Collection Selection Based on Historical Performance for Efficient Processing

Christopher T. Fallen and Gregory B. Newby

Arctic Region Supercomputing Center
University of Alaska Fairbanks
Fairbanks, Alaska

{fallen,newby}@arsc.edu

ABSTRACT

A Grid Information Retrieval (GIR) simulation was used to process the TREC Million Query Track queries. The GOV2 collection was partitioned by hostname and the aggregate performance of each host, as measured by qrel counts from the past TREC Terabyte Tracks, was used to rank the hosts in order of quality. Only the 100 highest quality hosts were included in the Grid IR simulation, representing less than 20% of all GOV2 documents. The IR performance of the GIR simulation, as measured by the topic-averaged AP, b-pref, and Rel@10 over the TREC Terabyte-Track topics is within one standard deviation of the respective topic-averaged TREC Million Query participant median scores. Estimated AP of the Million Query topic results is comparable to the topic-averaged AP of the Terabyte topic results.

1. INTRODUCTION

One goal of the ARSC multisearch experiment for the 2007 TREC Million Query Track is to estimate a practical upper bound of the number of document collections that can be independently searched in a distributed information retrieval application, while simultaneously completing Track requirements. The specific question is: if a set of internet hosts, each providing its own local search service, are to be chosen nearly at random, how many hosts can be chosen if 10,000 queries are to be processed in about 100 hours? A secondary goal is to answer or at least reduce the scope of the question: how should those hosts be chosen so as to maximize the IR performance while simultaneously minimizing the number of hosts searched?

2. Grid Information Retrieval

One of the main inspirations for this work is our continued interest in distributed information retrieval systems. The authors and colleagues are involved with the Open Grid Forum's standards working group on Grid Information Retrieval. The WG is striving to develop standards for distributed search in grid computing environments [10].

GIR spans several major themes: distributed indexing, transport methods for queries and result sets, human interface, and methods for query persistence. For TREC purposes, though, the emphasis is on result set merging. The issue is that different IR systems, with different collections or subcollections, each produce their own results for a given query in a distributed IR system. How can these different sets of results (each ordered by relevance, as produced by the independent IR systems), be effectively unified into one set? This issue was investigated for TREC 2006 [9]. Rather than ranking arbitrarily, GIR seeks to produce a single relevance-ranked set, with ordering that indicates the relative relevance of each document, regardless of which IR system it came from. In recognition of the vast number of potential collections (every Web host; every database server; etc.), the current work examines the potential for automatically selecting a sestet of collections to query. This investigation is highly relevant to GIR, where collection selection is viewed as a necessary early phase of any search.

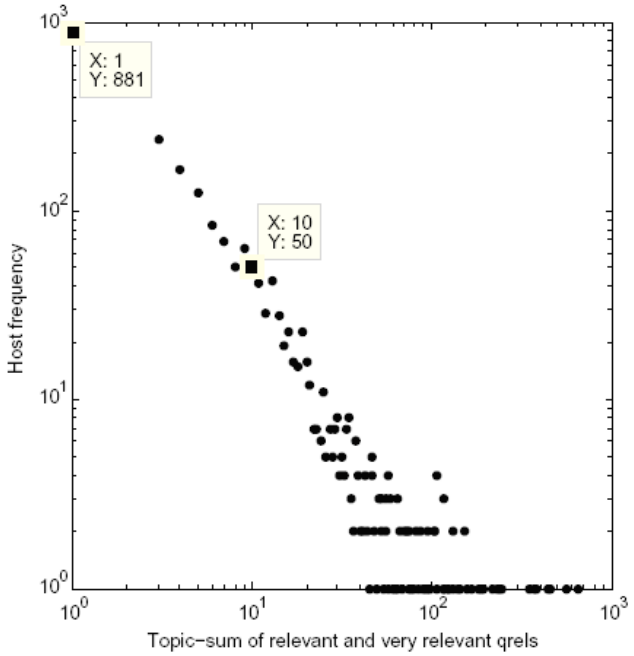
Table 1: Total count, mean and standard deviation over 149 topics of GOV2 Terabyte Track qrels

Relevance Score	$\sum$	μ	σ
Not Relevant	108,434	728	335
Relevant	22,566	151	136
Very Relevant	4351	29	49
Total	135,351	908	342

3. Host Quality and QREL Density

One criterion that may help to automatically decide which of many hosts to search for a particular query is past performance of each host for similar queries. The qrels from the TREC Terabyte Tracks are an available data set of past host relevance and were used to pick a subset of presumably high-quality hosts over which to process the 10,000 queries in the Million Query Track. The analysis in the remainder of this section also appears in [7] but is

Figure 0: Zipf plot of the host frequency vs. the total number of contained relevant and very relevant qrels over all TREC topics



included here for completeness. Each TREC Terabyte topic consists of an identifying number, a title, a description, and a narrative that encapsulates possible search keywords, an idea of the information desired, and a criterion for determining the relevance of a document to the topic. Participants could submit several runs of ranked result sets for the annual set of 50 topics. About 100 of the top documents from each of up to two runs were added to the judgment pool. The method of pooling to estimate the performance of an IR system on a large test collection is described in [11]. Assessors assign $(\text{topic}, \text{document})$ pairs in the judgment pool to relevance scores of 0, 1, or 2 to not relevant, relevant, or highly relevant pairs. The information contained in the topic is used by the assessor to determine relevance [5]. For the purpose of discussion, a *qrel* or *relevance judgment* will be defined as the triplet $(\text{topic}, \text{document}, \text{relevance score})$. The distribution of relevance judgments according to relevance score is summarized in Table 1.

From the perspective of distributed information retrieval, the set of relevance judgments for a topic can be partitioned by the document field into disjoint subsets corresponding to Web domain names. By assuming that the set of the top documents retrieved by the monolithic IR systems used in the TREC Terabyte track is about the same set of documents that would be returned by local IR search nodes at each Web host, the number of relevance judgments binned by Web domain name and topic can be used to estimate the number of domains that would respond to a given topic. Table 2 lists the number of hosts that contain the set of relevance judgments partitioned by relevance score. Many hosts that return relevant documents also return non-relevant documents so the total number of hosts responding to a query is about the same as the number of hosts retrieving non-relevant documents. Hundreds of hosts in GOV2 contain non-relevant qrels for each topic and generally only tens of hosts contain relevant qrels [7]. Using *a priori* statistical or experience-based knowledge like term-document frequencies or past performance of similar queries to identify the Web domain names most likely to retrieve documents relevant to a particular topic before transmitting the search query to remote search nodes is one way to reduce the bandwidth used during the distributed search. However, this is not viable in practice because *a priori* knowledge of which hosts contain documents that are relevant to a topic is not easily available and most hosts do not contain relevant documents for any particular topic.

Over the union of all the TREC Terabyte Track topics, only 2704 hosts out of the 6353 hosts represented in the judgment pool retrieved at least one relevant or very relevant document. As with the distribution of documents among hosts described in [7], the distribution of relevant or very relevant qrels among hosts in the judgment pool appears to follow a Zipfian distribution as illustrated in Figure 0. Quantitatively, 20% of the hosts in the judgment pool that retrieved a relevant document for any topic contain more than 78% of all the relevant and very relevant qrels. And more than 80% of the hosts that contain relevant (or very relevant) qrels for any TREC topic contain 10 or fewer relevant qrels for all topics. The qualitative similarity between the distribution of topic-aggregated relevant qrels and the distribution of documents among hosts is the premise of a model of relevance, described in [7], postulating that the number of relevant documents retrieved from a host that contains relevant documents for a topic is approximately, over many topics and hosts, proportional to the total number of documents contained in the host. This model may then be used to improve the efficiency of a distributed search. In the 2007 TREC Million Query Track where it was not practical to search every sub-collection for every query, only the sub-collections with the most relevant qrels in the previous TREC Terabyte Tracks were chosen to search. This decision is approximately equivalent to deciding to search only the largest GOV2 hosts.

Table 2: Total count of Web hosts containing the Terabyte Track qrels out of the 17,186 GOV2 hosts

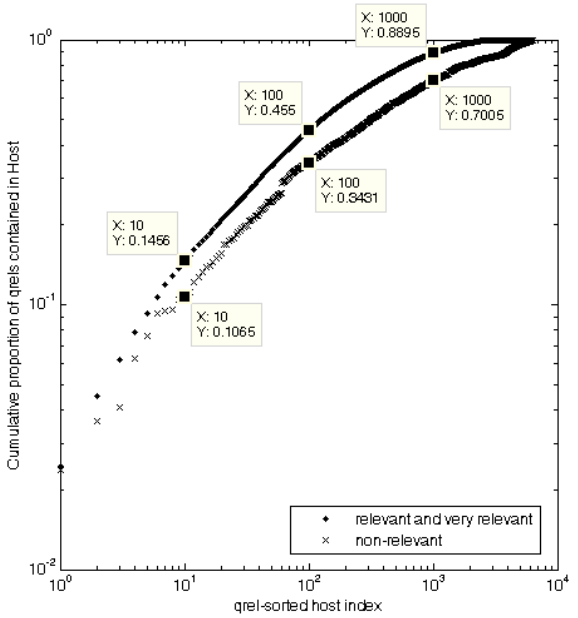
Relevance score	$\sum$
Not relevant	6109
Relevant	2553
Very relevant	970
Any	6353

4. EXPERIMENT DESCRIPTION

4.1 Search Software

Simple index and search applications were constructed from the demo classes of the Apache Lucene 2.1 toolkit. The Lucene framework has been used by the ARSC distributed IR group in a previous TREC Terabyte Track as described in [8].

Figure 1: Cumulative relevant and non-relevant qrels contained vs. qrel-sorted GOV2 Hosts



4.2 Search Hardware

ARSC's Cray XD1 ("Nelchina") was used to index the GOV2 collection and process the queries. This system was used in the 2006 TREC Terabyte Track [9] with Amberfish software [13] and this is the first year that the ARSC distributed IR group has used the Lucene toolkit on the XD1 to complete a TREC Track.

Nelchina features 108 2.6Ghz Opteron processor cores with 4GB of memory each, the PBS Pro scheduler, and Cray's variation on the SuSE Linux operating system. A disk subsystem, provided by Direct Data Networks (DDN), provides 18TB of high performance disk space for temporary storage via the Lustre cluster file system. Results from informal experiments at ARSC have revealed that five to ten simultaneous indexing or searching threads on dedicated nodes can operate simultaneously before the overall aggregate performance decreases.

4.3 Host Collection and Selection

The GOV2 collection was stored on the shared file system on the DDN disk and then partitioned into disjoint sub-collections at index-time according to hostname as described above and in [8] and [9]. The indexes were constructed using a simple Lucene-based index application and stored on the shared file system. For performance considerations, GOV2 source text was placed on a different physical disk than the target index.

Only a small fraction of the indexes were used to process all 10,000 queries. About 100 hosts or indexes could be independently searched within TREC time constraints using the software and hardware described above. The indexes chosen corresponded to the GOV2 hosts that contained the most relevant and very relevant qrels over the TREC Terabyte Tracks.

First the relevant and very relevant qrel counts were binned according to topic and containing host, so let $\mathrm{N}_{\mathrm{r}}(t,H)$ be the total number of relevant and very relevant qrels for TREC Terabyte Track topic number t and host H, defined as the set of documents in the GOV2 collection contained in the host. Then the qrel counts were summed over the topics

$$\tilde{\mathrm{N}}_{\mathrm{r}}(H)=\sum_{t=701}^{850}\mathrm{N}_{\mathrm{r}}(t,H)$$

The hosts were then sorted in descending order of the topic-summed qrel counts creating a sort permutation α so that $\tilde{\mathrm{N}}_{\mathrm{r}}(H_{\alpha(i)})$ is a monotonically decreasing function of i. Each topic was processed using the Lucene-based search application for every host in the set $\left\{H_{\alpha(i)}\right\}_{i\in\{1,\ldots,100\}}$. The cumulative proportions of relevant and non-relevant qrels

$$\sum_{i=1}^{m}\tilde{\mathrm{N}}_{\{\mathrm{r,nr}\}}\left(H_{\alpha(i)}\right)\Big/\sum_{i=1}^{6353}\tilde{\mathrm{N}}_{\{\mathrm{r,nr}\}}\left(H_{\alpha(i)}\right)$$

are plotted in Figure 1 against the number m of (relevant-qrel sorted) hosts. Similarly, the cumulative proportion of documents contained

$$\sum_{i=1}^{m}\left|H_{\alpha(i)}\right|\Big/\sum_{i=1}^{6353}\left|H_{\alpha(i)}\right|$$

is plotted in Figure 2 against m where $\left|H\right|$ denotes the cardinality or size of the set or host H. Note that the 100 hosts chosen in this experiment contain 19% of the documents in the set

Figure 2: Cumulative proportion of documents contained vs. relevant qrel-sorted hosts

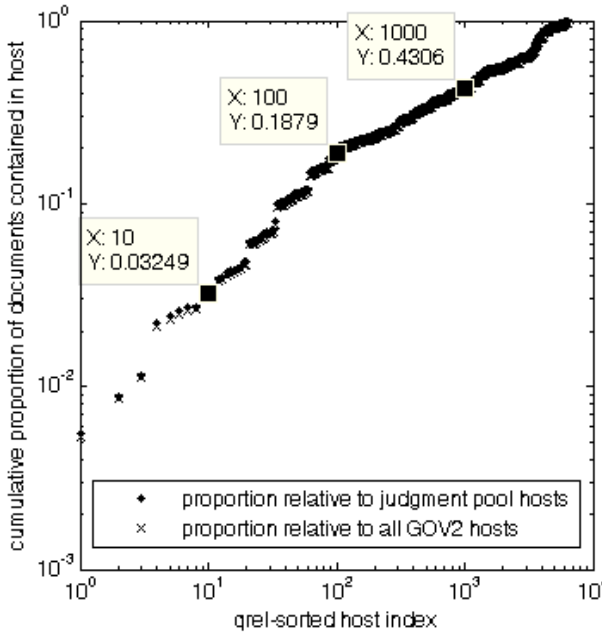
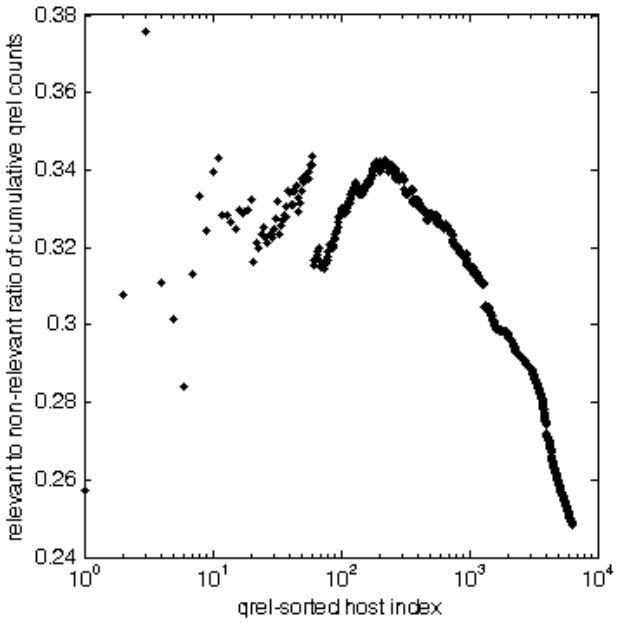


Figure 3: Ratio of cumulative sums of relevant to non-relevant qrels contained vs. relevant-qrel sorted hosts



of 6353 hosts sampled in the TREC Terabyte Track judgment pool and 18% of the documents in all the GOV2 hosts. Plotted in Figure 3 is the ratio of cumulative sums of relevant to non-relevant qrels

$$\sum_{i=1}^{m} \tilde{N}_{r}\left(H_{\alpha(i)}\right) \Big/ \sum_{i=1}^{m} \tilde{N}_{nr}\left(H_{\alpha(i)}\right)$$

as a function of m. The steady downward trend following the initial noise in the function of qrel-sorted hosts may indicate that only the first 300 or so most relevant hosts, just under 5% of all the hosts represented in the TREC Terabyte Track judgment pool or 27% of all GOV2 documents, can be included in distributed searches before the number of non-relevant documents likely to be found in the remaining hosts increases faster than the number of relevant documents likely to be found. Or in other words, the judgment pool sample of the aggregate performance of many systems over many topics begins to decrease as more hosts are included in the pool, where performance is defined by the total number of relevant documents found relative to the non-relevant documents. Therefore, it may be possible to increase the efficiency of a distributed, or even monolithic, search by restricting the search to only those hosts that performed well, i.e. to those hosts that contained the most relevant documents in the TREC Terabyte Track judgment pool drawn from the top documents from each system [4], over a sufficient number of topics without sacrificing IR performance of the search or perhaps even improving it.

Note that the results and inferences made here are based on general observations of judgment pool results sampled and contributed over many topics from a wide variety of systems and may not hold true for a single system or topic applied to a restricted domain. The number of topics in the TREC Terabyte Track is small relative to the number of likely topics a typical IR system may process. Results from the Million Query Track may help to distinguish how much of the aggregate trends can be used to improve the performance of a single system over most topics and how stable the trends are over many more topics.

4.4 Batch Query Processing Method

Between six to twelve search processes were started on three to six, respectively, dual processor-core nodes on Nelchina. A 48-hour job wall-clock limit is enforced on Nelchina, a shared resource at the Arctic Region Supercomputing Center, so several jobs were needed to process the entire 10,000 query Million Query set. The number of nodes used for any particular job depended strongly on the number of nodes available at the start of the job. Each search process selected queries from a query queue and ran the queries against the 100-host collection described in Section 4.3, saving the results to disk. After processing the entire set of queries, the TREC-formatted results from the jobs were concatenated to the final set of results and submitted to NIST.

5. IR PERFORMANCE RESULTS

Aggregate topic-averaged IR performance measures of the results submitted during this experiment, labeled `ffind07d`, and those submitted by MQ participants are presented in Table 3. The IR performance was calculated relative to the TREC Terabyte topics and qrels. Note that the performance of `ffind07d` is within one standard deviation of the MQ participant median even though nearly 81% of the GOV2 collection was discarded at search time. While it is not very surprising when commercial-quality IR software performs reasonably well at searching a fraction of the GOV2 collection already known to contain many documents relevant to the topics, it might be surprising if searching the same fraction of GOV2 over many new MQ topics yielded comparable IR performance results. A judgment pool constructed by MQ

Table 3: Mean and standard deviation of AP, bpref, and REL@10 over the 149 TREC Terabyte Topics

	MQ participant						`ffind07d`	
	worst		median		best			
	μ	σ	μ	σ	μ	σ	μ	σ
AP	0.0101	0.0302	0.2219	0.1529	0.3956	0.1787	0.1360	0.1298
bpref	0.0230	0.0477	0.2958	0.1845	0.4697	0.1854	0.2253	0.1760
REL@10	0.3423	0.9283	4.7718	2.6256	7.9262	2.4192	2.6510	2.5809

Track organizers using the *expected AP method* [3] and the *statistical evaluation method* [1] will be used to evaluate MQ system performance. The mean average precision of `ffind07d` estimated by the statistical evaluation method was 0.1633 over 1083 valid MQ topics and is within one standard deviation of the MAP calculated over the 149 Terabyte Track Topics. This is at least a promising indication that a distributed GIR search application could currently be constructed with IR performance comparable to that currently obtained by monolithic search applications provided that a relatively small number of likely relevant document collections can be pre-selected at search time. It is of additional interest to note that the estimated MAP of the system on the 1083 valid MQ topics is near the calculated MAP of the system on the 149 Terabyte topics even though the document collection searched over all topics corresponded to just 100 out of over 17,000 hosts in GOV2 that contained many relevant documents for the Terabyte topics. A stronger inference may be possible after comparing the statistical evaluation method estimated MAP here to the estimated MAP of other MQ participant systems.

6. TOWARD A GOV2 HOST SPACE

Due to bandwidth and other IR performance constraints inherent in any loosely coupled distributed information retrieval task like multisearch, an Internet-scale Grid IR application that effectively and dynamically relevance-ranks entire collections relative to a user query will tend to provide superior performance to the user in both search quality and bandwidth cost relative to a Grid IR application that searches all available collections for every query [12]. The vector space or document configuration space model [15, 16] for relevance-ranking documents with respect to a query has been enormously successful, and variations of it are used to some extent in nearly every IR application available today.

Conceptually, there are many seemingly reasonable and straightforward ways to extend a vector space model for automatic indexing of documents to the automatic indexing of entire hosts or other subsets of documents such as relevance-ranked result-sets for the purpose of collection ranking and result-set merging. Practical considerations ranging from the cooperativeness of the distributed document collections [17] to user expectations and requirements, however, will inevitably narrow the range of models that may be considered reasonable and possibly even preclude straightforward extensions of a typical document vector space model entirely. Vector space models for the automatic indexing and query relevance ranking of entire sub-collections include the "big document" approach briefly summarized in [17] where a union of documents in a collection is used to represent the collection as a single document, but may also include host vector spaces that are constructed dynamically at query time. One possible method may be to extract cluster descriptor terms from each of the collection subsets [6] and use those to define the basis of a coarse sub-collection configuration space. A simple similarity measure based on the cardinality of the intersection of descriptor sets may allow the *Cluster Hypothesis* [14] to be tested on collection subsets analogous to how it has been tested on documents. Broad but shallow judgment pools as constructed in the TREC Million Query Track will help to test this hypothesis as any subset similarity measure implemented in an efficient Grid IR application will likely need to operate on collection result-sets comparable in size to the single page of results.

6.1 Host Space Configuration Model

Consider the set G of GOV2 documents and let $\mathfrak{H}$ be a partition of G defined by the equivalence relation "is contained in the same web host as". An element $H \in \mathfrak{H}$ is a host or subset of G containing the documents available to the local search service of the host. Define the host space V_H as a subspace of $\mathbb{R}^{|H|}$ where $|H|$ is the number of documents or elements in the document set H. Put the documents of H in correspondence with the elements of the natural or canonical basis $\mathfrak{V}_H$ of V_H. Weighted sums of the documents or basis elements of $\mathfrak{V}_H$ can be associated with ranked result-sets or other arbitrary subsets of H.

Similarly, define the document space V_{T_H} as a subspace of $\mathbb{R}^{|T_H|}$, where T_H is set of stemmed and stop-filtered index terms of H and $|T_H|$ is the number of terms. Initially assume that the index terms are conceptually independent or orthogonal so each term can be put in correspondence with an element in the canonical or natural basis $\mathfrak{V}_{T_H}$ of V_{T_H}. A multi-term query submitted by a user can be represented as a point in V_{T_H} by summing the associated basis terms or elements in $\mathfrak{V}_{T_H}$. The document space V_{T_H} as defined above is similar in concept to the classic document space configuration in [15]. When considering problems of distributed information retrieval it will be convenient to treat document subsets such as result-sets from hosts or other collections as vectors in the host space V_H so that measures of similarity, akin to vector space similarity measures between documents, between result-sets from a host collection can be defined as functions of points in a vector space.

6.2 Example: Local result-set optimization

For instance, a local search service of host H can be represented as a map $\mathrm{s}_H : V_{T_H} \to V_H$ that sends a query, say of two terms $\mathbf{q}_H = \hat{\mathbf{e}}_{T_H^1} + \hat{\mathbf{e}}_{T_H^2}$ from the basis $\mathfrak{V}_{T_H}$ to a result set of, say three documents $\mathbf{d}_H = \hat{\mathbf{e}}_{H^1} + \hat{\mathbf{e}}_{H^2} + \hat{\mathbf{e}}_{H^3}$ from the basis $\mathfrak{V}_H$. A user that wishes to search multiple hosts $\{H_\alpha\}_\alpha$ simultaneously for a query will submit the query to a query processor or search portal that sends queries $\{\mathbf{q}_{H_\alpha}\}_\alpha$ to the respective search services $\{\mathrm{s}_{H_\alpha}\}_\alpha$ and then will be presented with multiple result sets $\{\mathbf{d}_{H_\alpha}\}_\alpha$ that will likely be merged into a single ranked list of results. Each result set $\mathbf{d}_{H_\alpha}$ will typically contain many more non-relevant documents than relevant documents so truncating or renormalizing the result set before transmitting it to the user is often desirable for efficiency [7] or search quality [2]. Then

finding result sets in V_{H_α} near to but shorter than $\mathbf{d}_{H_\alpha}$ or perhaps to a set of host documents known a priori to be relevant could be accomplished with a suitable similarity measure of result sets.

7. REFERENCES

[1] J. A. Aslam and V. Pavlu, *A Practical Sampling Strategy for Efficient Retrieval Evaluation*, Northeastern University, 2007.

[2] A. L. Calvé and J. Savoy, *Database merging strategy based on logistic regression*, Information Processing and Management, 36 (2000), pp. 341-359.

[3] B. Carterette, J. Allan and R. Sitaraman, *Minimal Test Collections for Retrieval Evaluation*, *19th annual international ACM SIGIR conference on Research and development in information retrieval*, ACM Press, Seattle, Washington, USA, 2006.

[4] C. L. A. Clarke, N. Craswell and I. Soboroff, *Overview of the TREC 2004 Terabyte Track*, *The Thirteenth Text REtrieval Conference*, NIST Special Publications, Gaithersburg, Maryland, USA, 2004.

[5] C. L. A. Clarke and F. Scholer, *The TREC 2005 Terabyte Track*, *The Fourteenth Text REtrieval Conference*, NIST Special Publications, Gaithersburg, Maryland, USA, 2005.

[6] D. Dubin, *Structure in Document Browsing Spaces*, University of Pittsburgh, Pittsburgh, 1996.

[7] C. T. Fallen and G. B. Newby, *Distributed Web Search Efficiency by Truncating Results*, *JCDL '07*, ACM, Vancouver, British Columbia, Canada, 2007.

[8] C. T. Fallen and G. B. Newby, *Logistic Regression Merging of Amberfish and Lucene Multisearch Results*, *The Fourteenth Text REtrieval Conference*, NIST Special Publications, Gaithersburg, Maryland, USA, 2005.

[9] C. T. Fallen and G. B. Newby, *Partitioning the Gov2 Corpus by Internet Domain Name: A Result-set Merging Experiment*, *The Fifteenth Text REtrieval Converence*, NIST Special Publications, Gaithersburg, Maryland, USA, 2006.

[10] K. Gamiel, G. B. Newby and N. Nassar, *Grid Information Retrieval Requirements (GFD.27)*, *Global Grid Forum*, Lamont, IL, 2003, pp. 18.

[11] K. S. Jones and C. V. Rijsbergen, *Report on the need for and provision of an "ideal" information retrieval test collection*, *Technical Report*, Computer Laboratory, University of Cambridge, 1975.

[12] W. Meng, C. T. Yu and K.-L. Liu, *Building efficient and effective metasearch engines*, ACM Computing Surveys, 34 (2002), pp. 48-49.

[13] N. Nassar, *Amberfish at the TREC 2004 Terabyte Track*, *The Thirteenth Text REtrieval Conference*, NIST Special Publications, Gaithersburg, Maryland, USA, 2004.

[14] G. Salton, *Automatic Text Processing*, Addison-Wesley, Reading, MA, 1989.

[15] G. Salton, A. Wong and C. S. Yang, *A Vector Space Model for Automatic Indexing*, Communications of the ACM, 18 (1975), pp. 613-20.

[16] J. W. Sammon, *Some Mathematics of Information Storage and Retrieval*, Griffis Air Force Base, New York, 1968.

[17] L. Si, *Federated Search of Text Search Engines in Uncooperative Environments*, *Language Technology Institute*, Carnegie Mellon University, 2006.