Comparing the Performance of Collection Selection Algorithms

ALLISON L. POWELL and JAMES C. FRENCH
University of Virginia

The proliferation of online information resources increases the importance of effective and efficient information retrieval in a multicollection environment. Multicollection searching is cast in three parts: collection selection (also referred to as database selection), query processing and results merging. In this work, we focus our attention on the evaluation of the first step, collection selection.

In this article, we present a detailed discussion of the methodology that we used to evaluate and compare collection selection approaches, covering both test environments and evaluation measures. We compare the *CORI*, *CVV* and *gGlOSS* collection selection approaches using six test environments utilizing three document testbeds. We note similar trends in performance among the collection selection approaches, but the *CORI* approach consistently outperforms the other approaches, suggesting that effective collection selection can be achieved using limited information about each collection.

The contributions of this work are both the assembled evaluation methodology as well as the application of that methodology to compare collection selection approaches in a standardized environment.

Categories and Subject Descriptors: H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*selection process*; H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*performance evaluation (efficiency and effectiveness)*

General Terms: Experimentation, Measurement, Performance

Additional Key Words and Phrases: Collection selection, distributed information retrieval, database selection, distributed text retrieval, metasearch engine, resource discovery, resource ranking, resource selection, server selection, server ranking, text retrieval

1. INTRODUCTION

The growth of the Internet and federated digital libraries has increased attention on the problem of retrieving data items found in multicollection

Report Documentation Page	*Form Approved* *OMB No. 0704-0188*

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE **2003**	2. REPORT TYPE	3. DATES COVERED **00-00-2003 to 00-00-2003**
4. TITLE AND SUBTITLE **Comparing the Performance of Collection Selection Algorithms**		5a. CONTRACT NUMBER
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)		5d. PROJECT NUMBER
		5e. TASK NUMBER
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of Virginia,Department of Computer Science,151 Engineer's Way,Charlottesville,VA,22904-4740**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR'S ACRONYM(S)
		11. SPONSOR/MONITOR'S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT **Approved for public release; distribution unlimited**		
13. SUPPLEMENTARY NOTES		
14. ABSTRACT		
15. SUBJECT TERMS		

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES **45**	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**			

Standard Form 298 (Rev. 8-98)
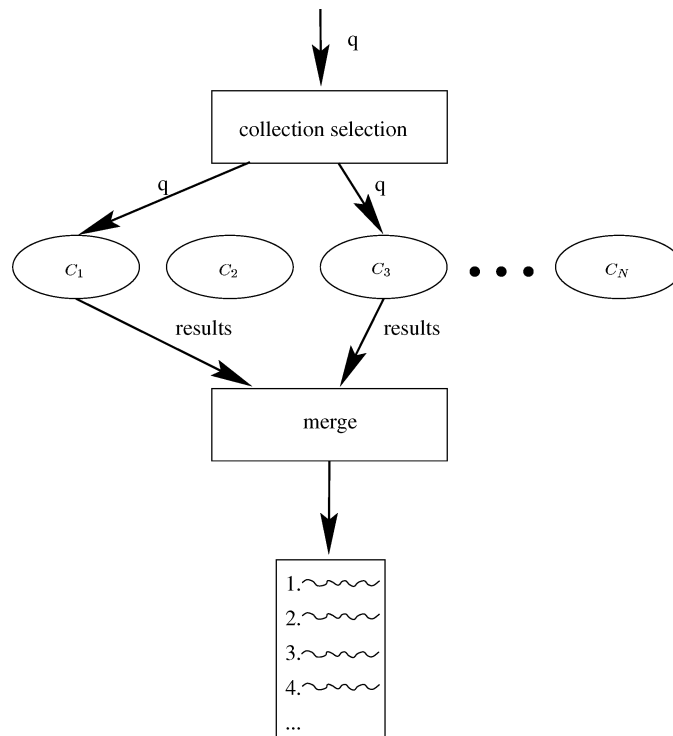Prescribed by ANSI Std Z39-18

Fig. 1.   An illustration of multicollection retrieval. The collection selection mechanism routes query q to collections C_1 and C_3. Query q is executed at those collections, then the two results lists are merged into a single, coherent list for presentation to the user.

environments. For reasons of efficiency, intellectual property or copyright, it may not be feasible or desirable to assemble all data items of interest at a central storage location or even to provide access through a centralized index. Within the information retrieval community, the problem of retrieving data items from a set of collections has typically been referred to as *distributed* information retrieval. We will refer to the problem as multicollection information retrieval to emphasize the fact that collections may or may not be physically distributed. While physical distribution is a common scenario, it is not a requirement.

The problem of data item retrieval in a multicollection environment can be broken down into three major subproblems, illustrated in a simplified fashion in Figure 1. Given a set of collections to which a user's query might be sent, the first subproblem is to choose the collections to be searched. We refer to this subproblem as the *collection selection* step. In our example, collections C_1 and C_3 are selected. This step becomes increasingly important as the number of collections grows or if the collections charge for access. The second subproblem is to forward the user's query to the selected collections. This step is challenging on several fronts. First there are the challenges faced in traditional information

retrieval—effectively and efficiently retrieving high-quality results from each of the collections. Additional challenges may be presented by heterogeneous environments where the underlying collections may use different query syntax or different retrieval models. The third subproblem is to take the individual results lists from each of the selected collections and to merge those results into a single coherent list of results to be presented to the user. In this article, we focus upon the collection selection subproblem.

The basic subproblems are similar between World Wide Web metasearch and multicollection retrieval over more stable collections; however, there are important differences. For example, in WWW metasearch, the problem of identifying the set of candidate collections may be more challenging and the contents of the collections may change. Collection selection has been applied to WWW environments [Gauch et al. 1996; Dreilinger and Howe 1997; Craswell et al. 2000]; however, our work focuses on a comparison of different collection selection approaches in an environment made up of stable and accessible collections.

At the time that we began this work, a number of techniques for collection selection had been proposed and independently evaluated; however, the evaluations varied in the underlying data, the performance goals and the evaluation methods. Due to the wide variety of test environments, it was difficult to directly compare the different published evaluation results. We had a number of goals for this work. The first goal was to demonstrate a uniform methodology for the study of collection selection approaches and their relative performance. The second, and more important, goal was to gain insight into both the collective and individual behavior of these algorithms. This article reports efforts to facilitate and perform direct comparisons of collection selection techniques. Comparisons of three collection selection techniques, *CORI* [Callan et al. 1995], *gGlOSS* [Gravano and García-Molina 1995; Gravano et al. 1999] and *CVV* [Yuwono and Lee 1997], using six test environments made up of three sets of collections and two query formulations, are reported.[1]

We begin in Section 2 with a discussion of related work. In Section 3, we describe our experimental environment, including the testbeds and queries, in considerable detail so that anyone interested in using these testbeds will have full information. Section 4 provides a summary of evaluation approaches and measures that were detailed in previous work. In Section 5, we describe the *CORI*, *gGlOSS* and *CVV* approaches in detail then present direct comparison results and analysis, including the effect of testbed features on performance. Section 6 concludes. The contributions of this work are both the assembled evaluation methodology as well as the application of that methodology to compare collection selection approaches in a standardized environment. The methodology and environments can be used to place any new collection selection approach in context.

---

[1]This article expands upon a previously-reported comparison of the *gGlOSS* and *CORI* collection selection approaches in a single test environment using a single query formulation [French et al. 1999b].

2. RELATED WORK

A number of different approaches for database or collection selection have been proposed and individually evaluated. Direct comparisons of these collection selection approaches is complicated by the variety of experimental environments and evaluation measures that have been used by different research groups. In addition, the methodology for evaluating collection selection is not yet as standardized as the methodology for evaluating document retrieval.

Collection selection approaches can be divided into three major classes based on their overall approach or evaluation approach. One group of approaches attempts to characterize the document-query similarities of the documents that would be returned if a query were sent to a collection C. These approaches typically have the stated goal of locating collections with a large number of similar documents or with highly similar documents. A second group of approaches use a relevance-based evaluation methodology, measuring the degree to which approaches identify collections that have a large number of relevant documents with respect to the query. Finally, a third group of approaches incorporates additional considerations, for example the cost to search a collection or the expected response time of a collection.

We will discuss a number of different collection selection approaches individually below, grouped by the three classes described above. Three of the approaches, *CORI* [Callan et al. 1995], *CVV* [Yuwono and Lee 1997] and *gGlOSS*[2] [Gravano and García-Molina 1995; Gravano et al. 1999] were evaluated in a common environment by French et al. [1998, 1999b] and [Callan et al. 2000], who found that there was significant room for improvement in all approaches, especially when very few collections were selected. Expanded versions of those experiments are presented in this article. When introducing those experiments, we will present a much more detailed discussion and analysis of the *CORI*, *CVV* and *gGlOSS* algorithms; summaries are provided here to place those approaches in the context of other related work.

2.1 Selecting Collections with Matching or Highly Similar Documents

The philosophy behind approaches that attempt to characterize the query-document similarities of documents within a collection was stated by Gravano and García-Molina [1995], who argued that "the best we can hope for any tool like *gGlOSS* is that it predicts the answers that the databases will give when presented with a query." A number of approaches have a similar goal. We summarize some of those approaches here, then briefly discuss environments in which they are applicable and environments for which they are less well suited.

Gravano et al. [1994] introduced *GlOSS*, the Glossary of Servers Server, which operates in an environment of Boolean information retrieval systems. *GlOSS* was later generalized to *gGlOSS* to handle the vector space information retrieval model [Gravano and García-Molina 1995]. *gGlOSS* consists of

---

[2]*gGlOSS* was later renamed *vGlOSS* [Gravano et al. 1999], but we will continue to refer to it as *gGlOSS* for consistency with our previously published work.

multiple parameterized collection selection algorithm implementations with the goal of estimating the sum of query-document similarities for a collection. *gGlOSS* needs two vectors of information from each collection in order to make its estimates: the document frequency df_j for each term t_j and the sum of the term weights w_{ij} of each term over all documents d_i in the collection.

The D-WISE multicollection retrieval system considered collection selection, query forwarding and results merging [Yuwono and Lee 1997]. Yuwono and Lee referred to the collection selection portion of their work as the *Cue Validity Variance (CVV)* ranking method. The goal of the *CVV* ranking method is to identify collections with a high concentration of query terms.

Meng et al. [1998] proposed a collection selection approach with goals that were similar to those of Gravano and García-Molina [1995]. Given a multicollection environment, their goal was to estimate the number of documents in the collection that would have a similarity to query q greater than some threshold *if a global similarity function had been employed*. While they use the information differently, Meng et al. require the same statistical information about each collection as Gravano and García-Molina–both document frequency information and average term weight information is required. In a series of papers, this work was later expanded and some assumptions were relaxed or modified [Liu et al. 1999; Meng et al. 1999; Yu et al. 1999a, 1999b].

The work of Baumgarten falls somewhat between the class of experiments described here and the class described in the next section. Baumgarten [1997] proposed a probabilistic model for multicollection information retrieval, assuming that the underlying collections make use of probabilistic information retrieval systems [Baumgarten 1997]. While document relevance is later used for evaluation [Baumgarten 1999], the goal of the overall approach is to maintain the overall top-ranked l documents that would be retrieved in a search of a single collection containing all documents or by selecting all collections, while at the same time restricting search to the collections that actually contribute documents to the set of l documents.

The majority of our experiments evaluate the degree to which collection selection approaches can locate collections with *relevant* documents instead of highly similar documents. However, we have studied the *gGlOSS* approach in detail [French et al. 1998; Powell 2001] and use it as a representative for approaches that utilize document term weight information and that attempt to locate collections with highly similar documents.

Traditionally, multicollection information retrieval performance has been compared to single-collection performance. While operational multicollection environments have yet to exceed the performance seen in equivalent single collection environments, recent experiments have shown that multi-collection retrieval performance has the potential to outperform single-collection retrieval [Powell et al. 2000; Craswell et al. 2000]. Approaches that are designed to replicate single-collection performance have two potential weaknesses. First, these approaches may search a large subset of the collections if the globally most similar documents are widely spread across the collections. Second, depending on the degree to which they are capable of replicating single-collection

performance, these approaches may also not be able to take advantage of the potential for higher multicollection performance.

2.2 Selecting Collections with Relevant Documents

The next group of collection selection approaches were proposed with the goal of identifying collections containing relevant documents. These approaches typically (but do not always) require less statistical information than collection selection approaches with the goal of identifying collections with highly similar documents.

CORI [Callan et al. 1995] is the collection selection mechanism associated with the Inquery [Callan et al. 1992] information retrieval system. In general, *CORI* treats collections as *virtual documents* using document frequency (*df*) and inverse collection frequency (*icf*) information. Collection selection can be considered as a sort of "document retrieval" over the set of virtual documents.

The work of Voorhees et al. [Voorhees et al. 1994, 1995; Voorhees 1995; Voorhees and Tong 1997] is more closely associated with results-merging; however, this work also has interesting collection selection aspects. Most selection approaches do not specify the number of documents to be retrieved from a selected collection. Instead, the proportion of documents from a given collection present in the merged results list is an artifact of the merge strategy. In contrast, Voorhees et al. [1995] defined two approaches for determining the number of documents to be retrieved from each collection. Because that number may be zero, these approaches serve as a collection selection step.

Xu and Callan [1998] focused on the nature of queries used for collection selection. Their premise was that queries intended for document retrieval (often containing only a few terms) are not appropriate for collection selection and that poor collection selection performance hinders multicollection document retrieval performance. They investigated the inclusion of phrase information and the use of query expansion for collection selection.

Xu and Croft [1999] argued that clustering may be necessary to create multicollection environments suitable for effective collection selection. They compared single-collection performance to four different approaches for constructing collections and collection representations.

2.3 Additional Considerations

Most collection selection approaches are based upon statistical information about the collections and are concerned primarily with locating relevant or highly similar documents. However, there are approaches that incorporate additional information; for example, considering both textual and nontextual information Dolin et al. [1997, 1998, 1999] or set different goals, for example, efficiency [Moffat and Zobel 1995], minimizing the cost of retrieving documents [Fuhr 1999] or accessing collections [Hawking and Thistlewaite 1999]. Craswell et al. [2000] argued that the retrieval performance at a collection should be incorporated into the collection selection step.

2.4 Additional Issues in Multicollection Retrieval

There are a number of issues that cut across many multicollection retrieval approaches, although they are not always explicitly mentioned. These issues include the effect of the use of different information retrieval systems at the underlying collections and information used for indexing collections. Of particular interest to us is the issue of collection representations for collection selection, the information used for those representations and how that information is acquired. If applying collection selection approaches to an operational environment, it is easier to obtain simple information such as *df* values than more detailed document indexing information.

2.4.1 *Heterogeneous Collections.* There are many ways in which the individual collections in a multicollection environment might differ. Collections can employ different document indexing techniques and different query processing techniques. The underlying search engine at one collection might support options not allowed by others. The acceptable query formats may differ among the search engines at the collections. The range of document scores can also vary by collection. In the context of describing the STARTS Internet metasearching protocol, Gravano et al. discussed these issues in detail [Gravano et al. 1997]. Here, we mention a few specific issues that have implications for our experiments.

Some approaches assume homogeneous underlying search engines, for example, *gGlOSS* assumes that all collections use the same similarity function to compute query-document similarities [Gravano and García-Molina 1995]. The general approaches that we consider do not assume homogeneous collections. In practice, this assumption is only feasible when the same individual or organization controls all collections in the multi-collection environment.

With the exception of *gGlOSS*, the approaches that we study in this work do not require document term weight information. In general, the approaches we study require only document frequency information (the number of documents containing each term) and other information that remains consistent even if different search engines or query formats are used at the underlying collections. However, there is still one issue that we must be aware of. Because collection selection indexes may be built from information provided by collection indexes, tokenizing, stopping and stemming can have implications for collection selection. Differences here can lead to incompatible vocabularies.

2.4.2 *Collection Representations.* Collection selection is difficult partly because collection selection algorithms do not typically have access to the full contents of a collection. Instead, they utilize summary statistical information about the collections. This has been referred to as *lexicon inspection* [Zobel 1997] and as *language modeling* [Callan et al. 1999; Xu and Croft 1999]. We will refer to the summary information about a collection as a *language model (LM)*. For our experiments, each collection C_i is represented by a corresponding language model LM_i. We denote the set of language models as $\mathcal{LM} = \{LM_1, LM_2, \ldots, LM_N\}$.

Collection selection algorithms differ in the type of information that they require in the language models. For example, as we mentioned earlier, *gGlOSS*

needs two vectors of information from each collection in order to make its estimates: the document frequency df_j for each term t_j and the sum of the term weights w_{ij} of each term over all documents d_i in the collection. *CORI* and *CVV* also utilize document frequency information.

A number of protocols for describing collections have been proposed, including those that assume the cooperation of collections [Gravano et al. 1997; Hawking and Thistlewaite 1999; Powell and Fox 1998] and those that acquire collection statistics from collections that are not actively cooperating [Callan et al. 1999; Lin et al. 1999; Liu et al. 1999; Xu et al. 1998; Ipeirotis and Gravano 2002]. Callan et al. [2000], Craswell et al. [2000], and Ipeirotis and Gravano [2002] have reported comparisons of the effect of sample-based language models on collection selection approaches.

3. EXPERIMENTAL ENVIRONMENT

One goal for this work was to describe a methodology for the study of collection selection approaches. Our discussion includes experimental environments and their features and evaluation measures. We begin here with our experimental environments and the underlying documents and queries. Later, in Section 3.4, we discuss the limitations of the test environment and its applicability to real-world scenarios.

3.1 TREC Data and Queries

Traditional (single collection) information retrieval test collections usually contain a set of data items $\mathcal{D}$, a set of queries $\mathcal{Q}$ and a set of relevance judgments $\mathcal{J}$. In most widely-available information retrieval test collections, the data items are text documents. Each query represents a statement of a user's information need and for each query, the relevance judgments identify the set of data items that are relevant to the query, that is, the data items that satisfy the information need.

Multicollection test environments generally contain the same components as traditional information retrieval test collections, with the exception that the documents are organized into more than one collection. The potential exists for duplicate data items within a collection; however, in most previously reported relevance evaluation-based experiments, the collections are a partition of the documents. The problem of duplicates is an important one and affects both collection selection and results merging. This issue bears consideration in operational environments; however, we chose to focus on partitioned collections for these experiments.

The large number of documents available in the TREC/TIPSTER data, plus the availability of relevance judgements have made it a popular choice as a basis for constructing multicollection test environments. At the time that we began these experiments, most of the test environments used in published work had fewer than 100 collections. This, plus an initial interest in a test environment with a controlled temporal component to the collections led us to construct a different test environment for our experiments. Because we were interested in both efficiency *and* effectiveness, and in evaluating systems using a large

number of collections, the TREC/TIPSTER data was the only realistic starting point. Given the availability of TREC topics from which queries could be created and TREC relevance judgements, we began by constructing a testbed, referred to here as *SYM-236*. Over the course of the experiments reported in Section 5, we noted that some collection selection algorithms have a tendency to prefer collections with a large number of documents. We created an additional testbed, *UDC-236*, to study this effect. We later added a testbed created at the University of Massachusetts, referred to as *UBC-100*.

In this section, we will describe and characterize the test environments that are used in the experiments presented in Section 5. We will focus mostly on the three testbeds that are components of the test environments. We start by describing the underlying TREC data from which each set of documents $\mathcal{D}$ will be drawn. The three testbeds described in this section contain no duplicate documents. We then discuss the TREC topics, the subset of the topics used for our experiments and the two sets of queries constructed using those topics. Finally, we describe the three testbeds (sets of collections) used in our experiments and discuss features of those testbeds.

3.1.1 *TREC Data.* The Text REtrieval Conferences (TREC) are a series of annual conferences co-sponsored by the National Institute of Standards and Technology (NIST) and DARPA. Each year, groups from industry, academia and government undertake a set of retrieval tasks, using a supplied set of documents and queries,[3] then meet to discuss the results.

The *SYM-236*, *UDC-236* and *UBC-100* testbeds were all constructed using data available to participants in the TREC-4 [Harman 1995] conference. To summarize, this data is approximately 3 GB of text spread over several years and from seven primary sources: AP Newswire (AP), Wall Street Journal (WSJ), Computer Select (ZIFF), the Patent Office (PAT), San Jose Mercury News (SJMN), Federal Register (FR), and Department of Energy (DOE). This data was distributed on three CD-ROMs and segments of data are sometimes referenced using the disk number on which they were distributed. Much of the TREC data is from news sources and so has easily identifiable date components. The one undated collection is the set of documents from DOE.

3.1.2 *Queries.* The TREC data is distributed along with a set of statements of information need and an accompanying set of relevance judgments. In TREC parlance, the statements of user information need are referred to as *topics*. Unlike average user queries (especially Internet search engine queries), most TREC topics are very detailed statements of information need. In many cases, they resemble detailed instructions to a professional searcher. The topics may also contain instructions to the TREC judges of what constitutes a relevant document. This information is contained in fields, all or some of which can be used to construct the query that is actually issued to a collection. As a result, the formulation of actual queries used in published results can differ widely. Later in this section, we discuss the approaches that we employed when creating

---

[3]The data available to TREC participants is generally referred to as TREC/TIPSTER or simply TREC collections.

queries. We will refer to our formulations as *queries* while retaining the TREC terminology of *topic* to refer to the original statement of information need. We retain the TREC topic numbering for our queries. We will apply the relevance judgments for a topic to each query generated from that topic.

In many years, the TREC conference has introduced new document sets. In every year, new topic sets have been introduced, generally in batches of 50 topics per set. Because of the evolutionary nature of the conference, relevance judgments are not available for all combinations of topic and document sets. Through TREC-4, there were a total of 250 topics with relevance judgments over some portion of the TREC documents.[4] Due to constraints of relevance judgment coverage, we use only topics 51–150 in our experiments. This maximizes the number of documents available for collection creation.

In the experiments reported here, we use two query formulation strategies, producing two sets of 100 queries each. We will refer to these formulations as "short" and "long". The short queries, Q_s, were constructed using the Title field of the TREC topics. These queries average 3.5 words per query and are a very brief description of the information need. The long queries, Q_l, were constructed using the Concepts field of the TREC topics and average 21 words per query. The Concepts field contains words, phrases and especially proper names that might be found in relevant documents. The terms found in the Concepts field have the potential to resemble very well-thought-out user queries; however, our resulting long queries contained more terms than queries typically received from real users [Abdulla et al. 1997; Jansen et al. 1998; Silverstein et al. 1999; Spink and Saracevic 1997]. We chose to use both short and long queries to account for the cases of longer, more detailed queries while also considering the shorter queries typically found in operational environments.

In some early experiments [French et al. 1998], we used an even longer query formulation. Those queries used all of the text available in the TREC topics and averaged 49 terms per query. We later determined that the short and long approaches were more commonly used in other research, so switched to those approaches. As a result, the *gGlOSS* results reported here differ slightly from those reported earlier.

3.2 The *SYM-236*, *UDC-236* and *UBC-100* Testbeds

Each of our experimental testbeds is a set of N collections, $\mathcal{C} = \{C_1, C_2, \ldots, C_N\}$. A testbed can be represented as data items, $\mathcal{D}$, plus a data item to collection map, $R_{\mathcal{D} \to \mathcal{C}}$, from which the set of collections, $\mathcal{C}$, can be constructed. The data item to collection map may be externally supplied, or constructed using some rule designed to create a testbed with some desired characteristic(s). We will use the notation $|\mathcal{D}|$ to denote the number of data items in the collections.

In this section, we describe the *SYM-236*, *UDC-236* and *UBC-100* testbeds. A more detailed description of the testbeds can be found in Powell [2001]. For all three testbeds, $\mathcal{C}$ is a partition of $\mathcal{D}$.

---

[4]A complete discussion of topic coverage can be found in French et al. [1998] and Powell [2001].

3.2.1 SYM-236 *(Source-Year-Month)*. When constructing the *SYM-236* testbed, we were working under a set of goals and requirements that we discussed in detail in French et al. [1998]. In short, we wanted to create a natural partition of the documents into collections, while at the same time producing at least 100 collections. Our decision to use date and source of publication as simple criteria by which to organize a collection of documents had the most effect on collection composition. We wanted to maintain the option to create composite collections containing documents published during the same time period. We did not do this for the experiments reported here, but the requirement affected the creation of the *SYM-236* testbed. As a result of these goals and requirements, there were some special cases in the selection of the documents in $\mathcal{D}$ and in the creation of the mapping relation of documents to collections, $R_{\mathcal{D} \to \mathcal{C}}$. The general rule for creating $R_{\mathcal{D} \to \mathcal{C}}$ was to partition the documents on TREC CDs 1, 2 and 3 by publishing source, then by year and month of the publication date. For example, all AP Newswire articles from February of 1988 were placed in the same collection. The exceptions and special cases primarily affected publication date resolution and which documents from TREC CDs 1, 2 and 3 were included in $\mathcal{D}$. For example, no documents from the Department of Energy publishing source (DOE) were included in $\mathcal{D}$ because the DOE data is undated.

The net result of combining the particular attributes of the TREC data and our own requirements was a partition comprised of 236 document collections derived from some but not all of TREC disks 1, 2, and 3. Summary characteristics of this partition are given below.

—$\mathcal{D}$—Data items are text documents from TREC CDs 1, 2, and 3 minus DOE documents (documents contain no date) and ZIFF documents from disk 3 (these documents overlapped temporally with ZIFF documents from disks 1 and 2).
—$R_{\mathcal{D} \to \mathcal{C}}$ can be found at `http://www.cs.virginia.edu/~cyberia/testbed.html` labelled trec123-236-by_source-by_month.
—$|\mathcal{D}| = 691,058$ documents in the testbed subdivided into
—$N = 236$ collections.

3.2.2 UDC-236 *(Uniform-Document-Count)*. At the time that *SYM-236* was created, an equally viable, alternative partitioning strategy would have split the data into N equal-sized collections. This partitioning approach has attractive characteristics in that (1) it is easy to control the number of collections and (2) collection size is held constant. We chose to pursue the *SYM-236* strategy first; however, we noted during early experiments that the vast size variation of *SYM-236* proved interesting during the evaluation of collection selection algorithms. The *UDC-236* (Uniform-Document-Count) testbed was designed to control for the tendency of some collection selection algorithms to prefer collections with a large number of documents. The *UDC-236* testbed contains exactly the same documents as *SYM-236*; however, the documents were organized into collections containing roughly 2,900 documents each, ordered as they appeared on the TREC CDs, and with the restriction that all of the

Table I. Summary Statistics for the Testbeds.

Testbed	Data Items per coll.			Bytes per collection		
	Min.	Avg.	Max.	Min.	Avg.	Max.
UBC-100	752	10,782	39,723	28,070,646	33,365,514	41,796,822
SYM-236	1	2,928	8,302	7,668	11,789,423	34,782,134
UDC-236	2,891	2,928	3,356	7,138,629	11,789,423	133,206,035

documents in a collection were from the same primary source. This testbed also contains 236 collections.

UDC-236 is summarized below.

—$\mathcal{D}$—Data items are exactly the same as those for *SYM-236*.

—$R_{\mathcal{D} \to \mathcal{C}}$ can be found at `http://www.cs.virginia.edu/~cyberia/testbed.html` labelled trec123-236-eq_doc_counts.

—$|\mathcal{D}| = 691,058$ documents in the testbed subdivided into

—$N = 236$ collections.

3.2.3 UBC-100 *(Uniform-Byte-Count)*. The *UBC-100* testbed was constructed at the University of Massachusetts and was not influenced by the goals and restrictions that affected the construction of *SYM-236*. All of the documents from TREC CDs 1, 2 and 3 were included in this testbed. Data items were organized into collections of roughly 30 megabytes each, ordered as they appeared on the TREC CDs, and with the restriction that all of the data items in a collection were from the same primary source. *UBC-100* has previously been used with other testbeds to study the scalability of *CORI* collection selection [French et al. 1999b] and the effect of sampled language models on collection selection [Callan et al. 2000].

UBC-100 is summarized below.

—$\mathcal{D}$—Data items are text documents from TREC CDs 1, 2, and 3.

—$R_{\mathcal{D} \to \mathcal{C}}$ can be found at `http://www.cs.cmu.edu/~callan/Data/` labelled trec123-100-bysource-callan99.v2a.

—$|\mathcal{D}| = 1,078,166$ documents in the testbed subdivided into

—$N = 100$ collections.

3.2.4 *A Summary of the Testbeds.* General characteristics of the testbeds appear in Table I. This table shows both features of the testbeds and the effects of particular constraints in testbed creation. The *UBC-100* and *UDC-236* testbeds are constructed to contain collections of approximately 30 MB and collections of approximately 2,900 documents,[5] respectively. Depending on individual document size, fixing one of these values can still result in variability in the other. Due to the goals and restrictions employed during construction, there was more variability in the sizes of the *SYM-236* collections. For example, there were generally few Patent Office documents in a given month, but there were often many articles from the AP Newswire.

---

[5]While creating *UDC-236*, we did not mix documents from different publishing sources in the same collection. As a result, there are small differences in collection size.

These three testbeds represent three convenient ways to organize documents into collections or to partition a large collection into several smaller ones. Xu and Croft [1999, p. 256] expressed concern that the distribution of relevant documents in sets of collections such as these may adversely affect the efficiency or effectiveness of multi-collection retrieval; however, in experiments reported elsewhere [Powell et al. 2000], we did not experience such difficulties with these three testbeds.

3.3 Features of the Testbeds

3.3.1 *Document Distributions.*   The upper portions of Figures 2–4 provide a visual illustration of the distribution of documents in the *SYM-236*, *UDC-236* and *UBC-100* testbeds. There are a number of main features to note:

—The distribution of documents in the *SYM-236* testbed is very skewed. There are a number of very large collections with more than 8,000 documents per collection plus several dozen very small collections 1 to 20 documents in size. The very small collections are mainly derived from early PAT data. In addition to the large number of PAT collections with very few documents, the AP and SJM collections tend to have twice as many documents as the FR, WSJ, and some of the ZIFF collections. The size variation of the *SYM-236* testbed will prove important for identifying a feature of some collection selection algorithms.

—While the same documents are used, the distribution of collections per publishing source is very different in *UDC-236* than in *SYM-236*. There are only two PAT collections and over fifty each of AP and WSJ collections.

—The illustrations re-emphasize that there are only 100 *UBC-100* collections as opposed to 236 *SYM-236* and *UDC-236* collections.

—The distribution of documents in the *UBC-100* testbed is skewed, but in a different way from *SYM-236*. There are a few small PAT collections, but the most striking feature are the six DOE collections and two ZIFF collections that contain a very large number of very small documents.

3.3.2 *Relevant Document Distributions.*   Figures 2–4 also contain graphs that show the distributions of relevant documents in the collections. Figures 2–4 should be viewed sideways; the two graphs of each figure are aligned so that points and bars for each collection line up vertically. In the lower graph for each figure, we show the number of queries for which each collection contains at least one relevant document. Then, for each of those queries, we plot the mean number of relevant documents in the collection along with error bars. Taken together, these values provide a rough characterization of the distribution of relevant documents in the collections. Some observations follow:

—For all three testbeds, we found that the PAT collections contain very few relevant documents and contain relevant documents for relatively few queries.

Fig. 2.   The distribution of documents and the number of queries for which a collection contains relevant documents in *SYM-236*.

—We also note the same feature observed by Voorhees et al. [1995], namely that many relevant documents are found in AP or WSJ collections. For most queries, picking AP or WSJ collections is a reasonable heuristic. We also note that SJM collections qualify for thisobservation.
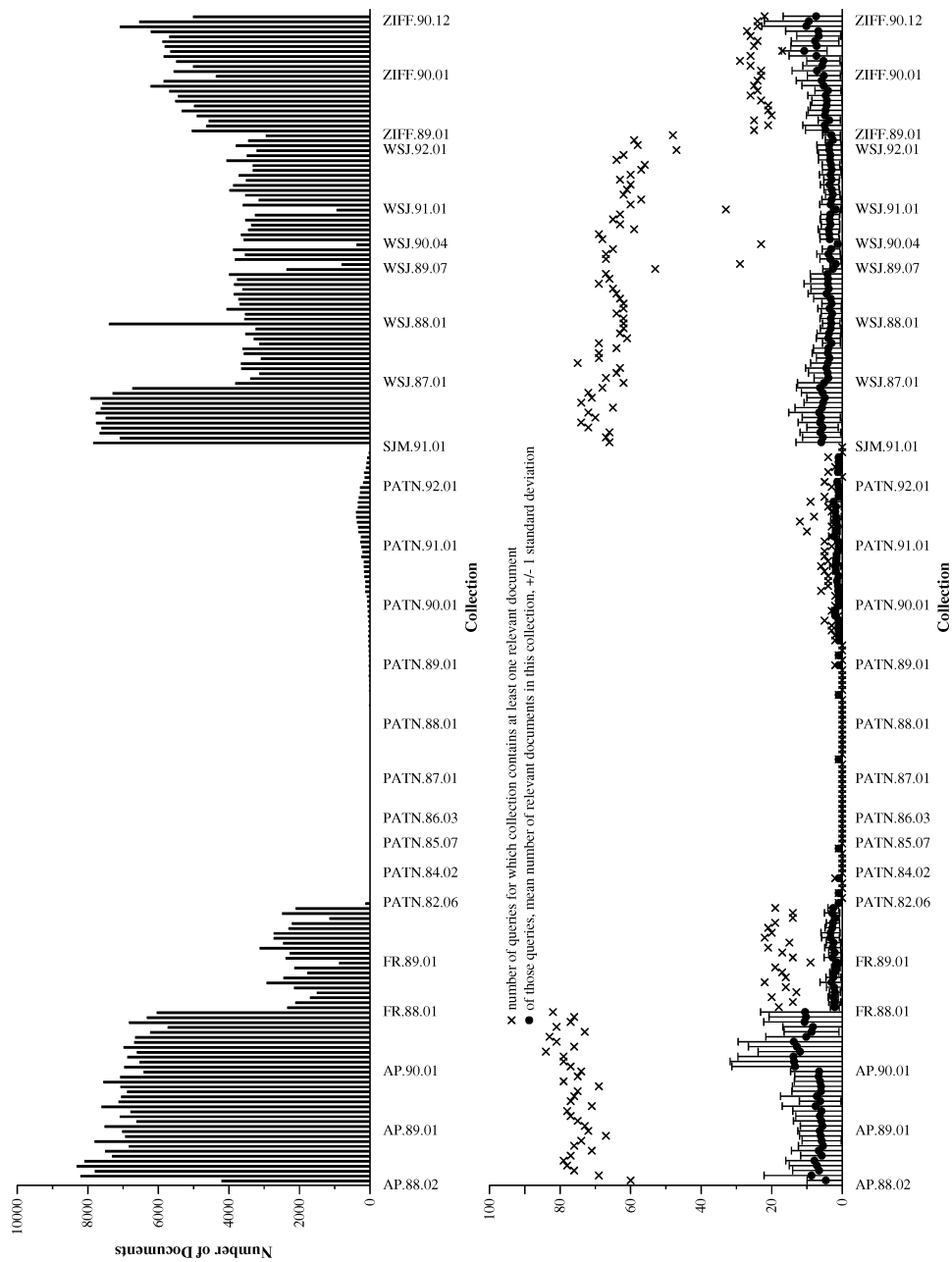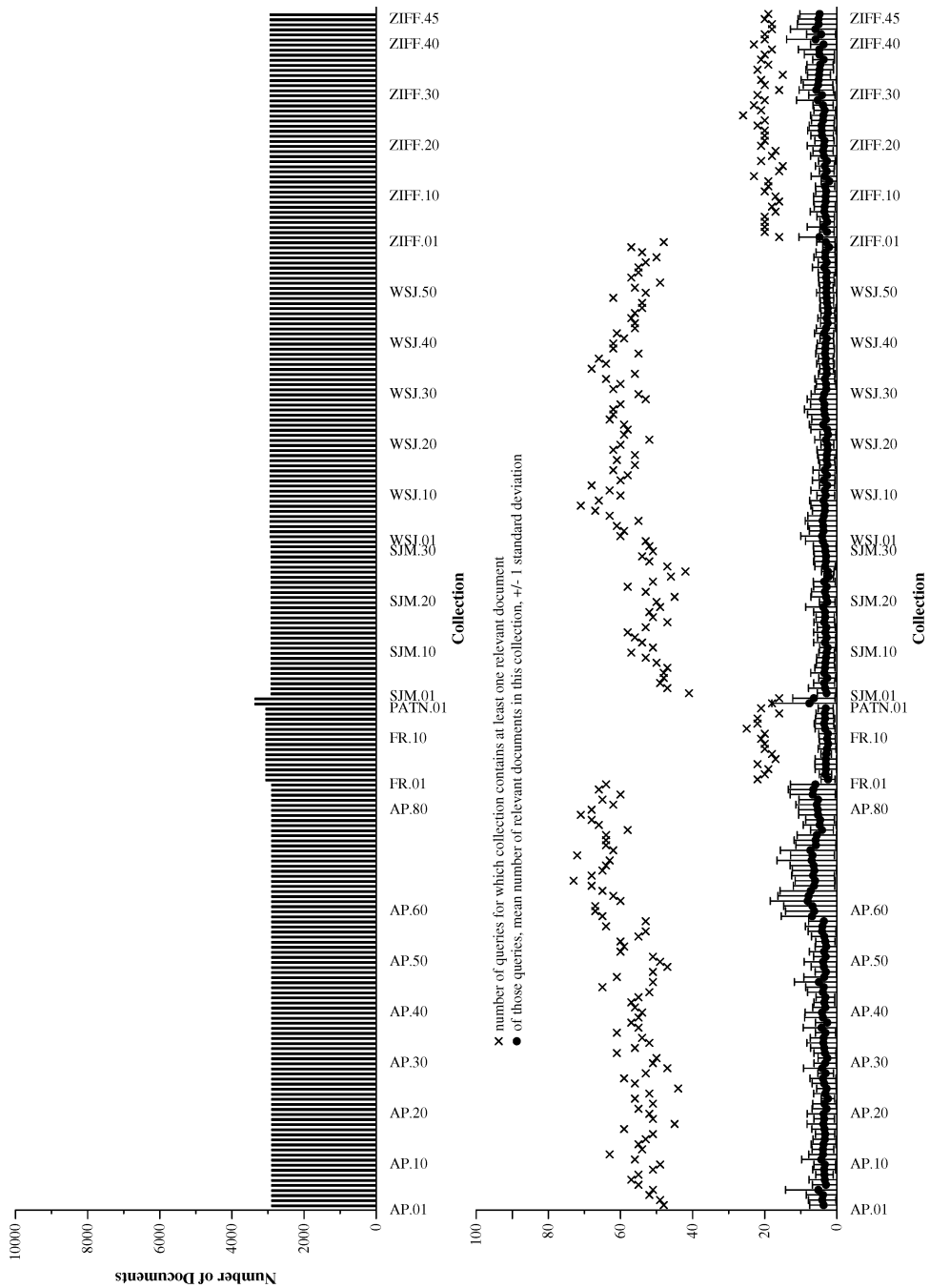
Fig. 3.   The distribution of documents and the number of queries for which a collection contains relevant documents in *UDC-236*.
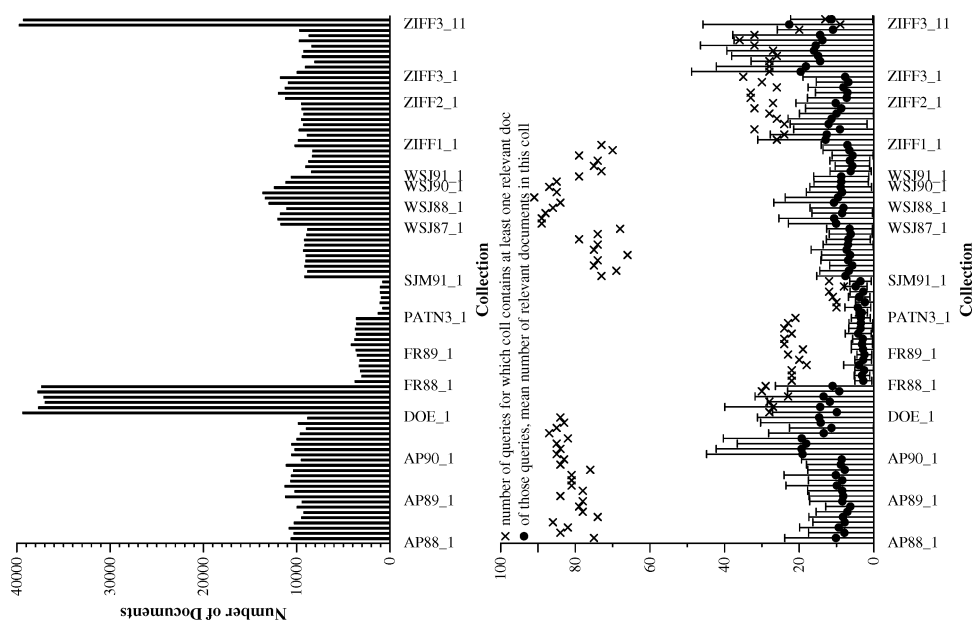
Fig. 4.   The distribution of documents and the number of queries for which a collection contains relevant documents in *UBC-100*.

—A more specific observation is that AP newswire articles published in 1990 tend to have a noticeably higher average number of relevant documents per query.

—Despite the very large number of documents per collection, the DOE collections of the *UBC-100* testbed tend to contain relevant documents for only a few queries. However, for those queries, the number of relevant documents in the DOE collections tends to be large but variable.

—The number of relevant documents in ZIFF collections tends to vary widely on a query-by-query basis.

We will consider the implications of these distributions of documents and relevant documents when we evaluate collection selection approaches in Sections 5.

3.4 Limitations of Test Environment

There are a number of objections one could raise regarding our test environment. Specifically these are centered around: (1) the choice of the TREC data; and (2) the way in which we decomposed the data into subcollections. We treat these separately below. The discussion is in the context of the TREC *ad hoc* retrieval task. That is implicitly the task we are examining in this work.

As we pointed out in Section 3.1, the TREC data was the only practical choice for our experiments. The TREC data is widely known, and can be used for effectiveness experiments because it includes queries and relevance judgments. An additional advantage is that the TREC data provides a single-collection environment to which we can compare our results. This provides a valuable context

in which to interpret the results of effectiveness experiments with multiple collections.

Any real-world environment of interest will typically have significant variability along many dimensions among which are the following:

Number of Collections. Our testbeds have 100 or 236 collections. We believe that this is a sufficient number to make direct measurements of effectiveness over an interesting range of collections. The trends observed also let us reason about the behavior of specific algorithms when the number of collections is increased.

One way to reason about the scalability of collection selection algorithms is to note that each collection is represented as a "document" where document frequencies have been substituted for term frequencies. Document ranking algorithms are known to be effective for very large collections of documents so by analogy one would expect collection ranking algorithms to be scalable for at least two orders of magnitude over our testbed size.

Size of Each Collection. Size may be captured in a variety of ways, for example, number of documents or number of bytes. We created testbeds having similar size according to each of these definitions. In addition, one of our testbeds exhibits a large variance in size. While these testbeds cover only a small portion of the size spectrum we feel that we have appropriately represented reasonable situations.

Topical Content of Collections. Our testbeds have significant topical heterogeneity. Generally one would expect a harder collection selection problem under these circumstances. We feel that our results are more broadly applicable because of this choice.

Overlap Among Collections. Our testbeds have no documents in common. This is representative of many interesting multicollection environments but even when it is not true the main effect is to require duplicate removal during the merge phase of searching. We feel that the effect on collection selection would not generally be appreciable if duplication is not too widespread. However, this is a conjecture that is open to investigation.

Underlying Query Engine. We make no assumption about the search engine technology used to index and search any specific collection. While that would certainly affect the outcome of retrieval effectiveness, if accurate collection statistics can be obtained then the search engine technology plays no role in collection selection effectiveness. All we assume is that we have access to document and term frequency statistics.

In summary, although there is really no "typical" multicollection scenario, we believe that our testbeds are representative of systems containing a few hundred collections such as will be found in medium-sized digital libraries or on corporate intranets. A recent study [French 2002] has shown that some vocabulary characteristics of categorized web data are very similar to the vocabulary characteristics of TREC data as measured by Araújo et al. [1997]. Thus, there is reason to believe that results derived from our testbeds will apply to specialized collections of web pages as well. Our testbeds do not model large heterogeneous web collections well. Thus we would not expect our results to apply to selection of collections held by the large search engines.

4. EVALUATION

In a multicollection retrieval environment, collection selection and document retrieval are often performed sequentially. Thus, the effects of collection selection performance have sometimes been evaluated via the document retrieval performance of the merged results. However, interpreting those results is not always straightforward. Poor document retrieval performance could be attributed to the collection selection step, the document retrieval approach at the selected collections or the merge step. For these experiments, we want to evaluate collection selection performance independently of other factors. In this section, we focus on measures used to evaluate collection selection directly. We give an overview of the evaluation approach, discuss general evaluation issues, then define the specific measures used in these experiments. A more detailed discussion of the evaluation measures, their features, the expected performance of random selection, and relationships among the measures can be found in French and Powell [2000].

4.1 Baselines and Estimators

Given some query q and a set of collections to which that query might be sent, the collection selection step may be viewed in two ways. Under one interpretation, the collection selection mechanism specifies the order in which the collections are searched. An alternate interpretation is that the collection selection mechanism chooses a subset of the collections to search. We use the former interpretation and assume that a collection selection mechanism seeking a subset of n collections would simply use the first n collections in the ranking.

To state the problem more formally, we have a set of N collections $\mathcal{C}$ where $\mathcal{C} = \{C_1, C_2, \ldots, C_N\}$ that we wish to search to satisfy some query q. A collection selection approach will produce a ranking of the collections in $\mathcal{C}$. The evaluation measures that we employ determine to what extent the collection ranking produced by a collection selection approach approximates some desired behavior.

We adopt the terminology of Gravano and García-Molina [1995] and refer to a ranking produced by a collection selection approach as an *estimated ranking*. A *baseline ranking* represents some desired behavior. Here we discuss baselines and estimators in general terms. Specifics of the baselines and estimators used in our evaluations are covered in much more detail in the context of the experiments in which they are used (see Section 5).

Baselines and estimators are both simply rankings of collections. The difference is one of interpretation. If a collection ranking is being used to represent some desired behavior then it is a baseline. If a collection ranking is being evaluated to determine if it exhibits the desired behavior, then it is an estimator. Generally speaking, baselines are created using information about collections that is not readily available in an operational setting, while estimators employ summary statistical information about the collections to create their rankings. However, the same collection ranking may be used as both a baseline and an estimator. For example, the *Ideal(0)* ranking was originally defined as a baseline for the *gGlOSS* collection selection algorithm [Gravano and García-Molina 1995]; however, for most experiments we use *Ideal(0)* as an estimator.

The purpose of this work is to compare the performance of a set of collection selection approaches proposed in the literature. The actual collection selection approaches evaluated are defined in Section 5. These approaches employ summary statistical information about a set of collections to produce a ranking of those collections. Given this estimated ranking, we can specify a subset of collections to be searched or a collection search order.

4.2 Merit

The baseline and estimated rankings can be cast in terms of *merit* and the concept of merit is used to define many of the evaluation measures that we use.

We have stated that a baseline ranking represents the desired behavior of a collection selection approach. Based on the desired behavior, we assume that each collection $C \in \mathcal{C}$ has some intrinsic merit, denoted $merit(q, C)$, with respect to a query q. Merit is related to the desired collection selection performance. Merit could be defined as the number of relevant documents in a collection, the proportion of relevant documents in a collection, the number of documents in a collection, the number of documents that have a given similarity to the query or any other assignment of values. A baseline ranking can be defined to be sorting collections in decreasing order of $merit(q, C)$.

Collection selection algorithms do not know the intrinsic merit of a collection with respect to a query. Rather, collection selection approaches can be viewed as a means with which to *estimate* that merit. An estimated ranking can be defined to be sorting collections in decreasing order of estimated merit.

Because the actual collection merits are not known, these estimated rankings based on estimated merit may not be the same as the baseline ranking based on actual (intrinsic) merit. One approach to evaluating a collection selection technique determines the degree to which the selection technique is able to produce collection orderings that approximate the baseline rankings.

4.3 General Evaluation Strategy

Before defining the specific evaluation measures we employ, we will cover some theoretical issues that arise in performing comparisons and some properties of baselines and estimators that can affect evaluation.

Given some goal baseline B and an algorithm producing an estimate, E, of that goal, we endeavor to determine the quality of the estimate by means of some measure $m(E, B)$ comparing the estimate to the goal. To make this discussion and later definitions of evaluation measures more concrete, we present an example of baseline and estimated rankings in Figure 5. Our example contains six collections, $\mathcal{C} = \{A, B, C, D, E, F\}$ and three queries $\mathcal{Q} = \{q_1, q_2, q_3\}$. In our example, we assume that the desired behavior is to locate collections in descending order of the number of relevant documents contained in the collection. Therefore, the intrinsic or baseline merit is the number of relevant documents in a collection with respect to a query. The estimated merits are produced by some hypothetical collection selection approach. Collections are sorted using those estimated merit values to create the estimated rankings shown. Note that for none of the queries do the estimated rankings exactly match the

Baseline Merits

	A	B	C	D	E	F
q_1	6	2	9	5	1	7
q_2	4	18	3	9	5	1
q_3	2	1	2	0	4	0

Estimated Merits

	A	B	C	D	E	F
q_1	0.7	0.4	0.6	0.2	0.1	0.5
q_2	0.2	0.8	0.4	0.7	0.9	0.1
q_3	0.2	0.1	0.3	0.4	0.5	0.2

Baseline Rankings

	1	2	3	4	5	6
q_1	C	F	A	D	B	E
q_2	B	D	E	A	C	F
q_3	E	A	C	B	D	F

Estimated Rankings

	1	2	3	4	5	6
q_1	A	C	F	B	D	E
q_2	E	B	D	C	A	F
q_3	E	D	C	A	F	B

Fig. 5.   An example baseline and estimate. Rankings are created using the corresponding merits.

baseline rankings. A given measure $m(E, B)$ is used to determine the degree and effect of the difference in the rankings.

Most evaluations are conducted over a query set $\mathcal{Q}$. The performance of an estimator with respect to a baseline is evaluated for each query. Overall results are usually presented as averaged summary measures of the following form:

$$\frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} m(E_q, B_q).$$

Note that $m(E_q, B_q)$ might itself already be an aggregate measure.

4.4 Specific Measures for Comparison

There is no general agreement on how this type of comparison should be done. The general problem is that we are given a baseline ranking for some query and a ranking produced by some collection selection algorithm. The goal is to decide how well the estimated ranking approximates the baseline ranking and to reveal potential performance implications of the quality of the approximation. We describe some of the approaches given in the literature and discuss new measures here.

4.4.1 *Recall and Precision Analogs.*   First, we discuss performance metrics that are analogous to the well known IR metrics of *recall* and *precision*. We begin by introducing some terminology and notation that tries to make this analysis neutral and generalizes it to include a variety of baselines.

Recall that for each query we provide a *baseline ranking* that represents a desired goal or query plan. Given some algorithm that produces an *estimated ranking*, our goal is to decide how well the estimated ranking approximates the baseline ranking.

To begin, we assume that each collection C in $\mathcal{C}$ has some merit, $merit(q, C)$, to a given query q. We expect the baseline to be expressed in terms of this merit; we expect the estimated ranking to be formulated by implicitly or explicitly estimating merit.[6]

---

[6]When we refer to *merit* and $merit(q, C)$, we mean the actual merit of a collection with respect to a query. We use the term *estimated merit* to refer to estimates produced by a collection selection approach.

Let C_{b_i} and C_{e_i} denote the collection in the ith ranked position of the baseline and estimated rankings respectively. Next we define two sequences, B and E, based on the merit of the collections in the two rankings. Let

$$B_i = merit(q, C_{b_i}) \text{ and } E_i = merit(q, C_{e_i})$$

denote the merit associated with the i-th ranked collection in the baseline and estimated rankings respectively. The total merit, M, is given by $M = \sum_{i=1}^{N} B_i$.

We note that, for viable baseline rankings, it should always be the case that

$$B_i \geq B_{i+1}, \quad i = 1 \cdots N - 1.$$

For the baselines discussed here, this is always true because we assume that the baseline ranking is determined by sorting the collections in decreasing order of merit for some appropriate definition of merit. However, it is not generally the case that $E_i \geq E_{i+1}$. The performance evaluation problem discussed here is an attempt to quantify the degree to which this is true for any estimated ranking.

This point needs a bit of explanation. The estimators will rank collections in decreasing order of *estimated* merit (as calculated by the estimator). However, note that E_i is the *actual* merit associated with C_{e_i}, that is, the merit used to create the baseline ranking. The degree to which $E_i \geq E_{i+1}$ reflects the accuracy of the algorithm's estimates of E_i.

Gravano and García-Molina [1995] defined $\mathcal{R}_n$ as follows:

$$\mathcal{R}_n(E, B) = \frac{\sum_{i=1}^{n} E_i}{\sum_{i=1}^{n} B_i}. \tag{1}$$

$\mathcal{R}_n(E, B)$ is a measure of how much of the available merit in the top n ranked collections of the baseline has been accumulated via the top n collections in the estimated ranking. This is a variant of the *normalized cumulative recall* measure defined by Tomasic et al. [1992] and later generalized by Gravano et al. [Gravano and García-Molina 1995; Gravano et al. 1999].

We propose an alternate definition of a recall-like measure that can be used to present performance results. First, we need one more definition. Let

$$n^* = k \text{ such that } B_k \neq 0 \text{ and } B_{k+1} = 0.$$

Intuitively, n^* is the ordinal position in the ranking of the last collection with non-zero merit; it is the breakpoint between the useful and useless collections. Clearly, $n^* \leq N$ and, moreover, the total merit, M, of a baseline is given by $M = \sum_{i=1}^{n^*} B_i$. With this definition we define our alternative recall metric as follows:

$$\hat{\mathcal{R}}_n(E, B) = \frac{\sum_{i=1}^{n} E_i}{\sum_{i=1}^{n^*} B_i} = \frac{\sum_{i=1}^{n} E_i}{M}. \tag{2}$$

The denominator is just the total merit contributed by all the collections that are useful to the query. Thus, $\hat{\mathcal{R}}_n(E, B)$ is a measure of how much of the total merit has been accumulated via the top n collections in the estimated ranking. This measure has also been proposed by Lu et al. [1996].

These two measures are clearly related. Because

$$\mathcal{R}_n(E, B)\sum_{i=1}^{n} B_i = \hat{\mathcal{R}}_n(E, B)\sum_{i=1}^{n^*} B_i, \tag{3}$$

we have $\mathcal{R}_n(E, B) \geq \hat{\mathcal{R}}_n(E, B)$ and $\mathcal{R}_{n^*}(E, B) = \hat{\mathcal{R}}_{n^*}(E, B)$.

Gravano and García-Molina [1995] have also proposed a precision-related measure, $\mathcal{P}_n(E, B)$. It is defined as follows:

$$\mathcal{P}_n(E, B) = \frac{|\{C_i \in Top_n(E)|merit(q, C_i) > 0\}|}{|Top_n(E)|}. \tag{4}$$

This gives the fraction of the top n collections in the estimated ranking that have non-zero merit. $Top_n(E)$ is just the set of collections given in the first n ranks. An alternative interpretation for $\mathcal{P}_n(E, B)$ is that it measures the degree to which collections with zero merit have been interleaved with those having nonzero merit.

In the remainder of the article, we simplify the notation by dropping all arguments to the measures when it is clear that we are referring to a specific algorithm's estimates (E) and measuring against a prespecified baseline (B).

To illustrate these recall and precision-based evaluation measures, consider the example shown in Figure 6 in which there are six collections, $\mathcal{C} = \{A, B, C, D, E, F\}$ and three queries $\mathcal{Q} = \{q_1, q_2, q_3\}$. This example is an extension of Figure 5 and Figure 5 is included as the upper portion of Figure 6. For each query, we wish to visit the collections in order of the number of relevant documents for that query. Therefore, the merit of a collection will be the number of relevant documents for a query.

Assume that for queries q_1, q_2 and q_3 there are 30, 40 and 9 relevant documents respectively, distributed as shown in the Merits-Baseline table in the upper left of Figure 6. Sorting the collections using merit produces the Rankings-Baseline table. Next assume that the estimator being evaluated produces the estimated merits and rankings shown in the Merits-Estimate and Rankings-Estimate tables. Note that the ranking produced by the estimator does not match the baseline ranking exactly; therefore, relevant documents will not be accumulated as quickly as they could be.

The middle portion of Figure 6 redisplays the baseline and estimate collection rankings and illustrates how the baseline merits are used to evaluate the estimate. This was alluded to in the definition of $B_i = merit(q, C_{b_i})$ and $E_i = merit(q, C_{e_i})$. Recall that while estimated merits are used to produce the estimated rankings, the actual merits (as used to compute the baseline) are substituted in when evaluating the estimator. This is because in our evaluation we want to determine how quickly we're accumulating actual merit (approximating the baseline in terms of the baseline merit). The example of Figure 6 makes this clearer. In our example, we are interested in locating relevant documents. So, given the estimated ranking for a query, we want to determine how many relevant documents are actually available in the n top-ranked collections. We use the estimated ranking, but substitute in the number of relevant documents found in the collections when performing the evaluation. Note that the merit

Baseline Merits

	A	B	C	D	E	F
q_1	6	2	9	5	1	7
q_2	4	18	3	9	5	1
q_3	2	1	2	0	4	0

Estimated Merits

	A	B	C	D	E	F
q_1	0.7	0.4	0.6	0.2	0.1	0.5
q_2	0.2	0.8	0.4	0.7	0.9	0.1
q_3	0.2	0.1	0.3	0.4	0.5	0.2

Baseline Rankings

	1	2	3	4	5	6
q_1	C	F	A	D	B	E
q_2	B	D	E	A	C	F
q_3	E	A	C	B	D	F

Estimated Rankings

	1	2	3	4	5	6
q_1	A	C	F	B	D	E
q_2	E	B	D	C	A	F
q_3	E	D	C	A	F	B

Merits as Used to Compute $\mathcal{R}_n$, $\widehat{\mathcal{R}}_n$ and $\mathcal{P}_n$

Ranked Collections

	q_1		q_2		q_3	
	B	**E**	**B**	**E**	**B**	**E**
1	C	A	B	E	E	E
2	F	C	D	B	A	D
3	A	F	E	D	C	C
4	D	B	A	C	B	A
5	B	D	C	A	D	F
6	E	E	F	F	F	B

Substitute in Merits

	q_1		q_2		q_3	
	B	**E**	**B**	**E**	**B**	**E**
1	9	6	18	5	4	4
2	7	9	9	18	2	0
3	6	7	5	9	2	2
4	5	2	4	3	1	2
5	2	5	3	4	0	0
6	1	1	1	1	0	1

$\mathcal{R}_n$, $\widehat{\mathcal{R}}_n$ and $\mathcal{P}_n$ results

		n					
		1	2	3	4	5	6
	$\mathcal{R}_n$	6/9	15/16	22/22	24/27	29/29	30/30
q_1	$\widehat{\mathcal{R}}_n$	6/30	15/30	22/30	24/30	29/30	30/30
	$\mathcal{P}_n$	1/1	2/2	3/3	4/4	5/5	6/6
	$\mathcal{R}_n$	5/18	23/27	32/32	35/36	39/39	40/40
q_2	$\widehat{\mathcal{R}}_n$	5/40	23/40	32/40	35/40	39/40	40/40
	$\mathcal{P}_n$	1/1	2/2	3/3	4/4	5/5	6/6
	$\mathcal{R}_n$	4/6	4/6	6/8	8/9	8/9	9/9
q_3	$\widehat{\mathcal{R}}_n$	4/9	4/9	6/9	8/9	8/9	9/9
	$\mathcal{P}_n$	1/1	1/2	2/3	3/4	3/5	4/6

Fig. 6. An example evaluation using the $\mathcal{R}_n$, $\hat{\mathcal{R}}_n$ and $\mathcal{P}_n$ evaluation measures. The collection rankings from the example in Figure 5 are evaluated.

of collection B in the estimated ranking for query q_3 is left empty because B is not selected by the estimator. Also note that the use of baseline merits in the computation of $\mathcal{R}_n$, $\hat{\mathcal{R}}_n$ and $\mathcal{P}_n$ means that, in general, these measures are not symmetric.

The lower portion of Figure 6 illustrates how the performance of the estimated rankings for q_1, q_2 and q_3 will appear when evaluated using $\mathcal{R}_n$, $\hat{\mathcal{R}}_n$ and $\mathcal{P}_n$. Again, $\mathcal{R}_n$ shows the rate at which the estimator accrues *available* relevant documents while $\hat{\mathcal{R}}_n$ shows the rate at which the total number of relevant documents are accrued. $\mathcal{P}_n$ shows the fraction of collections that have any relevant documents. Because all collections have relevant documents for queries q_1 and q_2, the $\mathcal{P}_n$ measure doesn't shed any light on performance for those queries.

4.4.2 *Spearman Coefficient of Rank Correlation.* The majority of our results are reported using the recall and precision analogs defined above. We also use the the Spearman coefficient of rank correlation [Gibbons 1976] to report some query-by-query results. The Spearman coefficient of rank correlation, ρ, is given by

$$\rho = 1 - \frac{6 \sum_{i=1}^{N} D_i^2}{N(N^2 - 1)},\tag{5}$$

where D_i is the difference in the ith paired ranks. We have $-1 \leq \rho \leq 1$ where $\rho = 1$ when two rankings are in perfect agreement and $\rho = -1$ when they are in perfect disagreement. In our work we use the midrank[7] method for assigning ranks when ties are present and we use the Spearman calculation corrected for ties.

5. COLLECTION SELECTION COMPARISONS

This section presents a comparative evaluation of three collection selection approaches, *CORI*, *gGlOSS* and *CVV*. When they were proposed, these approaches were independently evaluated; however, the evaluations were conducted using a variety of test environments. It was not possible to compare these algorithms reliably based solely on the published evaluations. Prior to our early comparative experiments [French et al. 1998, 1999b], there had been only one extremely limited comparison of these approaches. Yuwono and Lee [1997] compared the ability of *CORI*, *gGlOSS* and *CVV* to identify collections with highly similar documents using a test environment containing only four collections. This section represents an expansion of our early comparative experiments.

Here, we present a direct comparison of the *CORI*, *gGlOSS* and *CVV* approaches using the *SYM-236*, *UDC-236* and *UBC-100* testbeds and both the title and long query formulations of TREC topics 51–150. For this comparison, we are concerned only with the *collection selection* performance of these approaches. The effect that collection selection can have on data item retrieval in a multicollection environment was studied in other work [Powell et al. 2000]. Our goal for these experiments was to perform a direct comparison of the approaches in a variety of test environments to determine relative performance, the effect of test environment on performance, and whether the additional statistical information used by *gGlOSS* is beneficial.

In this section, we will first describe the *CORI*, *gGlOSS* and *CVV* collection selection approaches. We then describe the test environments and cover our experimental setup. We will present results of our direct comparison of the three approaches using the $\mathcal{R}_n$, $\hat{\mathcal{R}}_n$ and $\mathcal{P}_n$ measures defined in Section 4. We will also discuss the correlation of these approaches with collection size.

5.1 Approaches Considered

We compare the performance of three collection selection approaches in a variety of test environments. Here we provide the details of the collection selection algorithms.

---

[7]The mid-rank is simply the mean of the ranks of tied observations. For example, given four collections with merits 8, 6, 6, 3, the midranks of those four collections would be 1, 2.5, 2.5, 4, respectively.

5.1.1 *gGlOSS*.   In the implementation and evaluation of *gGlOSS*, Gravano and García-Molina [1995] assumed that all of the collections in $\mathcal{C}$ employ the same algorithms to compute term weights and similarities. Given a similarity function $sim(q, d)$ that computes the similarity between a query q and document d in a collection, Gravano and García-Molina defined a notion of *goodness* for each collection. For similarity threshold l, *goodness* is defined as the sum of all document similarities in the collection where $sim(q, d) > l$. The desired behavior of *gGlOSS* is to rank collections in decreasing order of goodness. Having established the desired behavior, Gravano and García-Molina then defined two *estimators* that estimate goodness using two assumptions of query term co-occurrence. The *Max(l)* estimator assumes the highest possible level of co-occurrence of query terms in documents while the *Sum(l)* estimator assumes that two terms appearing in the query do not appear together in a collection document. *gGlOSS* requires two vectors of information from each collection C_i in order to make its *Max(l)* and *Sum(l)* estimates,

(1) the document frequency df_{ij} for each term t_j in C_i; and
(2) the sum of the weight of each term t_j over all documents in C_i.

The vectors from each collection are stored in two matrices, *F* and *W* where the *F* matrix stores the document frequency vectors and the *W* matrix stores the term weight vectors. For both estimators, it is assumed that the weight of a term is distributed uniformly over all documents that contain that term. *gGlOSS* uses the assumptions underlying *Max(l)* (or *Sum(l)*) to estimate the number of documents in a collection C having similarity to a query greater than a threshold l. This forms the basis for the *gGlOSS* estimate of the goodness of C. Gravano and García-Molina used the $\mathcal{R}_n$ and $\mathcal{P}_n$ evaluation measures (which we discussed in Section 4) to evaluate the degree to which the *Max(l)* and *Sum(l)* estimators could rank collections in decreasing order of goodness. They found that both estimators perform well with respect to that evaluation criterion.

In early experiments, we confirmed that the *gGlOSS* estimators approximate the *gGlOSS* baselines very well [French et al. 1998]. We showed that both the estimators and baselines have very similar performance when used to approximate a baseline based on the number of relevant documents in each collection. For this work, we will use the *Ideal(0)* baseline to represent *gGlOSS* in comparisons with other collection selection techniques. Here, we motivate that decision. There were two other options, (1) using *Max(l)* or *Sum(l)*, or (2) using *Ideal(l)* for some $l > 0$. We cover the two cases separately.

An immediate question is why we chose a *gGlOSS* baseline instead of one of the estimators as the representative. After all, Gravano and García-Molina [1995] proposed the *Max(l)* and *Sum(l)* estimators for actual use. However, from the definitions of the *gGlOSS* baselines and estimators, *Max(0) = Sum(0) = Ideal(0)*, that is, at threshold $l = 0$ both estimators and the *Ideal(0)* baseline give identical rankings of the collections for all queries.

Given this equivalence, using the *Ideal(0)* formulation is simpler from an implementation standpoint. *gGlOSS* requires two vectors of information from each collection C_i in order to make its *Max(l)* and *Sum(l)* estimates, the *F* and

W matrices defined above. If the underlying collection cannot be made to divulge this information directly, it is in principle still possible to compute the estimates. However, the two vectors of information must be recovered in some way, possibly by issuing a single-term query for each vocabulary term. Our choice of *Ideal(0)* obviates this; if the information required to compute *Ideal(0)* is not readily available, we can compute *Ideal(0)* directly from the collections by simply issuing the test queries. Finally, when used to approximate a baseline based on the number of relevant documents in each collection, *Ideal(0)* consistently achieves good performance relative to the *Max(l)* and *Sum(l)* estimators [French et al. 1998; Powell 2001]. Given the general performance equivalence and the operational advantages of *Ideal(0)*, we chose *Ideal(0)* over *Max(l)* or *Sum(l)* as the *gGlOSS* representative.

A second question is whether an alternate choice of l for *Ideal(l)* would be more appropriate than *Ideal(0)*. For some experimental environments, this may very well be the case; however, the choice of $l > 0$ is problematic for two reasons. A first difficulty is related to scales of similarity values. Choosing a value of l that is too high will incorrectly estimate zero merit for many collections, seriously degrading average performance. Using *Ideal(0)* avoids this difficulty. A second difficulty is that as *gGlOSS* is defined, the value of l is a constant across all collections. When all collections use the same indexing scheme with the same ranges of possible similarity values, the only difficulty is the one noted above—choosing an appropriate value of l. However, in an operational environment it may be the case that the underlying collections use different information retrieval systems that produce differently scaled similarity values. This makes ranking collections based on these goodness values difficult.

When $l = 0$ is used as a threshold, all documents with non-zero similarity to the query contribute to the goodness of the collection. This allows a consistent comparison of collections with different underlying retrieval systems. Note that while the comparison is consistent, it may still not be straightforward. Generally speaking, sums of values from 1 to 100 will grow much faster than sums of 1 to 10.

For these reasons, $l = 0$ represents the simplest and generally most reliable choice for a threshold. We use *Ideal(0)* to represent *gGlOSS*. Here, we summarize the *Ideal(0)* baseline that we use as a representative for *gGlOSS* in all remaining experiments.

From the original definition of *Ideal(l)* [Gravano and García-Molina 1995], *Ideal(0)* can be computed by sorting collections in decreasing order of Goodness when $l = 0$.

$$Goodness(0, q, C_i) = \sum_{\{d \in C_i | sim(q,d) > 0\}} sim(q, d).$$

Because *Max(0) = Sum(0) = Ideal(0)*, *Ideal(0)* can also be computed using only the W matrix required by *gGlOSS*, using the following computation,

$$Estimate(0, q, C_i) = \sum_{j=1}^{n} q_j \times w_{ij}.$$

5.1.2 *CORI*.   Given a set of collections to search, the *CORI* [Callan et al. 1995] collection selection approach creates a *collection selection index* in which each collection is represented by its terms and their document frequencies *df*. Collections are ranked for a query q by a variant of the Inquery document ranking algorithm. The belief $p(t_j|C_i)$ in collection C_i due to observing query term t_j is determined by:

$$T = \frac{df}{df + 50 + 150 \cdot cw/\overline{cw}}$$

$$I = \frac{\log\left(\frac{N+0.5}{cf}\right)}{\log\left(N + 1.0\right)}$$

$$p(t_j|C_i) = 0.4 + 0.6 \cdot T \cdot I, \tag{6}$$

where:
  df   is the number of documents in C_i containing t_j,
  cf   is the number of collections containing t_j,
  N   is the number of collections being ranked,
  cw   is the number of words in C_i, and
  $\overline{cw}$ is the mean *cw* of the collections being ranked.

In the general case, the belief in a collection depends upon the query structure; for our experiments it is the average of the $p(t_j|C_i)$ values for each query term [Callan et al. 1995].

The *CORI* approach to ranking collections can be summarized as $df \cdot icf$, where *icf* is inverse collection frequency. Given a set of collections to search, the *CORI collection selection index* essentially represents each collection as a virtual document made up of a list of terms and their document frequencies in the underlying collections. The virtual documents are indexed by the Inquery information retrieval system [Callan et al. 1992]. A query q is applied to the collection selection index to rank the virtual documents. The resulting virtual document ranking is the *CORI* ranking of the collections.

5.1.3 *CVV*.   Yuwono and Lee [1997] proposed an approach to the broader problem of distributed search, considering collection selection, query forwarding and results merging. They referred to the collection selection portion of their work as the *Cue Validity Variance (CVV)* ranking method. *CVV* refers both to the ranking method and to a component in their calculation of a collection's estimated merit or score.

The *CVV* ranking method employs a combination of document frequency (*df*) information and cue validity variance. Cue validity variance, as defined by Yuwono and Lee, attempts to characterize the distribution of the density of *df* values, that is, the variability of the fraction of documents in a collection that contain a given term. Document frequency information is used to approximate how important a term is within a collection; the goal of the *CVV* component is to estimate whether a term is useful for differentiating one collection from another.

The *CVV* estimated merit computation is summarized below. Note that there are some notational differences between our summary and the version

presented by Yuwono and Lee. We have modified the notation to be conformant with the notation used in the rest of this article.

$$est\_merit(C_i, q) = \sum_{\{t_j \in q\}} CVV_j \cdot df_{ij}, \tag{7}$$

where t_j is a term in query q, df_{ij} is the document frequency of t_j in collection C_i, and

$$CVV_j = \frac{\sum_{i=1}^{N}(CV_{ij} - \overline{CV_j})^2}{N}$$

where

$$CV_{ij} = \frac{\frac{df_{ij}}{|C_i|}}{\frac{df_{ij}}{|C_i|} + \frac{\sum_{k \neq i}^{N} df_{kj}}{\sum_{k \neq i}^{N} |C_k|}}$$

and

$$\overline{CV_j} = \frac{\sum_{i=1}^{N} CV_{ij}}{N}.$$

The *CVV* ranking method uses only information from (or derivable from) the matrix F used by *gGlOSS*. The goal of the *CVV* merit estimation method is to identify collections with a high concentration of query terms.

5.2 Relevance-Based Ranking as Baseline

We have chosen to focus on the degree to which collection selection approaches can locate collections containing relevant documents. Therefore, we use a relevance-based ranking (RBR) as the primary baseline for our evaluations. RBR is constructed by ordering the collections in decreasing order of the number of relevant records contained in the collection. RBR was used by Callan et al. [1995] for their evaluation of *CORI*.

There are other possible baseline rankings. For example, as we discussed in Section 5.1.1, *Ideal(l)* was used as a baseline by Gravano and García-Molina [1995] to report their performance evaluation of *gGlOSS*. Our choice of RBR as the desired collection selection performance could be considered controversial, especially in the evaluation of *gGlOSS*. *gGlOSS* was designed to characterize the similarities of documents in a collection to a query and was not originally evaluated using relevance. We chose to study the degree to which collection selection approaches could identify collections containing relevant documents. We included the *gGlOSS Ideal(0)* baseline so that we could consider an approach that focused on document similarity, and to determine if the additional collection information utilized by the *gGlOSS* approach would provide an improved ability to locate relevant documents. Even though we are using *gGlOSS* in a slightly different context than that for which it was designed, we feel that its inclusion in the study is appropriate and instructive. While we use the RBR baseline, the overall methodology described here is compatible with other choices of baseline.

5.3 Test Environments and Experimental Setup

We used six test environments for the experiments reported in this section. As we mentioned earlier, we used the *SYM-236*, *UDC-236* and *UBC-100* testbeds and both the long and short query formulations of TREC topics 51–150. The six test environments can be specified as:

—(SYM-*236*, $\mathcal{Q}_l$, $\mathcal{J}_{TREC4}$), (SYM-*236*, $\mathcal{Q}_s$, $\mathcal{J}_{TREC4}$),

—(UDC-*236*, $\mathcal{Q}_l$, $\mathcal{J}_{TREC4}$), (UDC-*236*, $\mathcal{Q}_s$, $\mathcal{J}_{TREC4}$),

—(UBC-*100*, $\mathcal{Q}_l$, $\mathcal{J}_{TREC4}$) and (UBC-*100*, $\mathcal{Q}_s$, $\mathcal{J}_{TREC4}$).

Note that the relevance judgments for all six test collections are those provided with the TREC-4 data. When describing the results, we will refer to the test environments using the testbed and query set names. We found that collection selection results vary more on a testbed-by-testbed basis than on a query set basis. Therefore, we present results by testbed, with long and short query results displayed side-by-side for easier comparison.

We prepared each collection in each testbed using version 11.0 of the SMART information retrieval system [Buckley 1992]. Statistical information extracted from the SMART indexes is used to create the *F* and *W* matrices that are the input to *gGlOSS*. The document frequency (*df*) information required by *CORI* and *CVV* was taken from the *F* matrix used by *gGlOSS*'s *Ideal(0)*. This provides a consistent indexing vocabulary for all competing methods and eliminates potential differences due to lexical processing steps such as stopping and stemming.[8] To facilitate the use of external *df* information, we use our own implementation of the published *CORI* algorithm. The versions of *Ideal(0)* and *CVV* that we use are also our own implementations of the published algorithms.

Queries were processed using SMART to convert them into term lists compatible with the vocabularies used by the collection selection approaches. However, SMART was not used for the implementations of the collection selection approaches or for actually issuing the queries for collection selection.

5.4 Correlation with a Size-Based Ranking

Before we present the results of our comparative experiments, we need to discuss a feature that the three collection selection approaches share to varying degrees-a tendency to select collections with large numbers of documents.

As background, our interest in the tendency of collection selection approaches to select large collections started with a detailed analysis of the *Ideal(0)* results reported in French et al. [1998, 1999b]. We were attempting to isolate the cause of the observed performance difference between *Ideal(0)* and *CORI*.[9] A detailed examination of the collection rankings revealed that many of the same collections were highly ranked by *Ideal(0)* for a startling number of the queries. Further examination revealed that the highly-ranked collections tended to be the collections with the largest number of documents.

---

[8]More detail on this aspect of our research can be found elsewhere [Powell 2001].

[9]Recall that these experiments used the very long query formulations.

We created a new size-based collection ranking (SBR) to provide a means to examine the tendency of *Ideal(0)* and other algorithms to prefer larger collections. SBR is defined by ordering collections in descending order of the total number of documents they contain. Note that this ranking is constant for all queries.

SBR is a special case ranking in our experiments because we use it both as an estimator and as a baseline. As an estimator, SBR is a simple heuristic that can also serve as a useful lower bound on performance (an effective collection selection approach should be able to perform at least as well as SBR). Over the course of our experiments we found that SBR is also useful as a baseline when studying the tendency of collection selection algorithms to select large collections.

We use Spearman's ρ as defined in Section 4 to measure the correlation between the SBR rankings and the rankings produced by RBR and the three collection selection approaches. Figure 7 presents the Spearman correlation coefficient values for the short and long queries for the *SYM-236* and *UBC-100* testbeds.[10] Each graph of Figure 7 is a scatterplot of ρ values for a testbed, query set pair. The values of ρ are computed for each of the 100 short queries and each of the 100 long queries and presented as a scatterplot. Each point represents a query under the labeled approach.

First, consider the Spearman values for the *SYM-236* testbed, the testbed for which a correlation between *Ideal(0)* and SBR was originally suspected. The correlations between *Ideal(0)* and SBR and between *CVV* and SBR are very strong for both long and short queries. The correlation between *CORI* and SBR is still positive but not as dramatic. For later reference, note that correlation between RBR and SBR is positive, suggesting that for many queries, collections containing a large number of documents also tend to contain relevant documents. The correlations between all of the approaches and SBR for the *UBC-100* testbed are not as pronounced, but are still positive on the whole.

We note that a preference for large collections is not necessarily a liability. If large collections tend to have high merit, selection based on this heuristic and selection approaches correlated with collection size can be effective. However, SBR and approaches highly correlated with it will generally perform poorly if the largest collections have little or no merit with respect to the queries.

5.5 Results

This section presents the comparison of the *CORI*, *CVV* and *gGlOSS* collection selection approaches in the test environments described above. We consider the results on a testbed by testbed basis, using the $\hat{\mathcal{R}}_n$, $\mathcal{R}_n$ and $\mathcal{P}_n$ measures together in an attempt to present a comprehensive view of performance. We will discuss the results for each testbed individually, drawing out features of

---

[10]Because the *UDC-236* testbed was constructed to contain collections with roughly the same number of documents, SBR is not applicable for this testbed. In fact, SBR essentially devolves to alphabetical order (our tiebreaker) and comparisons with it are not illuminating. There are no correlations with SBR for Figure 7 and no SBR.RBR plot presented for any of the later *UDC-236* testbed results.
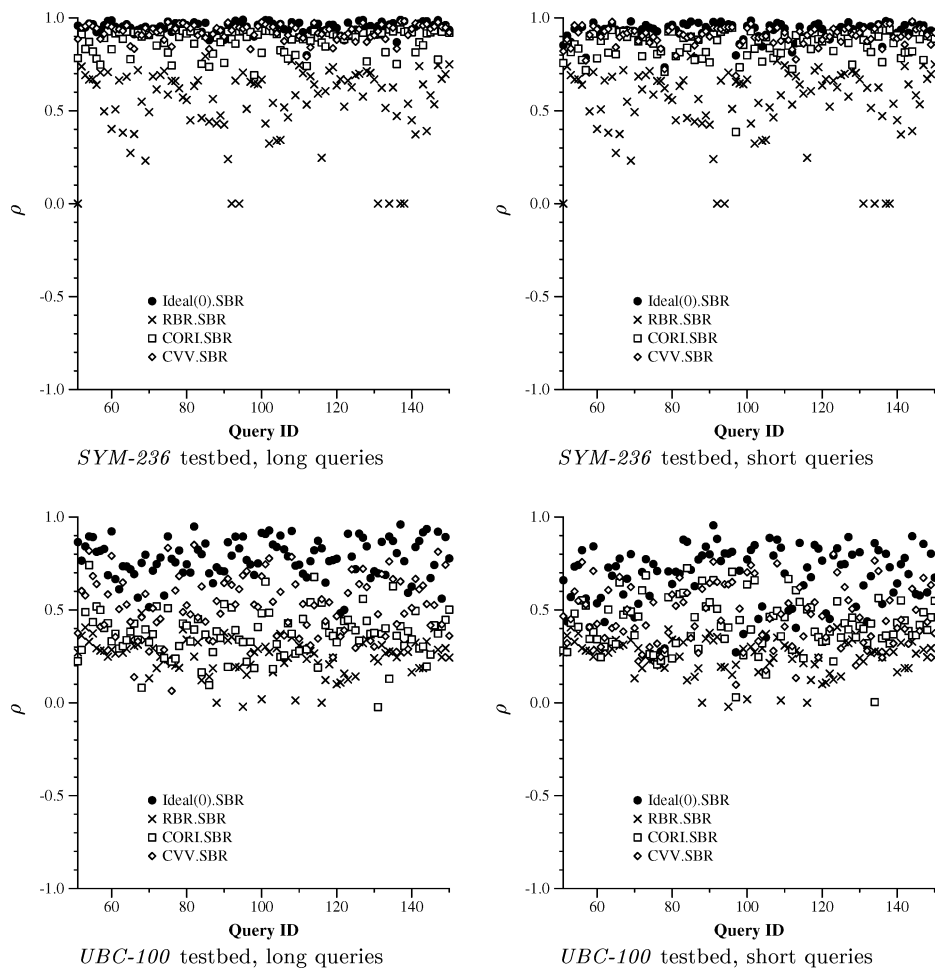
Fig. 7.   Spearman correlation of selection approaches with SBR baseline.

the testbeds that affect performance. In Section 5.6, we will present a more unified discussion of the overall results.

5.5.1 SYM-236.   The first two test environments we consider utilize the *SYM-236* testbed and both short and long queries. These test environments can be specified as (SYM-236, $\mathcal{Q}_l$, $\mathcal{J}_{TREC4}$) and (SYM-236, $\mathcal{Q}_s$, $\mathcal{J}_{TREC4}$). The collection selection evaluation results for these environments are presented in Figure 8. Figure 8 contains six graphs and presents results from two test environments using three different evaluation measures. Each graph is labeled with the testbed, evaluation measure and query set it represents. The results for test environments utilizing the *UDC-236* and *UBC-100* testbeds will follow the same organizational approach.

The first thing to note is that the *SYM-236* testbed is the same testbed used for some of our earlier experiments [French et al. 1998, 1999b]. The difference
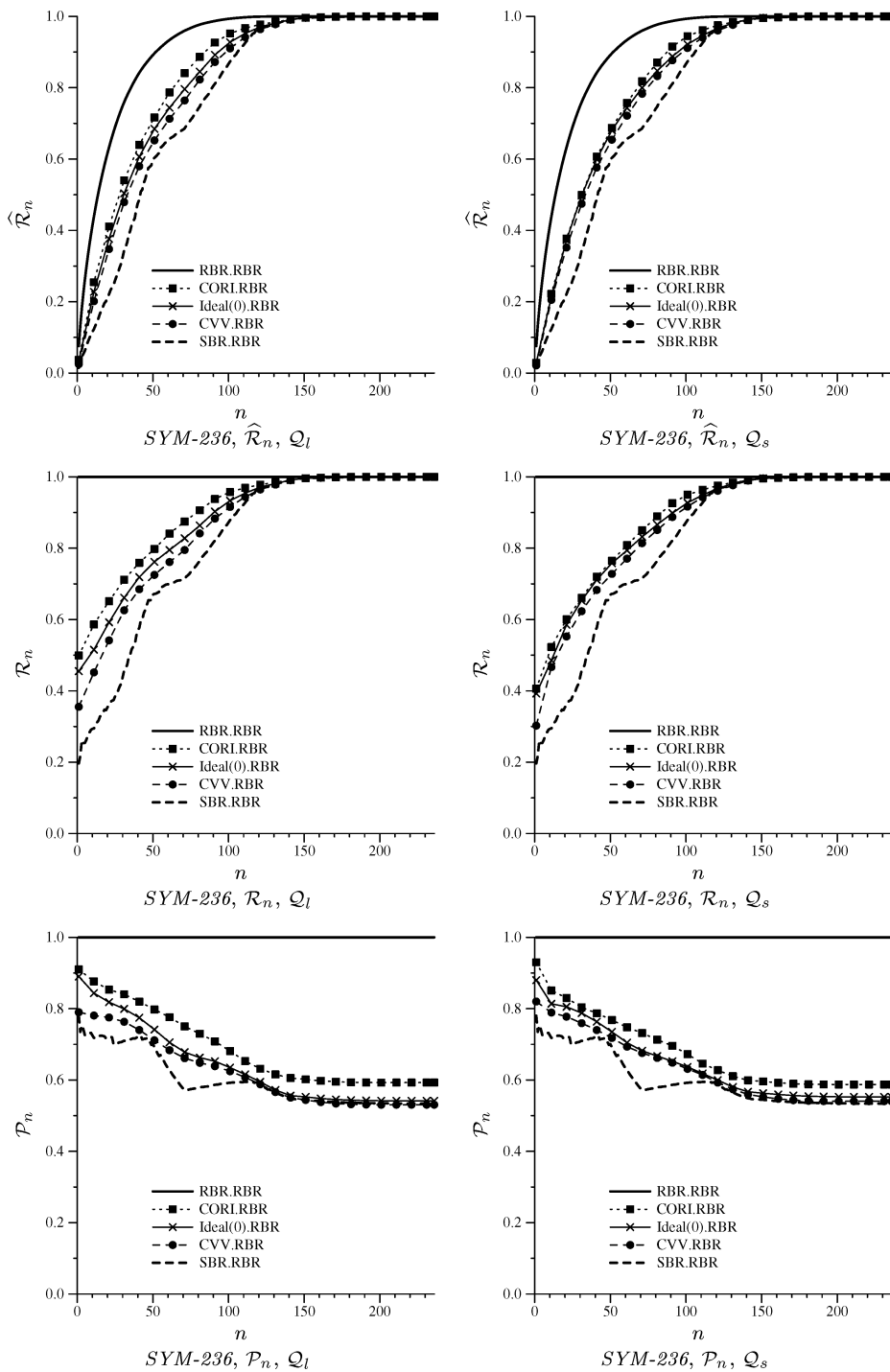
Fig. 8.   Collection selection results for the *SYM-236* testbed, long and short queries, $\mathcal{R}_n$, $\hat{\mathcal{R}}_n$ and $\mathcal{P}_n$ evaluation measures.

between these graphs and the graphs reported in earlier work is that different query formulations were used. While the query formulations are different, the distribution of relevant documents is the same. As we discussed in Section 3, the *SYM-236* testbed contains a number of collections that have no merit for most queries. The "knee" in the RBR.RBR plots on the $\hat{\mathcal{R}}_n$ graphs represents the transition from high- and medium-merit collections to very low-merit collections. The plateaus in the plots for *CORI*, *CVV*, *Ideal(0)* and SBR at approximately $n = 140$ show that all of these approaches do a similarly good job at placing the very low-merit collections at the bottom of their rankings.

The steep climb of the RBR.RBR curve under the $\hat{\mathcal{R}}_n$ evaluation measure reveals that for most queries, relevant documents are not evenly distributed across the collections in the *SYM-236* testbed. In other words, for many queries, there are a few collections with a large number of relevant documents as well as collections with no relevant documents. Any approach that is able to correctly identify the collections with a large number of relevant documents will perform very well under the $\hat{\mathcal{R}}_n$ and $\mathcal{R}_n$ measures. Conversely, collection selection approaches that do not identify high-merit collections will perform poorly. One of the hallmarks of the *SYM-236* testbed is that small perturbations in ranks can have larger effects under our evaluation measures.

The performance of *CORI*, *CVV* and *Ideal(0)* is best revealed by examining the $\mathcal{R}_n$ and $\mathcal{P}_n$ graphs together. Recall that $\mathcal{R}_n$ measures the rate at which approaches have accrued *available* merit, while $\mathcal{P}_n$ measures the degree to which zero merit collections have been interleaved in the ranking. Considered together, the $\mathcal{R}_n$ and $\mathcal{P}_n$ graphs reveal that, especially for small values of n, while the approaches tend to identify collections with some relevant documents, they do not identify the collections with the *most* relevant documents. For example, consider the "*SYM-236*, $\mathcal{R}_n$, $\mathcal{Q}_l$" and "*SYM-236*, $\mathcal{P}_n$, $\mathcal{Q}_l$" graphs of Figure 8. When 10 collections are selected by *CORI* ($n = 10$), 88% of those collections contain relevant documents, but only 58% of the available relevant documents have been accrued.

Comparisons between the performance of the approaches using the two different query formulations are facilitated by the RBR.RBR and SBR.RBR curves. These curves are not affected by the query formulation and are the same for each pair of $\hat{\mathcal{R}}_n$, $\mathcal{R}_n$ and $\mathcal{P}_n$ graphs. Overall, we note that the relative performance of the three approaches is similar under both the long and short query formulations, but that the overall performance of the approaches is slightly worse for the short queries. The differences among the approaches are smaller under the short queries; however, on average, *CORI* appears to be affected more by the short query effect.

Overall, the performance curves for *CORI*, *Ideal(0)* and *CVV* appear to be similar under all three evaluation measures. The tendency of the plots to follow a similar path can obscure the difference between the curves; the differences between the curves need to be observed vertically. The vertical differences between the points are greater than they appear when only the track of the curves is considered. For example, consider the *SYM-236*, $\mathcal{R}_n$, long queries graph of Figure 8. Averaged over the values $1 \leq n \leq 50$, we find that *CORI* performs 10% better than *Ideal(0)* and 20% better than *CVV*.
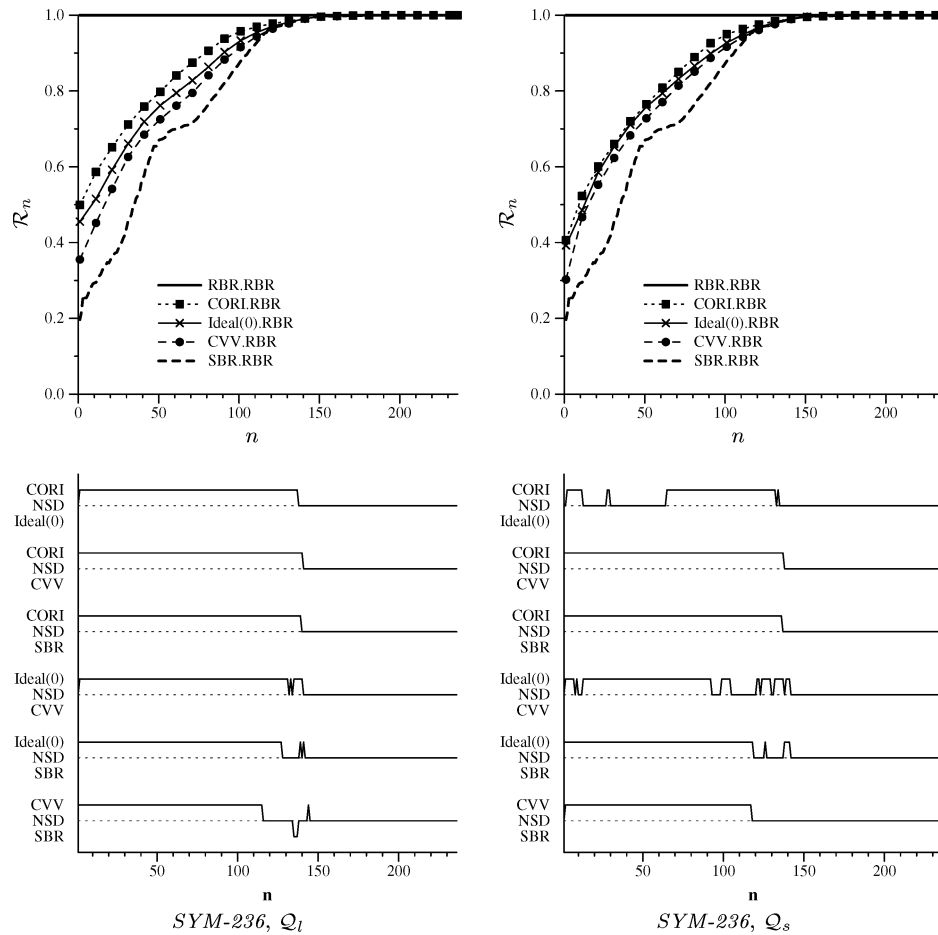
Fig. 9. Comparison of approaches using the $\mathcal{R}_n$ measure, *SYM-236* testbed, long and short queries. The lower graphs show the results of paired Wilcoxon (p = 0.05) significance tests conducted between pairs of methods for all values of n (number of collections selected). NSD means no significant difference and the fine dotted lines plotted at the NSD points are provided for reference.

Figure 9 presents a pairwise comparison of approaches using a paired Wilcoxon (p = 0.05) significance test for the evaluation results under the $\mathcal{R}_n$ measure. The two $\mathcal{R}_n$ graphs from Figure 8 are repeated here, aligned with graphs that show pairwise significance comparisons of the plots for all values of n. For each comparison, the baseline is no significant difference (NSD); the plot shifts to one of the approaches when that approach is significantly better than the approach with which it is paired. Figure 9 reveals that the pairwise differences between the approaches are significant for most values of n less than approximately 140. For the *SYM-236* testbed, all approaches are equally effective at placing the very low-merit collections (which also contain a very small number of documents) at the bottom of the rankings. As a result, there is little visible (or significant) difference between the approaches for n greater than approximately 140. We also note that the drop in the performance of *CORI* for the

short queries results in no significant difference between *CORI* and *Ideal(0)* for some of the lower values of n.

One additional thing to note about Figure 8 is the slight shift in SBR performance for $50 < n < 70$. This is due to the presence of a consecutively-ranked group of ZIFF collections. Note, in Figure 2, that the ZIFF collections contain a fairly large number of documents. However, there are relatively few queries for which these documents are relevant. These large, but often nonrelevant collections adversely affect the performance of SBR.

5.5.2   UDC-236.   The next pair of test environments that we consider utilize the *UDC-236* testbed and both long and short queries. These test environments can be specified as $(UDC\text{-}236, \mathcal{Q}_l, \mathcal{J}_{TREC4})$ and $(UDC\text{-}236, \mathcal{Q}_s, \mathcal{J}_{TREC4})$. Results are presented in Figure 10.

Recall that the *UDC-236* testbed uses the same document set $\mathcal{D}$ and the same number of collections N as the *SYM-236* testbed, the documents are merely organized into collections differently. *UDC-236* was designed so that the number of documents per collection is roughly uniform. A related feature, although a result that was not explicitly designed into the testbed, is that the number of relevant documents per collection is more uniform for *UDC-236* than for *SYM-236*. As a result, in Figure 10 we don't see the pronounced plateaus in the performance curves for *CORI*, *CVV* and *Ideal(0)* that we saw for the *SYM-236* testbed. However, the shape of the RBR.RBR curve under the $\hat{\mathcal{R}}_n$ evaluation measure reveals that for most queries there are collections for which some query has no relevant documents. The RBR.RBR curve is not as steep as that seen in the *SYM-236* testbed, however it is still much steeper than the curves for the three collection selection approaches. Similar to what we observed for *SYM-236*, *UDC-236* contains collections with large numbers of relevant documents with respect to the queries. The differences in relevant documents per collection simply are not as pronounced as they were in the *SYM-236* testbed (see Figures 2 and 3).

The $\mathcal{R}_n$ and $\mathcal{P}_n$ graphs of Figure 10 reveal comparative performance of *CORI*, *CVV* and *Ideal(0)* that is similar to that we saw for the *SYM-236* testbed. All three approaches tend to identify collections with some relevant documents but not the collections containing the highest numbers of relevant documents. Comparisons between the performance of the approaches using the two different query formulations are more difficult because of the lack of an SBR.RBR curve. The difference between the performance of short and long queries is most obvious for the $\mathcal{R}_n$ measure. Overall, we note that the relative performance of the three approaches is similar under both the long and short query formulations, but that the overall performance is slightly worse for the short queries. The differences among the approaches are smaller under the short queries.

Figure 11 presents a pairwise comparison of the approaches using a paired Wilcoxon (p=0.05) significance test for the evaluation results under the $\mathcal{R}_n$ measure. For most values of n, the differences between the three approaches are significant for the long queries. For the short queries, there is no significant difference more often, most notably between *CORI* and *Ideal(0)* for very small values of n. However, for most comparisons and most values of n, the pairwise
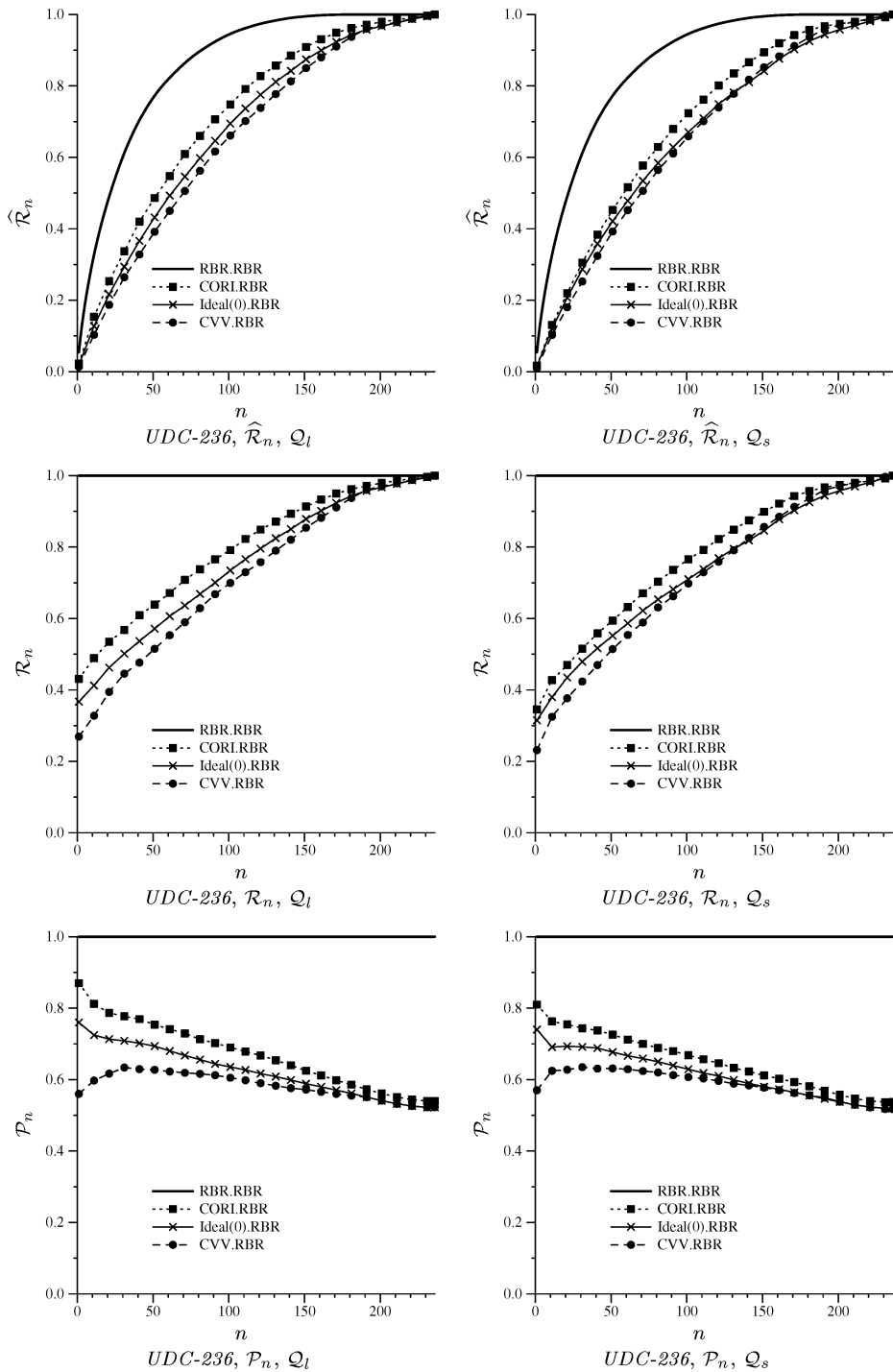
Fig. 10.   Collection selection results for the *UDC-236* testbed, long and short queries, $\mathcal{R}_n$, $\hat{\mathcal{R}}_n$ and $\mathcal{P}_n$ evaluation measures.
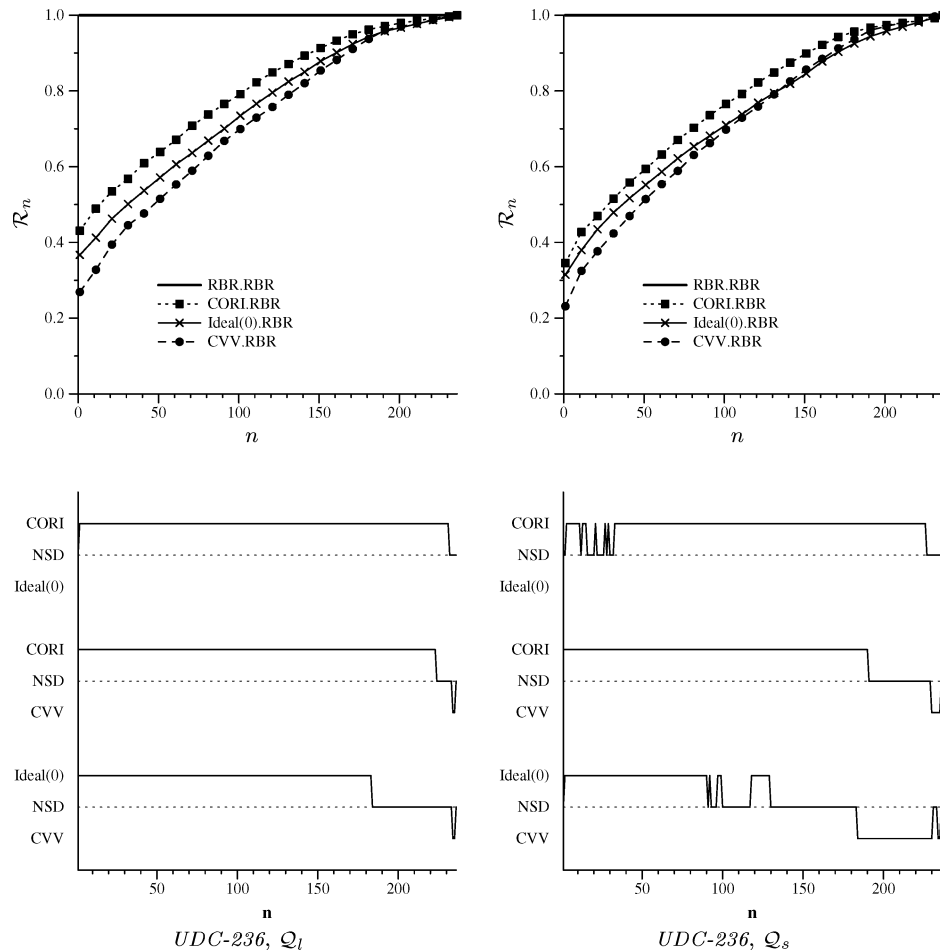
Fig. 11. Comparison of approaches using the $\mathcal{R}_n$ measure, *UDC-236* testbed, long and short queries. The lower graphs show the results of paired Wilcoxon (p = 0.05) significance tests conducted between pairs of methods for all values of n (number of collections selected). NSD means no significant difference and the fine dotted lines plotted at the NSD points are provided for reference.

differences are significant. It is interesting to note that the *CVV* and *Ideal(0)* performance curves for short queries cross over at approximately $n = 140$ and *CVV* becomes significantly better at approximately $n = 180$. However, this is unlikely to have a large effect due to the very high value of n.

One final thing to note about Figure 10 is that *UDC-236* is a more difficult testbed than *SYM-236*. As we noted earlier, the relevant documents are more evenly distributed. While the performance curves for all three approaches are similar to those seen for *SYM-236* the overall values under the performance measures are lower.

5.5.3 UBC-100. The third pair of test environments that we consider utilize the *UBC-100* testbed and both long and short queries. These test environments can be specified as (*UBC-100*, $\mathcal{Q}_l$, $\mathcal{J}_{TREC4}$) and (*UBC-100*, $\mathcal{Q}_s$, $\mathcal{J}_{TREC4}$).

The results for these test environments using the $\hat{\mathcal{R}}_n$, $\mathcal{R}_n$ and $\mathcal{P}_n$ evaluation measures are presented in Figure 12.

When examining the results in Figure 12, recall that there are a few differences between the *UBC-100* testbed and the *SYM-236* and *UDC-236* testbeds. The *UBC-100* testbed contains more documents spread over fewer collections. Despite these differences, the shape of the RBR.RBR curve under the $\hat{\mathcal{R}}_n$ measure is similar to that seen for the *UDC-236* testbed.

The most striking thing about the results of Figure 12 is the performance of SBR and *Ideal(0)*. The performance of SBR is much poorer than that seen for the *SYM-236* testbed—while SBR tracked the other approaches closely in *SYM-236*, the performance in *UBC-100* is obviously much poorer. The $\mathcal{P}_n$ graphs provide a first clue as to the cause of the poor performance. The $\mathcal{P}_n$ graphs reveal that for very small values of n, SBR chooses collections with zero merit approximately 80% of the time. The source of this difficulty is illustrated by Figure 4. Six DOE and two ZIFF collections are far larger than the other collections but contain relevant documents for few queries.[11]

While *Ideal(0)* is somewhat less correlated with SBR in this testbed than in *SYM-236* (see Figure 7), that correlation is enough to adversely affect performance. For the *UBC-100* testbed, *Ideal(0)* performs significantly more poorly than *CVV* whereas in *SYM-236* and *UDC-236*, *Ideal(0)* performs better than *CVV*. Under the evaluation measures that we use, very poor early performance can have ramifications for values of n other than those for which poor selection occurred. The *UBC-100* testbed reveals the potential downfall of collection selection approaches that prefer collections with a large number of documents. The presence of large collections with few relevant documents will often confound these approaches.

It is interesting to note that with the exception of *Ideal(0)* and SBR, the overall performance trends presented in Figure 12 are similar to those seen for the *SYM-236* and *UDC-236* testbeds. The pairwise Wilcoxon (p=0.05) significance tests presented in Figure 13 reveal that for most values of n, the differences between the three approaches are significant for the long and short queries.

5.6 Discussion

In this section, we have presented a comparative collection selection evaluation for *CORI*, *CVV* and *gGlOSS Ideal(0)* using six different test environments. We have also compared the performance of those collection selection approaches to that of SBR.

Despite the variety of experimental environments, the overall results are fairly consistent. While there is significant difference between *CORI*, *Ideal(0)*, *CVV* and SBR, they tend to perform similarly when compared to the performance of the baseline against itself (RBR.RBR). Some of the observed performance of the approaches is explained by features of the testbeds. The most

---

[11]Recall that ZIFF collections as a whole proved problematic for SBR within the *SYM-236* testbed. For *UBC-100*, the other ZIFF collections are sufficiently scattered through the SBR ranking that they have no visible effect.
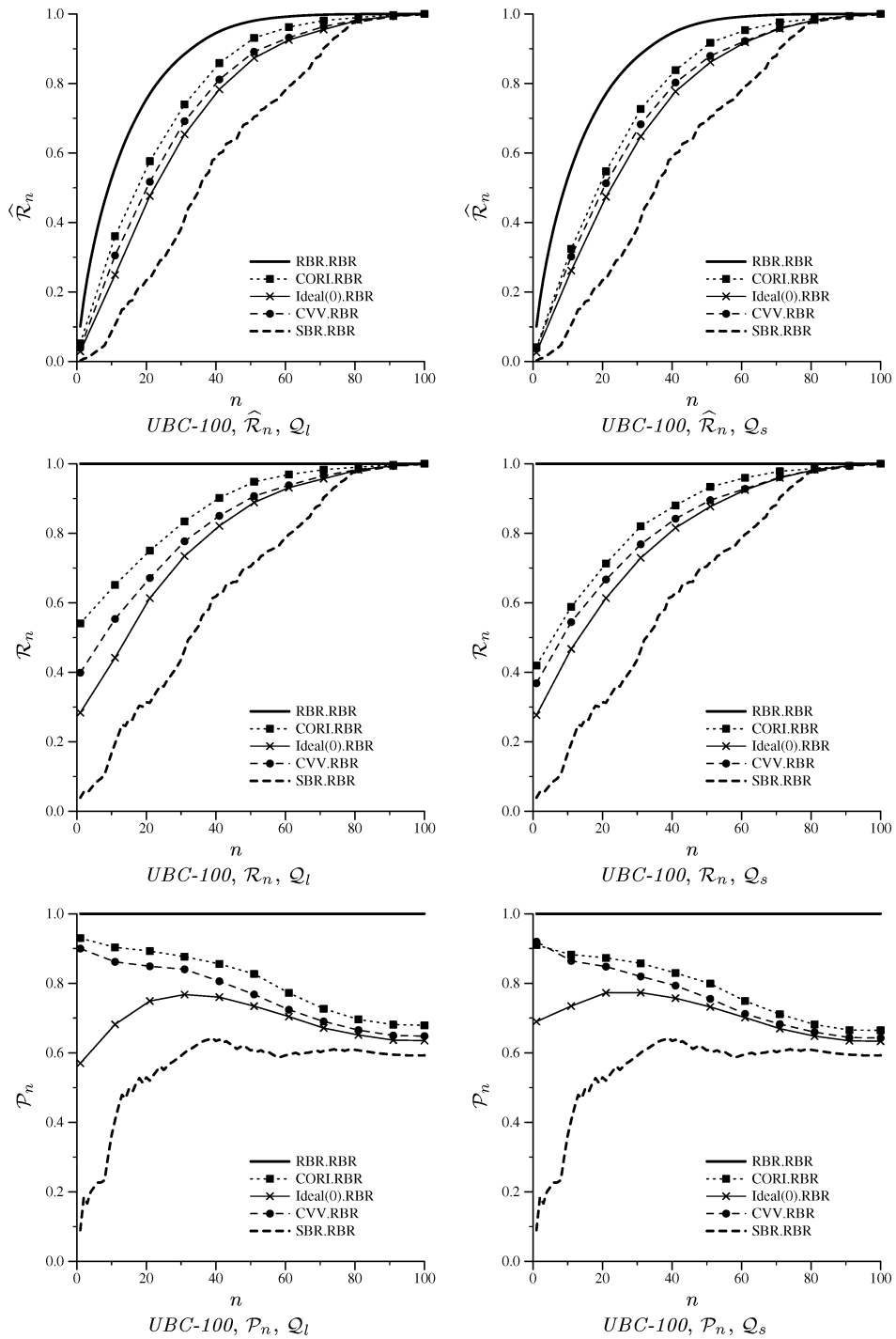
Fig. 12.   Collection selection results for the *UBC-100* testbed, long and short queries, $\mathcal{R}_n$, $\hat{\mathcal{R}}_n$ and $\mathcal{P}_n$ evaluation measures.
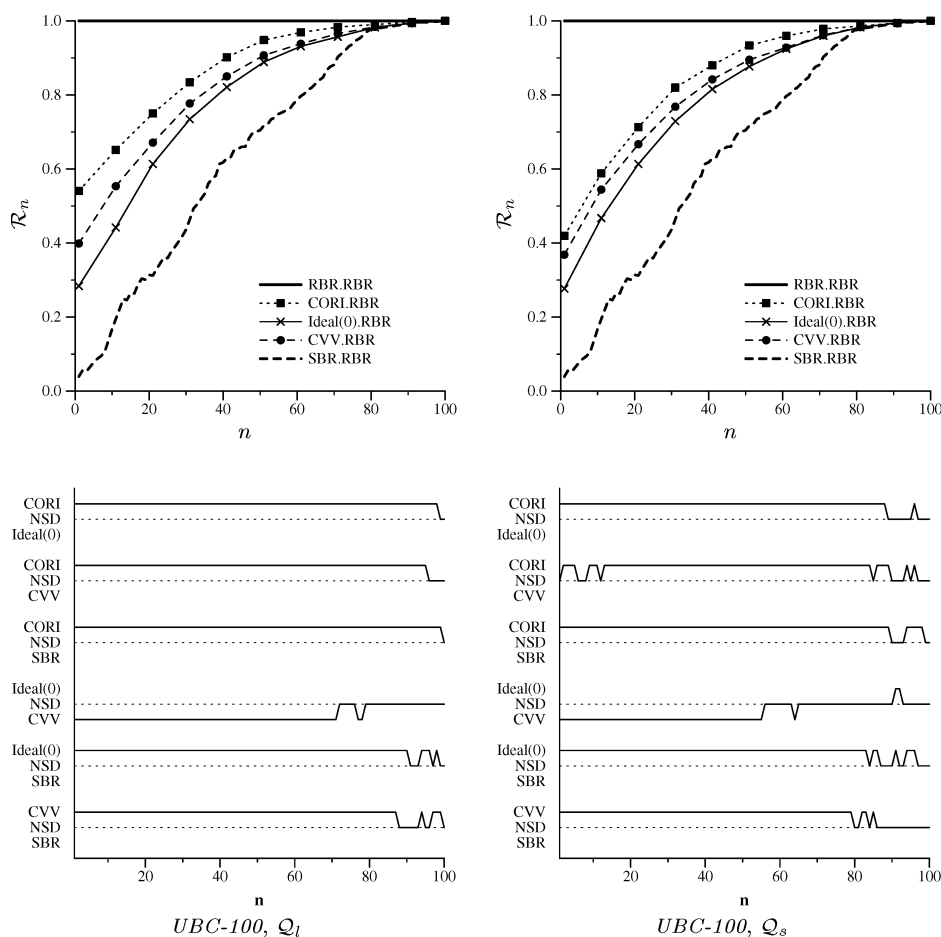
Fig. 13. Comparison of approaches using the $\mathcal{R}_n$ measure, *UBC-100* testbed, long and short queries. The lower graphs show the results of paired Wilcoxon (p = 0.05) significance tests conducted between pairs of methods for all values of n (number of collections selected). NSD means no significant difference and the fine dotted lines plotted at the NSD points are provided for reference.

dramatic differences were seen in the performance of *gGlOSS Ideal(0)* and SBR in the *UBC-100* testbed.

    SYM-236 exhibits a skewed distribution of documents and relevant documents. We found that for *SYM-236* all approaches performed comparably in their ability to differentiate collections containing relevant documents from those containing no relevant documents. However, while all approaches tended to locate collections containing relevant documents, they tended not to find collections containing the most relevant documents. For all approaches, we also noted a tendency to locate collections containing relevant documents but not the greatest number of relevant documents in the *UDC-236* testbed. *UDC-236* proved to be a more challenging testbed than *SYM-236*. While it is still the case that *UDC-236* contains collections with large numbers of relevant documents,

the distribution of relevant documents is more uniform in *UDC-236*. All approaches performed less well at differentiating collections containing relevant documents from those containing no relevant documents. *UBC-100* proved to be an interesting testbed because it illustrates a potential drawback to using the SBR heuristic. *UBC-100* contains six collections with a very large number of documents, but relatively few relevant documents. The performance of SBR and *Ideal(0)* (which is very highly correlated with SBR) was adversely affected by a tendency to select those very large collections. The performance of *CORI* and *CVV* in the *UBC-100* testbed was similar to that seen in *SYM-236* and *UDC-236*.

When compared to the performance of RBR.RBR, we see that there is substantial room for improvement for all of the collection selection approaches in all of our testbeds. Overall, the *CORI* collection selection approach performs most accurately and most consistently. For all six experimental environments, *CORI* is significantly better than (or not significantly different from) the other approaches. In work not reported here [French et al. 1999a; Powell 2001], we examine the influence of the components of $df \cdot icf$-based approaches, of which *CORI* is an instance. We found that on the whole, $df \cdot icf$ collection selection approaches tend to perform well.

The *gGlOSS* representative *Ideal(0)* performed adequately but its correlation with SBR proved problematic for the *UBC-100* testbed. When comparing *CORI* and *Ideal(0)* in a broader scope, the main advantage of *CORI* is that it requires less statistical information about the collections. The *df* information that *CORI* requires is also easier to compare across collections using different information retrieval systems than the term weight information required by *gGlOSS*. Also, as shown by Callan et al. [2000] *df* information can be efficiently approximated by sampling techniques.

While *CVV* is an intuitively appealing approach that also requires limited statistical information, *CVV* tended to be the worst-performing approach in all of our experiments.

For all approaches and all testbeds, collection selection performance tended to decline slightly when shorter queries were used.

6. CONCLUSIONS

This paper reports the results of a large empirical study that has produced a number of interesting findings that can be usefully applied to the engineering of multi-collection information retrieval systems. We have presented detailed results on a number of fronts; unfortunately, this may have obscured the broader picture.

Our experiments have focused on enabling and performing comparisons of collection selection techniques. Collection selection is one subproblem of the larger task of information retrieval in multicollection environments and is concerned with effectively selecting the collections to which queries should be sent.

In preparation for our experiments, we created two testbeds that organized TREC documents into collections. We have made the assignments of documents to collections available so that other researchers can use these testbeds. We also

used an additional existing testbed for comparison. We characterized features of these testbeds, including document and relevant document distributions, to provide insight into collection selection performance for those testbeds. To evaluate collection selection performance, we collected and described a number of evaluation measures. These measures gauge the degree to which a collection ranking produced by a collection selection approach approximates a desired ranking.

We conducted comparisons of three published approaches, *CORI* [Callan et al. 1995], *gGlOSS* [Gravano and García-Molina 1995] and *CVV* [Yuwono and Lee 1997] using six test environments consisting of static and accessible collections. Overall, we found that *CORI* consistently performed the best for our test environments, with the performance of *gGlOSS* and *CVV* being more dependent on the test environment. On a more detailed front, we found that all approaches that we considered exhibit some degree of positive correlation with SBR (the number of documents per collection). A positive correlation with SBR is not necessarily a detriment; the number of *relevant* documents per collection also tends to be correlated with SBR. However, *gGlOSS* exhibits a very strong positive correlation with SBR, which led to performance downfalls for some of our test environments.

We hope that with these experiments we have helped to resolve, or at least clarify the ongoing debate about whether detailed statistical information about collections is necessary or useful. There is still substantial room for collection selection improvement, so an as-yet-unforeseen use of detailed information may eventually prove to be useful. However, our experiments showed simpler approaches to be more effective.

For future work, through careful examination of our collection selection results, we have noted instances where query-specific issues have affected average performance. For example, some queries have very few relevant documents or have relevant documents in few collections. We also noted that that collection selection and document retrieval performance can vary significantly on a query-by-query basis. We believe that an in-depth study of query-specific performance would be instructive.

REFERENCES

ABDULLA, G., LIU, B., SAAD, R., AND FOX, E. A. 1997. Characterizing world wide web queries. Tech. Rep. TR-97-04, Department of Computer Science, Virginia Polytechnic Institute and State University.

ARAÚJO, M. D., NAVARRO, G., AND ZIVIANI, N. 1997. Large text searching allowing errors. In *Proceedings of the 4th South American Workshop on String Processing (WSP '97)*. 2–20.

BAUMGARTEN, C. 1997. A probabilistic model for distributed information retrieval. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'97)*. ACM, New York, 258–266.

BAUMGARTEN, C. 1999. A probabilistic solution to the selection and fusion problem in distributed information retrieval. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*. ACM, New York, 246–253.

BUCKLEY, C. 1992. SMART version 11.0. ftp://ftp.cs.cornell.edu/pub/smart/.

CALLAN, J., CONNELL, M., AND 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, New York, 479–490.

CALLAN, J., POWELL, A. L., FRENCH, J. C., AND CONNELL, M. 2000. The effects of query-based sampling on automatic database selection algorithms. Tech. Rep. CMU-LTI-00-162, Language Technologies Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, Pa.

CALLAN, J. P., CROFT, W. B., AND HARDING, S. M. 1992. The INQUERY Retrieval System. In *Proceedings of the 3rd International Conference on Database and Expert Systems Applications (DEXA'92)*. 78–83.

CALLAN, J. P., LU, Z., AND CROFT, W. B. 1995. Searching distributed collections with inference networks. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'95)*. ACM, New York, 21–28.

CRASWELL, N., BAILEY, P., AND HAWKING, D. 2000. Server selection on the world wide web. In *Proceedings of the 5th ACM Conference on Digital Libraries*. ACM, New York, 37–46.

DOLIN, R., AGRAWAL, D., ABBADI, A. E., AND DILLON, L. 1997. Pharos: A scalable distributed architecture for locating heterogeneous information sources. In *Proceedings of the 6th International Conference on Information and Knowledge Management (CIKM'97)*. 348–355.

DOLIN, R., AGRAWAL, D., AND ABBADI, E. E. 1999. Scalable collection summarization and Selection. In *Proceedings of the 4th ACM Conference on Digital Libraries (DL'99)*. ACM, New York, 49–58.

DOLIN, R., AGRAWAL, D., ABBADI, E. E., AND PEARLMAN, J. 1998. Using Automated Classification for Summarizing and Selecting Heterogeneous Information Sources. *D-Lib Mag*. http://www.dlib.org/dlib/january98/dolin/01dolin.html.

DREILINGER, D. AND HOWE, A. E. 1997. Experiences with selecting search engines using MetaSearch. *ACM Trans. Inf. Syst. 15*, 3, 195–222.

FRENCH, J. C. 2002. Modeling web data. In *Proceedings of the Joint Conference on Digital Libraries (JCDL'02)*.

FRENCH, J. C. AND POWELL, A. L. 2000. Metrics for evaluating database selection techniques. *3*, 3, 153–163.

FRENCH, J. C., POWELL, A. L., AND CALLAN, J. 1999a. Effective and efficient automatic database selection. Tech. Rep. CS-99-08, Department of Computer Science, University of Virginia.

FRENCH, J. C., POWELL, A. L., CALLAN, J., VILES, C. L., EMMITT, T., PREY, K. J., AND MOU, Y. 1999b. Comparing the performance of database selection algorithms. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*. ACM, New York, 238–245.

FRENCH, J. C., POWELL, A. L., VILES, C. L., EMMITT, T., AND PREY, K. J. 1998. Evaluating database selection techniques: A testbed and experiment. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98)*. 121–129.

FUHR, N. 1999. A decision-theoretic approach to database selection in networked IR. *ACM Trans. Inf. Syst. 17*, 3, 229–249.

GAUCH, S., WANG, G., AND GOMEZ, M. 1996. ProFusion: Intelligent fusion from multiple, distributed search engines. *J. Univ. Comput. 2*, 9, 637–649.

GIBBONS, J. D. 1976. *Nonparametric Methods for Quantative Analysis*. Holt, Rinehart and Winston.

GRAVANO, L., CHANG, C.-C. K., GARCIA-MOLINA, H., AND PAEPCKE, A. 1997. STARTS: Stanford proposal for internet meta-searching. In *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data (SIGMOD'97)*. ACM, New York, 207–218.

GRAVANO, L. AND GARCÍA-MOLINA, H. 1995. Generalizing GlOSS to vector-space databases and broker hierarchies. In *Proceedings of the 21st International Conference on Very Large Data Bases (VLDB'95)*. 78–89.

GRAVANO, L., GARCÍA-MOLINA, H., AND TOMASIC, A. 1994. The effectiveness of GlOSS for the text database discovery problem. In *Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data (SIGMOD'94)*. ACM, New York, 126–137.

GRAVANO, L., GARCÍA-MOLINA, H., AND TOMASIC, A. 1999. GlOSS: Text-source discovery over the internet. *ACM Trans. Datab. Syst. 24*, 2, 229–264.

HARMAN, D. K., Ed. 1995. *Proceedings of the 4th Text Retrieval Conference (TREC-4)*. NIST Special Publication 500–236. Department of Commerce, National Institute of Standards and Technology, Gaithersburg, Md.

HAWKING, D. AND THISTLEWAITE, P. 1999. Methods for Information Server Selection. *ACM Trans. Inf. Syst. 17*, 1, 40–76.

IPEIROTIS, P. G. AND GRAVANO, L. 2002. Distributed Search over the Hidden Web: Hierarchical Database Sampling and Selection. In *Proceedings of the 28th International Conference on Very Large Data Bases (VLDB 2002)*. 394–405.

JANSEN, M. B. J., SPINK, A., BATEMAN, J., AND SARACEVIC, T. 1998. Real life information retrieval: A study of user queries on the web. *SIGIR Forum 32*, 1, 5–17.

LIN, Y., XU, J., LIM, E.-P., AND NG, W.-K. 1999. ZBroker: A query routing broker for Z39.50 databases. In *Proceedings of the 8th International Conference on Information and Knowledge Management (CIKM'99)*. 202–209.

LIU, K.-L., YU, C., MENG, W., WU, W., AND RISHE, N. 1999. A statistical method for estimating the usefulness of text databases. Tech. rep., Department of EECS, University of Illinois at Chicago, Chicago, Ill.

LU, Z., CALLAN, J. P., AND CROFT, W. B. 1996. Measures in collection ranking evaluation. Tech. Rep. TR-96-39, Computer Science Department, University of Massachusetts.

MENG, W., LIU, K.-L., YU, C., WANG, X., CHANG, Y., AND RISHE, N. 1998. Determining text databases to search in the internet. In *Proceedings of the 24th International Conference on Very Large Data Bases (VLDB'98)*. 14–25.

MENG, W., LIU, K.-L., YU, C., WU, W., AND RISHE, N. 1999. Estimating the usefulness of search engines. In *Proceedings of the 15th International Conference on Data Engineering*. 146–153.

MOFFAT, A. AND ZOBEL, J. 1995. Information retrieval systems for large document collections. In *Proceedings of the 3rd Text Retrieval Conference (TREC-3)*. 85–94.

POWELL, A. L. 2001. Database selection in distributed information retrieval: A study of multi-collection information retrieval. Ph.D. dissertation, Department of Computer Science, University of Virginia.

POWELL, A. L., FRENCH, J. C., CALLAN, J., CONNELL, M., AND VILES, C. L. 2000. The impact of database selection on distributed searching. In *Proceedings of the 23rd International Conference on Research and Development in Information Retrieval (SIGIR'00)*. ACM, New York, 232–239.

POWELL, J. AND FOX, E. A. 1998. Multilingual federated searching across heterogeneous collections. *D-Lib Mag*. http://www.dlib.org/dlib/september98/powell/09powell.html.

SILVERSTEIN, C., HENZINGER, M., MARAIS, H., AND MORICZ, M. 1999. Analysis of a very large Web Search Engine Query Log. *SIGIR Forum 33*, 1, 6–12.

SPINK, A. AND SARACEVIC, T. 1997. Interaction in information retrieval: Selection and effectiveness of search terms. *J. ASIS 48*, 8, 741–761.

TOMASIC, A., GRAVANO, L., LUE, C., SCHWARZ, P., AND HAAS, L. 1997. Data structures for efficient broker implementation. *ACM Trans. Inf. Syst. 15*, 3, 223–253.

VOORHEES, E., GUPTA, N. K., AND JOHNSON-LAIRD, B. 1994. The collection fusion problem. In *Proceedings of the 3rd Text REtrieval Conference (TREC-3)*. 95–104.

VOORHEES, E., GUPTA, N. K., AND JOHNSON-LAIRD, B. 1995. Learning collection fusion strategies. In *Proceedings of the 18th International Conference on Research and Development in Information Retrieval (SIGIR'95)*. ACM, New York, 172–179.

VOORHEES, E. M. 1995. Siemens TREC-4 Report: Further Experiments with Database Merging. In *Proceedings of the 4th Text REtrieval Conference (TREC-4)*. 121–130.

VOORHEES, E. M. AND TONG, R. M. 1997. Multiple Search Engines in Database Merging. In *Proceedings of the 2nd ACM International Conference on Digital Libraries (DL'97)*. 93–102.

XU, J. AND 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 (SIGIR'98)*. ACM, New York, 112–120.

XU, J., CAO, Y., LIM, E.-P., AND NG, W.-K. 1998. Database selection techniques for routing bibliographic queries. In *Proceedings of the 3rd ACM International Conference on Digital Libraries (DL'98)*. ACM, New York, 264–273.

XU, J. AND CROFT, W. B. 1999. Cluster-based language models for distributed retrieval. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99)*. ACM, New York, 254–261.

YU, C., LIU, K.-L., WU, W., MENG, W., AND RISHE, N. 1999a. Finding the most similar documents across multiple text databases. In *Proceedings of the IEEE Conference on Advances in Digital Libraries (ADL'99)*. IEEE Computer Society Press, Los Alamitos, Calif., 150–162.

YU, C., MENG, W., LIU, K.-L., WU, W., AND RISHE, N. 1999b. Efficient and effective metaSearch for a large number of text databases. In *Proceedings of the 8th International Conference on Information and Knowledge Management (CIKM'99)*. 217–224.

YUWONO, B. AND LEE, D. L. 1997. Server ranking for distributed text retrieval systems on internet. In *Proceedings of the 5th International Conference on Database Systems for Advanced Applications*. 41–49.

ZOBEL, J. 1997. Collection selection via lexicon inspection. In *Proceedings of the 2nd Australian Document Computing Symposium*. 74–80.