The Effect of Database Size Distribution on Resource Selection Algorithms

Luo Si and Jamie Callan

School of Computer Science, Carnegie Mellon University.
5000 Forbes Ave, Pittsburgh PA, U.S.A.
{lsi, callan}@cs.cmu.edu

Abstract. Resource selection is an important topic in distributed information retrieval research. It can be a component of a distributed information retrieval task and can also serve as an independent application of database recommendation system together with the resource representation part. There is a large body of valuable prior research on resource selection but very little has studied about the effects of different database size distributions on resource selection. In this paper, we propose extended versions of two well-known resource selection algorithms: CORI and KL divergence in order to consider the factors of database size distributions, and compare them with the lately proposed Relevant Document Distribution Estimation (ReDDE) resource selection algorithm. Experiments were done on four testbeds with different characteristics, and the ReDDE and the extended KL divergence resource selection algorithm have been shown to be more robust in various environments.

1 Introduction

The proliferation of online searchable databases on local area networks and the Internet has increased attention on the problem of finding information that may be distributed among many text databases (*distributed information retrieval*) [1]. There are three important sub-problems in distributed information retrieval: first, the content of each text database must be represented in a suitable form (resource representation); second, given an information need (a query), several relevant databases must be selected to do the search (resource selection); and third, the results from searching the selected databases have to be merged into a single final list (result merging) [1].

This paper focuses on resource selection. Many valuable resource selection algorithms have been proposed and compared in the literature [1, 2, 3, 8, 9], but no thorough study has been conducted to study the performance of different algorithms in environments with different database size distributions. Two well-known algorithms, CORI [1, 2] and KL divergence [7, 9] resource selection, were evaluated and extended to explicitly incorporate database size factors. Four testbeds with different characteristics were created. The CORI and the KL divergence algorithms (and their extensions) were compared to the recently-proposed ReDDE algorithm [8]. The experiments support two conclusions. First, in order to have good performance in dif-

ferent environments, it is very important to include a database size factor into a resource selection algorithm. Second, the ReDDE and the extended KL divergence algorithms are the better and more stable algorithms among the algorithms considered in this research.

The rest of this paper is organized as follows. Section 2 discusses previous research. Section 3 describes the proposed extensions to the CORI and KL divergence resource selection algorithms; it also describes the ReDDE algorithm. Section 4 discusses our experimental methodology. Section 5 presents and analyzes experimental results. Section 6 concludes and discusses future work.

2 Previous Work

There has been considerable research on acquiring resource representation, resource selection, and merging results. As this paper is focused on resource selection algorithm, we mainly survey related work in resource representation and resource selection in this section, and briefly introduce the work of results merging.

2.1 Resource Representation

The STARTS protocol is one solution for acquiring resource descriptions [4]. It needs the cooperation from every resource provider to provide accurate resource descriptions. STARTS is a good solution in environments where cooperation can be guaranteed. However, it does not work in multi-party environments where some resource providers may not cooperate.

Query-based sampling [1, 2] is an alternative approach to acquiring resource descriptions that does not require explicit cooperation from resource providers. Query-based sampling only asks the search engines to run queries and return a list of downloadable documents. In practice, this solution has been shown to acquire rather accurate resource descriptions using a relatively small number of randomly-generated queries (e.g., 75) to retrieve a relatively small number of documents (e.g., 300).

In order for a resource selection algorithm to use database size information each resource representation must contain database size statistics. In cooperative environments each database may cooperate to provide this information. In uncooperative environments learning algorithms must be applied to estimate database sizes. The capture-recapture [5] and the sample-resample algorithms [8] have been proposed to fulfill this task. In the recent work [8], the sample-resample algorithm was shown to acquire relatively accurate database size estimates efficiently and robustly in different environments. It can be extremely efficient when query-based sampling is used to create resource descriptions [8].

2.2 Resource Selection

The goal of resource selection is to select a small set of resources that contain a lot of relevant documents. There are many successful resource selection algorithms, for example, CORI [1, 2], KL divergence [7, 9] and ReDDE [8].

The CORI algorithm uses a Bayesian inference network and an adaptation of the Okapi term frequency normalization formula to rank resources. Prior research by several different researchers using different datasets has shown the CORI algorithm to be one of the most stable and effective algorithms [1, 3]. The belief $p(r_k \mid c_i)$ in database db_i according to the query term r_k is determined by:

$$T = \frac{df}{df + 50 + 150 * cw_i / avg\_cw} \tag{1}$$

$$I = \frac{\log(\frac{|DB| + 0.5}{cf})}{\log(|DB| + 1.0)} \tag{2}$$

$$p(r_k \mid c_i) = b + (1 - b) * T * I \tag{3}$$

where: df   is the number of documents in db_i that contain r_k;
         cf   is the number of databases that contain r_k;
         $|DB|$ is the number of databases to be ranked;
         cw_i is the number of words in db_i;
         avg_cw is the average cw of the databases to be ranked; and
         b   is the default belief, usually set to 0.4.

The Kullback-Leibler (KL) divergence collection selection method was proposed by Xu and Croft [9]. In their work, the Kullback-Leibler divergence between the word frequency distribution of the query and the database is used to measure how well the content of the database matches with the query. This algorithm can be represented in the language modeling framework as [7]. The algorithm and an extension are described in the next section.

The database size normalization components of the CORI and KL-divergence algorithms are weak, at best. CORI represents database size by the number of words the database contains as compared with a mean (cw_i and avg_cw). The KL-divergence algorithm has no explicit normalization for database size. Indeed, although much valuable research has studied the performance and robustness of resource selection algorithms [3, 8, 9], very little of it has done evaluated the effects of different database size distributions on resource selection.

The ReDDE resource selection algorithm [8] is a newly proposed algorithm that explicitly tries to estimate the distribution of relevant documents across the set of available databases. The ReDDE algorithm considers both content similarity and database size when making its estimates (see Section 3).

2.3 Result Merging

The database representation and the resource selection components can be combined to create a database recommendation system that suggests which online databases are the best for a query. If the user wants the system to further search the selected databases and merge the returned results from each database, a result-merging component is needed. For example, the Semi Supervised Learning (SSL) [6] result merging algorithm uses the documents acquired by query-based sampling as training data and linear regression to learn the database-specific, query-specific merging models.

3 Resource Selection Algorithms That Normalize for Database Size

In this section, we extend the CORI and KL-divergence resource selection algorithms to normalize for database sizes; we also describe the ReDDE algorithm.

3.1 The Extension of the CORI Resource Selection Algorithm

Callan [1] pointed out that Equation 1 in the CORI resource selection algorithm is a variation of Roberson's term frequency (tf) weight, in which term frequency (tf) is replaced by document frequency (df), and the constants are scaled by a factor of 100 to accommodate the larger df values. We generalize Equation 1 to create Equation 4 as follows:

$$T = \frac{df}{df + df\_base + df\_factor * cw_i / avg\_cw} \tag{4}$$

where df_base and df_factor represents the two constants. There are three issues in the above formula that should be addressed to incorporate the database size factor.

First, df in Equation 4 is the number of sampled documents from the ith database that contain r_k;. In order to estimate the actual number of corresponding documents in the database, it should be scaled as:

$$df^{'} = \frac{df}{N_{C_i\_samp}} * \hat{N}_{C_i} \tag{5}$$

where $N_{C_i\_samp}$ is the number of documents sampled from the database, and $\hat{N}_{C_i}$ is the estimated number of documents in the database.

Second, the cw_i in Equation 4 is the estimated number of words in the ith database, which is calculated from the documents sampled from the database. In order to consider the database size, this number should also be scaled. Formally as:

$$cw_i^{'} = \frac{1}{N_{C_i\_samp}} * \hat{N}_{C_i} * cw_i \tag{6}$$

Taking a similar approach to avg_cw in Equation 4 yields:

$$avg\_cw^{'} = \frac{1}{C} \sum_i \frac{1}{N_{C_i\_samp}} * \hat{N}_{C_i} * cw_i \qquad (7)$$

Finally, just as the original CORI algorithm adjusts the tf_base and tf_factor constants by a factor of 100 to accommodate the larger df values (among several hundred sampled documents), we scale the tf_base and tf_factor constants as shown in Equations 8 and 9 to compensate for variations in database size.

$$tf\_base^{'} = \frac{\hat{N}_{C_i}}{N_{C_i\_samp}} * 50 \qquad (8)$$

$$tf\_factor^{'} = \frac{\hat{N}_{C_i}}{N_{C_i\_samp}} * 150 \qquad (9)$$

In this work, we make two extensions of the CORI algorithm to address these two issues. The first extension, which we call CORI_ext1, uses the new document frequency, the new collection word count and the new average collection word count in Equation 5, 6 and 7. The second extension, which we call CORI_ext2, leverages all the new document frequency, the new collection word count (average count) and the new tf_base and tf_factor constants.

3.2 The Extension of the KL-Divergence Resource Selection Algorithm

The KL-divergence resource selection algorithm has been given an interpretation in the language-modeling framework in [7]. In this framework, the documents sampled from a database are collapsed into one single, giant 'document' and used in the computation of the collection-query similarity. More formally, we need to find the collections that have largest probabilities of P (Ci | Q), i.e. the conditional probability of predicting the ith database given the query. By the Bayesian rule, P (Ci | Q) can be calculated as:

$$P(C_i \mid Q) = \frac{P(Q \mid C_i) * P(C_i)}{P(Q)} \qquad (10)$$

where P (Q | Ci) denotes the probability of generating the query Q from the ith database. P(Ci) is the prior probability of the specific database. P(Q) is the generation probability of the query, which is database independent and can be neglected. Collections with the largest generation probabilities P (Ci | Q) are selected.

Following the principle of a unigram language model with independent terms, the value of P (Q | Ci) is calculated as:

$$P(Q \mid C_i) = \prod_{q \in Q} \left(\lambda P(q \mid C_i) + (1 - \lambda) P(q \mid G) \right) \qquad (11)$$

where q is a query term. P (q | Ci) is the language model for the ith database and P (q | G) is the language model for the whole collection. The linear interpolation constant λ smoothes the collection-based language model with the global language model, which was set to 0.5 in our experiments.

The only item left in Equation 10 that needs to be estimated is the prior probability P(Ci). If it is assigned as a uniform distribution, the language model resource selection algorithm can be shown by simple mathematical manipulation to be equal to the original KL-divergence algorithm [7, 9].

The natural way to introduce a database size factor into the KL-divergence resource selection algorithm is to assign the database prior probabilities according to the database size. Formally as:

$$P(C_i) = \frac{\hat{N}_{C_i}}{\sum_j \hat{N}_{C_j}} \tag{12}$$

where the denominator is the estimated total testbed size. Plugging Equation 11 and 12 into Equation 10, we get a new version of the KL-divergence resource selection algorithm, which we call the KL_ext algorithm.

3.3 The ReDDE Resource Selection Algorithm

The newly proposed ReDDE (Relevant Document Distribution Estimation) algorithm explicitly estimates the distribution of relevant documents across all the databases. It considers both content similarity and database size when making its estimates.

The number of documents relevant to query Q in the ith database Ci is estimated as:

$$\mathrm{Re}\,1\_\hat{Q}(i) = \sum_{d_j \in C_{i\_samp}} P(rel \mid d_j) * \frac{1}{N_{C_{i\_samp}}} * \hat{N}_{C_i} \tag{13}$$

Again, $N_{C_{i\_samp}}$ is the number of sampled documents from the ith database and $\hat{N}_{C_i}$ is the estimated size of this database. P (rel | dj) is the probability of relevance given a specific document. The probability is calculated by reference to the *centralized complete database*, which is the union of all of the individual databases available in the distributed IR environment. We simplify P (rel | dj) as the probability of relevance given the document rank when the centralized complete database is searched by an effective retrieval method. The probability distribution function is a step function, which means that for the documents at the top of the ranked list the probabilities are a positive constant, and are 0 otherwse. This can be modeled as follows:

$$P(rel \mid d_j) = \begin{cases} C_Q & \text{if } Rank\_central(d_j) < ratio * \sum_i \hat{N}_{C_i} \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

The centralized complete database is not accessible, but the rank of a document in the centralized complete database can be estimated, as shown below, from its rank

within the *centralized sample database,* which is the union of all the sampled documents from different databases.

$$Rank\_\hat{Central}(d_j) = \sum_{\substack{d_k \\ Rank\_Samp(d_k)<Rank\_Samp(d_j)}} \frac{N_{c(d_k)}}{N_{c(d_k)\_samp}} \quad (15)$$

The last step is to normalize the estimates in Equation 13 to remove the query dependent constants, which provides the distribution of relevant documents among the databases.

$$Dist\_\hat{Rel}\_Q(i) = \frac{\hat{Rel\_Q}(i)}{\sum_j \hat{Rel\_Q}(j)} \quad (16)$$

ReDDE selects the databases with the highest mass in the distribution.

3 Experimental Methodology

Two well-known testbeds were used in our experiments.

1. **Trec123-100col-bysource:** 100 databases were created from TREC CDs 1, 2 and 3. They were organized by source and publication date [1]. The sizes of the databases are not skewed. This testbed has been used often in prior research [1, 7, 8].

2. **Trec4-kmeans:** 100 databases were created from TREC 4 data. A k-means clustering algorithm was used to organize the databases by topic [9], so the databases are homogenous and the word distributions are very skewed. The sizes of the databases are moderately skewed.

The characteristics of these two testbeds are shown in Table 1.

Table 1. Summary statistics for the Trec123-100colbysource and Trec4-kmeans testbeds.

Name	Query Count	Size (GB)	Number of Docs			Megabytes (MB)		
			Min	Avg	Max	Min	Avg	Max
Trec123	50	3.2	752	10782	39713	28.1	32	41.8
Trec4_kmeans	50	2.0	301	5675	82727	3.9	20	248.6

To investigate the effect of various database size distributions on resource selection algorithms, two new testbeds were created based on the Trec123-100col testbed.

1. **Trec123-AP-WSJ-60col ("relevant"):** The 24 Associated Press collections in the trec123-100col-bysource testbed were collapsed into a single large APall database. The 16 Wall Street Journal collections were collapsed into a single large WSJall collection. Details are shown in Table 2. The other 60 collections were left unchanged. The APall and WSJall collections are much larger than the other 60 databases, and they also have a higher density of documents relevant to TREC queries than the other collections.

2. **Trec123-FR-DOE-81col ("nonrelevat"):** The 13 Federal Register collections and
   the 6 Department of Energy collections in the trec123-100col-bysource testbed
   were collapsed into two large collections FRall and DOEall respectively. Details
   are shown in Table 2. The other 81 collections were left unchanged. The FRall and
   DOEall databases are much larger than the other databases, but have a much lower
   density of relevant documents than other databases.

Table 2. Summary statistics for the large databases.

Collection	Num of Docs	Size (MB)
APall	242,918	764.3
WSJall	173,252	533.2
FRall	45,820	491.6
DOEall	226,087	192.7

These two testbeds were created to simulate environments where large collections
contain i) a large percentage of relevant documents, and ii) a small percentage of rele-
vant documents.

50 short queries were created from the title fields of TREC topics 51-100 for the
Trec123-100bycol, relevant and nonrelevant testbeds. 50 longer queries were created
from the description fields of TREC topics 201-250 for the Trec4-kmeans testbed.

4 Experimental Results

The experiments were done on 4 testbeds to evaluate 6 resource selection algorithms,
namely CORI, CORI_ext1, CORI_ext2, KL divergence, KL_ext and the ReDDE
algorithms. The testbeds cover a wide range of conditions: relatively uniform sizes
and content distribution (trec123-100col), relatively uniform sizes and skewed content
distribution (trec4-kmeans), bimodal size distribution and a much higher density of
relevant documents in the large databases (relevant), and bimodal size distribution and
a much lower density of relevant documents in the large databases (nonrelevant).

Query based sampling was used to acquire the database resource descriptions by
repeatedly sending one-term queries to databases and downloading the top 4 docu-
ments returned until 300 unique documents were downloaded. The sample-resample
method was used to estimate database sizes.

Resource selection algorithms are typically compared using the recall metric R_n [1,
3, 8]. Let us denote B as a baseline ranking, which is often the RBR (relevance based
ranking), and E as a ranking provided by a resource selection algorithm. And let B_i
and E_i denote the number of relevant documents in the ith ranked database of B or E.
Then R_n is defined as shown below.

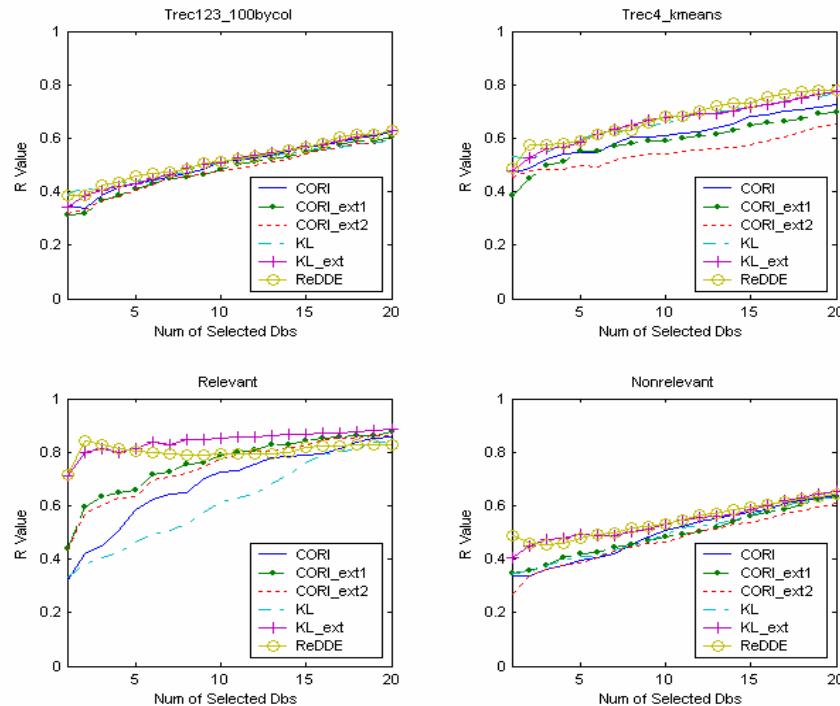$$R_k = \frac{\sum_{i=1}^{k} E_i}{\sum_{i=1}^{k} B_i} \tag{17}$$

Fig. 1. The performance of the resource selection algorithms on four different testbeds.

Usually the goal is to search only a few databases, so in our experiments the algorithms were only evaluated over the top 20 databases as shown in Figure 1.

The resource selection algorithms that do not consider the database size factor (CORI and KL-divergence algorithm) do reasonably well on the testbeds with normal or modestly skewed database size distributions (trec123_100bycol and the trec4_kmeans testbed), but their performance on the testbeds with very skewed database size distributions (relevant and nonrelevant testbeds) are not satisfactory. This suggests that the database size distribution is a very important factor for robust resource selection algorithms that must work in a wide range of environments.

The two extensions of the CORI resource algorithms (CORI_ext1 and CORI_ext2) have inconsistent performance on the four testbeds. Sometimes they are better than the original version (relevant testbed), and sometimes they are even worse than the original algorithm (trec4_kmeans testbed). This does not mean there is no other good approach to modifying the CORI algorithm; it just indicates that the two extensions proposed in this paper are not robust.

The extension of the KL- divergence algorithm (KL_ext) and the ReDDE algorithm had consistently good performance on all of the four testbed. We conclude that the database size normalizations in the ReDDE algorithm and the extension of the KL-divergence resource algorithm are effective, and that normalizing for database sizes is important to obtaining robust behavior when database sizes are skewed.

5 Conclusions and Future Work

 Resource selection is a hot research topic in distributed information retrieval and many valuable algorithms have been studied. However, to our knowledge the effects of database size distributions on different resource selection algorithms has not been evaluated thoroughly. In this work, we propose extensions of two well-known algorithms: CORI and KL-divergence. We evaluated original and extended versions of several resource selection algorithms on four testbeds that have different characteristics. The experiments show that it is important to include a database size factor into resource selection algorithms; the ReDDE and extended KL-divergence algorithms had the best and most stable performance among the algorithms evaluated.

By carefully studying the experiment results, it can be seen that the advantage of the ReDDE algorithm when selecting more than 10 databases is smaller than when selecting a few databases (e.g. relevant testbed). We attribute this to the simplification of Equation 14 with a step function. We believe that the function can be estimated by using training data to improve the accuracy in the future work.

References

1. Callan, J.: Distributed information retrieval. In: Croft, W.B. (eds.): Advances in Information Retrieval. Kluwer Academic Publishers (2000) 127-150
2. Callan, J., Connell, M.: Query-based sampling of text databases. ACM Transactions on Information Systems (2001) 97-130
3. French, J.C., Powell, A.L., Callan, J., Viles, C.L., Emmitt, T., Prey, K.J., Mou, Y.: Comparing the performance of database selection algorithms. In Proceedings of the Twenty Second Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (1999)
4. Gravano, L., Chang, C., Garcia-Molina, H., Paepcke, A.: STARTS: Stanford proposal for internet Meta-Searching. In Proceedings of the ACM-SIGMOD International Conference on Management of Data (1997)
5. Liu, K.L., Yu, C., Meng, W., Santos, A., Zhang, C.: Discovering the representative of a search engine. In Proceedings of 10th ACM International Conference on Information and Knowledge Management (2001)
6. Si, L., Callan, J.: Using sampled data and regression to merge search engine results. In Proceedings of the Twenty Fourth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (2002)
7. Si, L., Jin, R., Callan, J., Ogilvie, P.: A language model framework for resource selection and results merging. In Proceedings of the eleventh International Conference on Information and Knowledge Management, ACM (2002)
8. Si, L., Callan, J.: Relevant document distribution estimation method for resource selection. In Proceedings of the Twenty Fifth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (2003)
9. Xu, J., Croft, W.B.: Cluster-based language models for distributed retrieval. In Proceedings of the Twenty Second Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (1999)