showed that different forms of the TREC queries (short queries using topic titles only, longer queries including the title and concepts fields and the "very long" queries including the title, concepts, description and narrative) behaved quite differently in searching the distributed collection, so three different regression equations were derived and applied automatically based on the length of the query. In a following section we will examine the effectiveness of the algorithms when compared to the CORI collection ranking algorithm for the same queries.

In the logistic regression model of IR, the estimated probability of relevance for a particular query and a particular collection (or collection document) $P(R \mid Q, C)$, is calculated and collections are ranked in order of decreasing values of that probability. In the current system $P(R \mid Q, C)$ is calculated as the "log odds" of relevance $\log O(R \mid Q, C)$, Logistic regression provides estimates for a set of coefficients, c_i, associated with a set of S statistics, X_i, derived from the query and database of collection documents, such that:

$$\log O(R \mid Q, C) \approx c_0 \sum_{i=1}^{S} c_i X_i \qquad (1)$$

where c_0 is the intercept term of the regression. For the set of M terms that occur in both a particular query and a given collection document. The statistics used in this study were:

$X_1 = \frac{1}{M} \sum_{j=1}^{M} logQAF_{t_j}$. This is the log of the absolute frequency of occurrence for term t_j in the query averaged over the M terms in common between the query and the collection document.

$X_2 = \sqrt{QL}$. This is square root of the query length (i.e., the number of terms in the query disregarding stopwords).

$X_3 = \frac{1}{M} \sum_{j=1}^{M} logCAF_{t_j}$. This is is the log of the absolute frequency of occurrence for term t_j in the collection averaged over the M common terms.

$X_4 = \sqrt{\frac{CL}{10}}$. This is square root of the collection size. (We use an estimate oc collection size based on the size of the harvested collection representative).

$X_5 = \frac{1}{M} \sum_{j=1}^{M} logICF_{t_j}$. This is is the log of the *inverse collection frequency*(ICF) averaged over the M common terms. ICF is calculated as the total number of collections divided by the number that contain term t_j

$X_6 = logM$. The log of the number of terms in common between the collection document and the query.

For short (title only) queries the equation used in ranking was:

$$\log O(R \mid Q, C) = -3.70 + (1.269 * X_1) + (-0.310 * X_2)$$
$$+ (0.679 * X_3) + K$$
$$+ (0.223 * X_5) + (4.01 * X_6);$$

(K is a constant because query term frequency is always 1 in short queries) For long (title and concepts) the equation used was:

$$\log O(R \mid Q, C) = -7.0103 + (2.3188 * X_1) + (-1.1257 * X_2)$$
$$+ (1.0695 * X_3) + (-0.00294 * X_4)$$
$$+ (5.9174 * X_5) + (2.3612 * X_6);$$

And for very long queries the equation used was:

$$\log O(R \mid Q, C) = -20.9850 + (9.6801 * X_1) + (-1.8669 * X_2)$$
$$+ (1.1921 * X_3) + (-0.00537 * X_4)$$
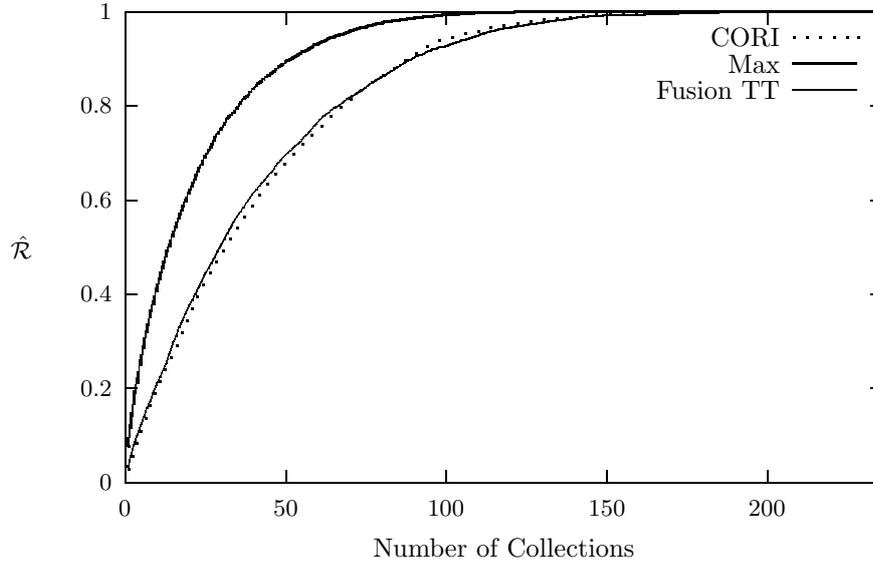$$+ (6.2501 * X_5) + (7.5491 * X_6);$$

Fig. 1. Distributed Search Evaluation - Title Queries

For the evaluation discussed below we used the "Fusion Search" facility in the Cheshire II system to merge the result sets from multiple probabilistic searches. For all of the results reported here separate probabilistic searches were performed on two different elements of the collection representatives (information derived from the titles of the documents in the harvested collections, and terms derived from anywhere in the documents. The ranked results (obtained using the algorithm above) for each search were then merged into a single integrated result set for a given query. The "Fusion Search" facility was developed originally to support combination of results from different searches. We have used this approach in both single document collections, and for distributed collection representative databases with good results. For example, we have exploited this facility in our retrieval processing for the INEX information retrieval of XML evaluation[9].

The "Fusion Search" facility functions by combining the ranking weights of separated searches. When the same documents, or document components, have been retrieved in differing searches, their final ranking value is based on combining the weights from each of the source sets. It should be noted, however, that in the current implementation this final ranking value is not a an estimated probability but a simple summation of probabilistic collection representative weights. The facility also permits combinations of weighted and Boolean values, although only the estimated probabilities for harvested title terms and full-text terms were combined in the following evaluation.

4 Evaluation

For this study we used collections formed by dividing the documents on TIP-STER disks 1, 2, and 3 into 236 sets based on source and month[5]. Collection relevance information was based on based on whether one or more individual documents in the collection were relevant according to the relevance judgements for TREC queries 51-150. The relevance information was used both for estimating the logistic regression coefficients (using a sample of the data) and for the evaluation (with full data).

In French, et al.[4], the authors describe three effectivenes measures for evaluation of distributed information retrieval. The effectiveness measures assume that each database of the distributed collection has (potentially) some *merit* for a given query q. As "ground truth" we assume the that optimal ranking is one in which each database containing any relevant documents is ranked before all databases that do not, and for all of the databases that do have one or more relevant documents, the ranking is in descending order of the number of relevant documents in the database. This forms the upper limit on the performance possible for the ranking algorithm. So, for this optimal ranking B and an estimated or test ranking E, let db_{bi} and db_{ei} denote the database in the i-th ranked position of rankings B and E. Also, let $B_i = merit(q, db_{bi})$ and $E_i = merit(q, db_{ei})$, where $merit(q, db)$ is simply the number of relevant documents in db. Using these, French, et al., define two "Recall Analogs" and a "Precision Analog". The first Recall analog (suggested by Gravano, et al.[6])

$$\mathcal{R}_n = \frac{\sum_{i=1}^{n} E_i}{\sum_{i=1}^{n} B_i} \tag{2}$$

However, as French, et al. point out this measure operates over all databases whether or not they have any relevant documents. They suggest a more appropriate measure might be one where the denominator only takes into account the databases with at least one relevant document:

$$\hat{\mathcal{R}}_n = \frac{\sum_{i=1}^{n} E_i}{\sum_{i=1}^{n^*} B_i} \tag{3}$$

Where $n^* = \max k$, such that $B_k \neq 0$.

The Precision analog is simply the fraction of top n databases in the estimated ranking that have non-zero merit:

$$\mathcal{P}_n = \frac{|db \in \text{Top}_n(E)|merit(q, db) > 0|}{|\text{Top}_n(E)|} \tag{4}$$

Figures 1, 2 and 3 summarize the results of the evaluation. The X axis of the graphs is the number of collections in the ranking and the Y axis, $\hat{\mathcal{R}}$, is the suggested Recall analog measuring the proportion of the total possible relevant documents that have been accumulated in the top N databases, averaged across all of the queries. The Max line in the figures shows the optimal (baseline) results based where the collections are ranked in order of the number of relevance

documents they contain. The figures contain the results of Callan's Inference net approach[2], indicated by CORI (described in [11]). The "Fusion TT" line is the fusion and logistic regression method described above using the title and fulltext elements of the harvested database representatives. The CORI results are the best (to date) results reported for distributed search using the collection used in this evaluation.

For title queries (i.e. those that use only the title portion of the TREC queries), shownFigure 1) the described method achieves higher recall than the CORI algorithm for up to about 100 collections, where CORI exceeds it. Similar performance is shown for long queries where the the fused logistic regression method and CORI perform virtually identically. However, for very long queries shown in Figure 3, the CORI results exceed those of our fusion approach. We would suggest based on these results that CORI would be a better approach for very large queries (such as using an entire document as a query) where the user (or system) is willing to search across many databases. However, for the short queries typical of most user interaction with digital libraries and search systems, the fusion approach with the logistic regression coefficients described above appears to be a better choice.

We are still examining different combinations of elements for the fusion approach, and also examining fusing different probabilistic models. We have tested a version of the well-known Okapi BM-25 (altered for application to collection representatives instead of documents) both alone and fused with the "Fusion TT" results above. The fused results were virtually indistinguishable from the "Fusion TT" results, while the Okapi BM-25 used along performed very poorly.

Preliminary analyses suggest that, since it is impossible to know or estimate from the summary frequency information available in the collection representatives how *many* relevant documents may occur in a given database, there may not be enough information to significantly improve over the results reported here.

5   Applications

Over the past several years we have started to use the Cheshire II system to implement production-level services providing access to full-text SGML and XML document for a number of digital library systems in the United States and the United Kingdom, including the UC Berkeley Digital Library Initiative project sponsored by NSF, NASA and ARPA, The Archives Hub sponsored by JISC in the UK, The History Data Service of AHDS in the UK and the Resource Discovery Network in the UK. The Cheshire system is also beginning to be used to provide scalable distributed retrieval for consortia of institutions providing access to online catalogs and archival collections. At present there are two major distributed systems using the techniques and algorithms described above: the Merseylibraries.org system and the Distributed Archives Hub.

MerseyLibraries.org is sponsored by an organization known as "Libraries Together: Liverpool Learning Partnership" with support from The British Li-
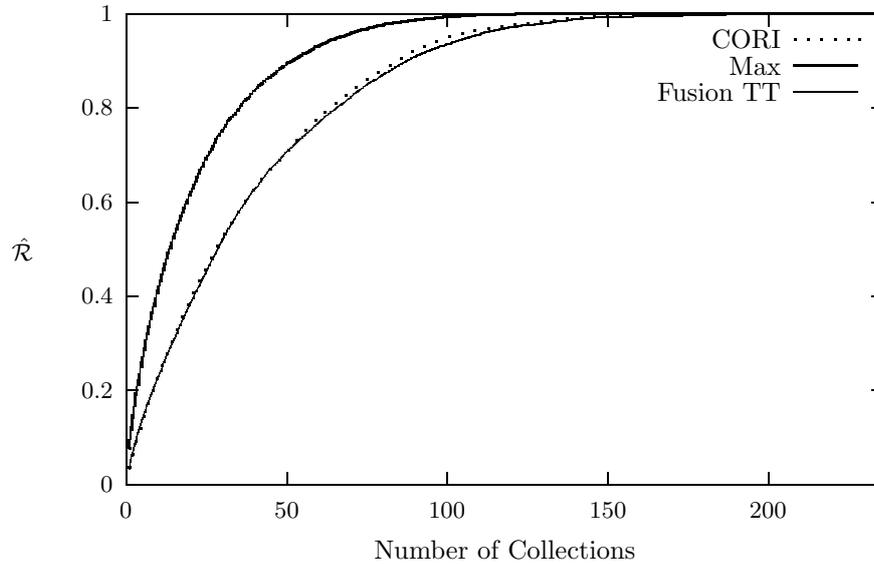
Fig. 2. Distributed Search Evaluation - Long Queries

brary's Co-operation and Partnership Programme (CPP) and the Library and Information Cooperation Council in the UK. Its aims are:

- To provide a web-searchable interface for all the libraries in the Merseyside region.
- To publicize and promote the collections of Merseyside.
- To highlight the benefits of cooperative activity for regional libraries.
- To bring together information about library, archives, and information services.
- To provide a platform for the implementation of cutting edge technologies in libraries and archive repositories.

Currently, the service provides access to the catalogues from of the region's major libraries including: Liverpool City Libraries, Wirral Libraries, Halton Libraries, Sefton Libraries, Warrington Libraries, University of Liverpool Library, Liverpool Hope University College, Liverpool John Moores University, Liverpool Community College, Liverpool Institute for Performing Arts, University of Liverpool Library (archives), and the Liverpool and Merseyside Record Offices. The "virtual union catalog" provided by this service can be searched at http://www.merseylibraries.org. This represents millions of catalog records.

The Distributed Archives Hub is the developing distributed version of the Archives Hub (http://www.archiveshub.ac.uk – which is a centralized Cheshire II based system). The Archives Hub provides a single point of access to descriptions of archives held in repositories throughout the UK. The repositories (libraries,
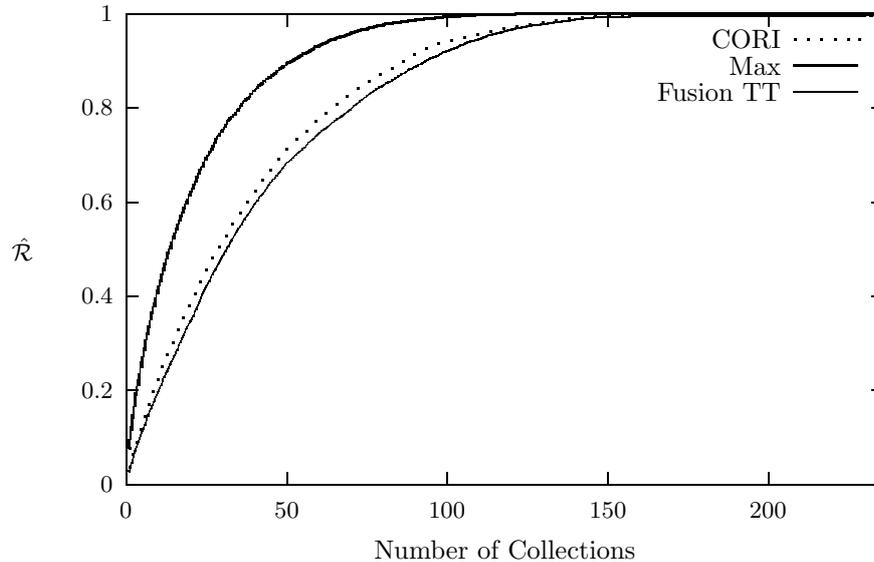
Fig. 3. Distributed Search Evaluation - Very Long Queries

archives, or special collections departments) with archival descriptions available on the Hub currently include over sixty UK institutions, primarily Universities and Colleges, and thousands of archival descriptions. In the distributed version of the Archives Hub each repository (or consortia of repositories) will maintain their own individual servers, which will then amalgamated and made accessible using the techniques discussed in this paper.

6 CONCLUSION

In this paper we have described a standards-based method for building a resource discovery database from distributed Z39.50 servers and examined the algorithms that can be used to select collection representatives likely to contain information relevant to a given query. In addition we have described algorithms that can be used to search these collection representatives and to rank the databases for more effective searching.

We have shown how the presented algorithms and fusion methods appear to perform quite well for the kinds of short queries typically submitted by searchers to digital libraries and search systems.

We have also briefly described the application of the techniques described in two large-scale distributed systems in the UK. We believe that these distributed search techniques can be applied in a wide variety of Digital Library context and can provide an effective and scalable way to provide distributed searching across diverse Digital Libraries.

7 Acknowledgments

References

1. M. K. Buckland and C. Plaunt. Selecting libraries, selecting documents, selecting data. In *Proceedings of the International Symposium on Research, Development & Practice in Digital Libraries 1997, ISDL 97, Nov. 18-21, 1997, Tsukuba, Japan*, pages 85–91, Japan, 1997. University of Library and Information Science.
2. J. Callan. Distributed information retrieval. In W. B. Croft, editor, *Advances in Information Retrieval: Recent research from the Center for Intelligent Information Retrieval*, chapter 5, pages 127–150. Kluwer, Boston, 2000.
3. W. S. Cooper, F. C. Gey, and A. Chen. Full text retrieval based on a probabilistic equation with coefficients fitted by logistic regression. In D. K. Harman, editor, *The Second Text Retrieval Conference (TREC-2)*, pages 57–66, Gaithersburg, MD, 1994. NIST.
4. J. C. French, A. L. Powell, J. P. Callan, C. L. Viles, T. Emmitt, K. J. Prey, and Y. Mou. Comparing the performance of database selection algorithms. In *SIGIR '99*, pages 238–245, 1999.
5. J. C. French, A. L. Powell, C. L. Viles, T. Emmitt, and K. J. Prey. Evaluating database selection techniques: A testbed and experiment. In *SIGIR '98*, pages 121–129, 1998.
6. L. Gravano, H. García-Molina, and A. Tomasic. GlOSS: text-source discovery over the Internet. *ACM Transactions on Database Systems*, 24(2):229–264, 1999.
7. Jamie Callan and M. Connell. Query-based sampling of text databases. Technical report, Center for Intelligent Information Retrieval, Dept. of Computer Science, University of Massachusetts, 1999. Technical Report IR-180.
8. R. R. Larson. Distributed resource discovery: Using Z39.50 to build cross-domain information servers. In *JCDL '01*, pages 52–53. ACM, 2001.
9. R. R. Larson. Cheshire II at INEX: Using a hybrid logistic regression and boolean model for XML retrieval. In *Proceedings of the First Annual Workshop of the Initiative for the Evaluation of XML retrieval (INEX)*, page IN PRESS. DELOS workshop series, 2003.
10. Y. Lin, J. Xu, E.-P. Lim, and W.-K. Ng. Zbroker : A query routing broker for z39.50 databases, 1999.
11. A. L. Powell. *Database Selection in Distributed Information Retrieval: A Study of Multi-Collection Information Retrieval.* PhD thesis, University of Virginia, Virginia, 2001.
12. H. Varian and P. Lyman. How much information? Available as *http://sims.berkeley.edu/research/projects/how-much-info/.*, 2002.