Database Selection Using Document and Collection Term Frequencies

Rashmi Srinivasa, Tram Phan, Nisanti Mohanraj, Allison Powell, Jim French
Department of Computer Science, University of Virginia
{rashmi, tap5r, nm3j, alp4g, french}@cs.virginia.edu

Technical Report CS-2000-32
May 16, 2000

Abstract

We examine the impact of two types of information — document frequency (*df*) and collection term frequency (*ctf*) — on the effectiveness of database selection. We introduce a family of database selection algorithms based on this information, and compare their effectiveness to two existing database selection approaches, CORI [3] and gGlOSS [9]. We demonstrate that a simple selection algorithm that uses only document frequency information is more effective than gGlOSS, and achieves effectiveness that is very close to that of CORI.

1 Introduction

As the number of online databases or collections increases, database or collection selection becomes an important problem. For a given query, an exhaustive search over all databases can be expensive in terms of time and/or money. Therefore, there is a need for a selection algorithm that prunes the search space of databases without significantly reducing retrieval effectiveness.

Database selection is one step in the larger problem of distributed information retrieval which is composed of three fundamental activities:

- database selection, choosing the specific databases to search;
- searching the chosen databases; and
- merging the results into a cohesive response.

In this work, we focus specifically on database selection. The database selection problem can be described as follows. Given a query Q and a set of databases $\{DB_1, DB_2, ..., DB_N\}$, we are required to rank the databases such that visiting the databases in that order results in a high retrieval effectiveness for Q. For the experiments reported here, the performance of a selection technique is measured as how well the technique estimates the given baseline ranking. Database selection is also called collection selection [3] or text database resource discovery [10].

Several algorithms for database selection have been proposed and independently evaluated [1-4, 8, 9, 12-14, 16-20]. Two well-known selection algorithms, CORI [3] and gGlOSS [9], have been compared to each other, and CORI has been found to be more effective than gGlOSS when the goal is to locate collections containing the largest number of relevant documents [6].

In this work we introduce a family of database selection algorithms that use one or both of document frequency (*df*) information and collection term frequency (*ctf*) information. We demonstrate that a simple technique that uses only document frequency information achieves retrieval effectiveness that is close to that of CORI and better than that of gGlOSS. Adding collection term frequency considerations does not significantly improve, and sometimes penalizes, retrieval effectiveness.

2 Related Work

2.1 Selection Techniques

The generalized Glossary-Of-Servers Server (gGlOSS) selection algorithm [9] estimates the *goodness* of every database with respect to the given query, and then ranks the databases according to the estimated goodness. gGlOSS needs two matrices of information in order to create its database rankings.

1. The **F** matrix contains the document frequency for every term-database pair. The document frequency of a term in a database is the number of documents in the database that contain the term.
2. The **W** matrix contains, for every term-database pair, the sum of the weights of each term over all the documents in the database, where the weight of a term with respect to a document is generally a measure of the similarity between the term and the document.

Gravano and Garcia-Molina define Ideal(*t*) as the desired ranking of databases for a given query at threshold *t*. French et al. [6] discussed the use of Ideal(0) to represent gGlOSS in comparisons of database selection techniques. In this work, we also use Ideal(0) as the gGlOSS representative.

The collection retrieval inference network (CORI) selection technique [3] uses the following information to rank databases:

1. The **F** matrix and information derivable from it.
2. The number of words (*cw*) in each collection.

2.2 Testbeds

We evaluate our new *df* and *ctf* database selection approaches using three testbeds based on the TIPSTER data used in TREC [11] conferences. These testbeds — UBC-100, SYM-236 and UDC-236[1] — have been used in previous experiments to measure the impact of database selection on document retrieval effectiveness [15]. They are defined as follows.

- The UBC-100 (Uniform Byte Count) testbed contains 100 databases, each with approximately 30 megabytes of data.
- The SYM-236 (Source Year Month) testbed contains 236 databases, and is a decomposition based on the primary source and the month and year of publication of the documents.
- The UDC-236 (Uniform Document Count) testbed contains 236 databases, each with approximately 2900 documents.

We used the Concepts field of the TREC topics 51-150 to create the test queries.

2.3 Effectiveness of Different Selection Techniques

For this work, we are concerned with the ability of a database selection technique to approximate a ranking that represents the desired database selection behaviour (referred to as a *baseline*). We use three performance metrics, R_n, $\hat{R}_n$ and P_n — the recall and precision analogues used in previous work [7, 9] and described in detail in [5] — to measure the degree to which a database selection approach approximates a baseline.

In this work, we want to determine the degree to which a database selection approach can locate databases containing relevant documents. We represent the desired database selection behavior using the RBR (Relevance Based Ranking) baseline, in which databases are ranked in decreasing order of the number of documents relevant to each query Q. We construct the RBR baseline ranking from the relevance judgements that are supplied with the TREC topics. For reference, we also use a simple heuristic ranking, the SBR (Size Based Ranking) baseline, to serve as an operational lower bound for performance. SBR simply orders databases in decreasing order of the number of documents per database.

In previous studies [7], it has been demonstrated that gGlOSS estimates Ideal(0) quite well, but is not very effective when compared against the RBR baseline. It has also been shown that CORI is more effective than gGlOSS when compared against the RBR baseline [6]. In this work, we add *df*- and *ctf*-based database selection approaches to the comparison.

3 Database Selection Using *Df* and *Ctf* Information

The *document frequency* of a term in a database is defined as the number of documents in which the term occurs in the database. The *collection term frequency* of a term in a database is the number of times the term occurs in all the documents in the database, i.e. the sum of term frequency values over all documents in the database. In our selection algorithms, we use two factors to estimate the relevance of a database to a query term: *df* proportion and *ctf* proportion. In other words, the estimated relevance of database DB_i to a query term t_j depends on:

1. $dfproportion_{ij} = df_{ij} / (df_{1j} + df_{2j} + ... + df_{Nj})$
2. $ctfproportion_{ij} = ctf_{ij} / (ctf_{1j} + ctf_{2j} + ... + ctf_{Nj})$

where df_{ij} is the document frequency of query term t_j in database DB_i, ctf_{ij} is the collection term frequency of query term t_j in database DB_i, and N is the number of databases.

Intuitively, of all the documents that contain a given query term, if database DB_i has a bigger fraction of such documents than database DB_j, then we estimate that database DB_i is more related to the given term than database DB_j. Similarly, of all the occurrences of a given query term (over all documents over all databases), if database DB_i has a bigger fraction of such occurrences than database DB_j, then we estimate that database DB_i is more related to the given term than database DB_j.

1. A description of these testbeds can be found at http://www.cs.virginia.edu/~cyberia/testbed.html.

4 Experiments on the UBC-100 Testbed

We first studied the SUM and PROD database selection algorithms. For a query Q containing terms t_1, t_2, ..., t_q, and a set of databases $\{DB_1, DB_2, ..., DB_N\}$, the SUM algorithm estimates the merit of database DB_i to Q (*usefulness* of DB_i) as: $\sum_{j=1}^{q} \{tf_j \times \langle dfproportion_{ij} + ctfproportion_{ij} \rangle\}$, and PROD estimates the merit of database DB_i to Q as: $\sum_{j=1}^{q} \{tf_j \times \langle dfproportion_{ij} \times ctfproportion_{ij} \rangle\}$

where tf_j is the term frequency of term t_j, that is, the number of times term t_j occurs in query Q.

Figure 1 shows the database selection performance of RBR, CORI, SUM, PROD, Ideal(0) and SBR in terms of recall analogues R_n and $\hat{R}_n$, and precision analogue P_n. The SBR algorithm ranks databases in decreasing order of their size (the number of documents they contain). The results show that both SUM and PROD perform better than Ideal(0), but worse than CORI. In all cases, the selection effectiveness of SUM is better than that of PROD.

In an attempt to isolate the effects of the two components — *df* proportion and *ctf* proportion — we broke SUM into the DFPROP and CTFPROP selection algorithms. The DFPROP algorithm estimates the merit of database DB_i as $\sum_{j=1}^{q} \{tf_j \times dfproportion_{ij}\}$ The CTFPROP algorithm estimates the merit of DB_i as $\sum_{j=1}^{q} \{tf_j \times ctfproportion_{ij}\}$

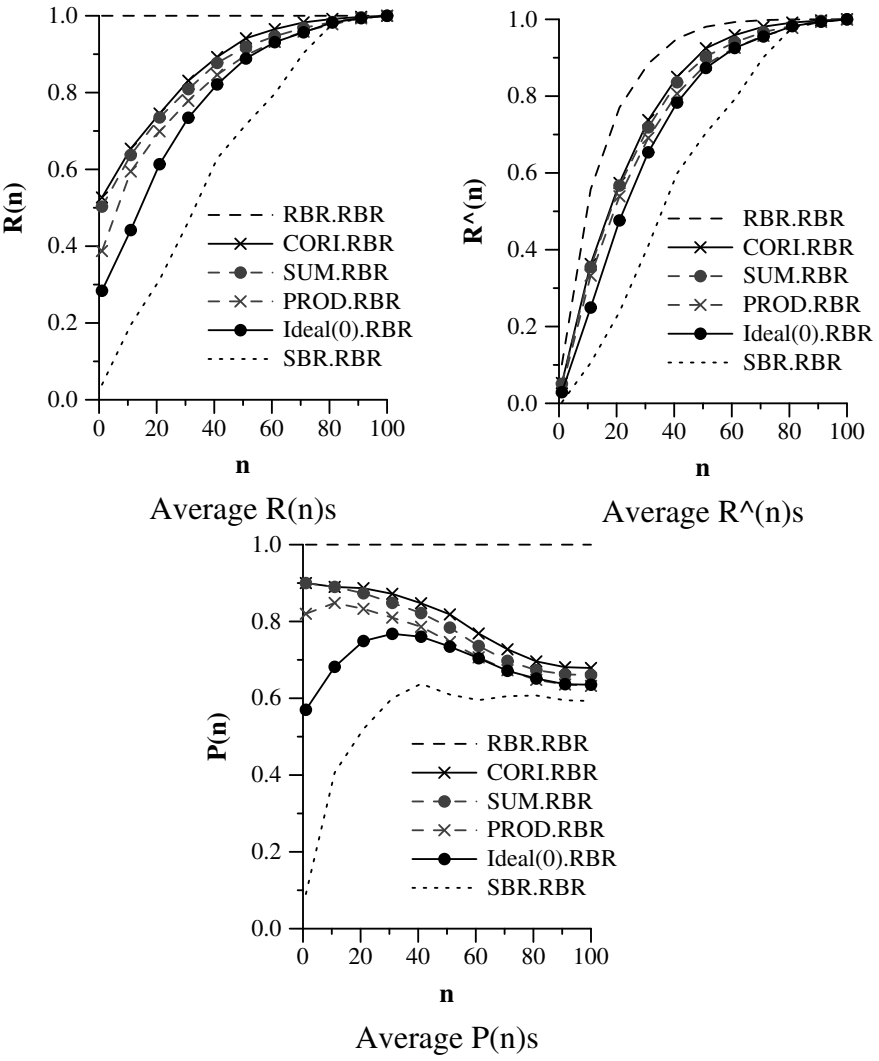

Figure 1: Performance of SUM and PROD

Figure 2 shows the database selection performance of RBR, CORI, DFPROP, SUM, CTFPROP and PROD. The results show that the performance of CTFPROP is similar to that of PROD, and that the performance of DFPROP is better than that of SUM, but slightly worse than that of CORI. These results indicate that the *df* proportion compo-

nent should be given a higher weight than the *ctf* proportion component, in order to achieve more effective selection.

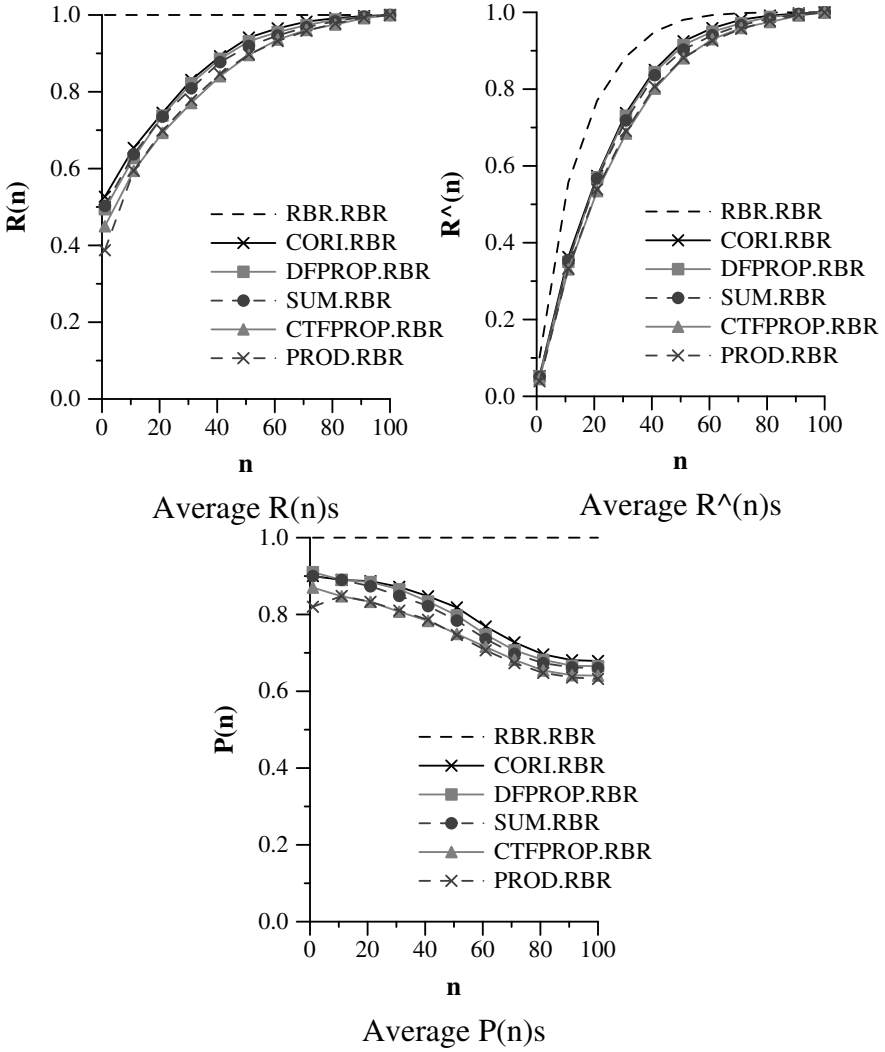

Figure 2: Performance of DFPROP and CTFPROP

Our next experiment was designed to determine whether adding a small *ctf* proportion component improves the effectiveness of selection. We implemented the CTF20 algorithm, which estimates the merit of database DB_i as $\sum_{j=1}^{q} \{tf_j \times (0.8 \times dfproportion_{ij} + 0.2 \times ctfproportion_{ij})\}$.

Figure 3 shows the database selection performance of RBR, CORI, DFPROP, CTF20 and SUM. Table 1 shows the $R\hat{}_n$ values for the different selection algorithms. The results show that the performance of CTF20 is slightly better than that of DFPROP at some observation points, but slightly worse than that of DFPROP at other observation points. Therefore, the addition of a small *ctf* proportion component as described above did not improve

the effectiveness of database selection in a noticeable manner.

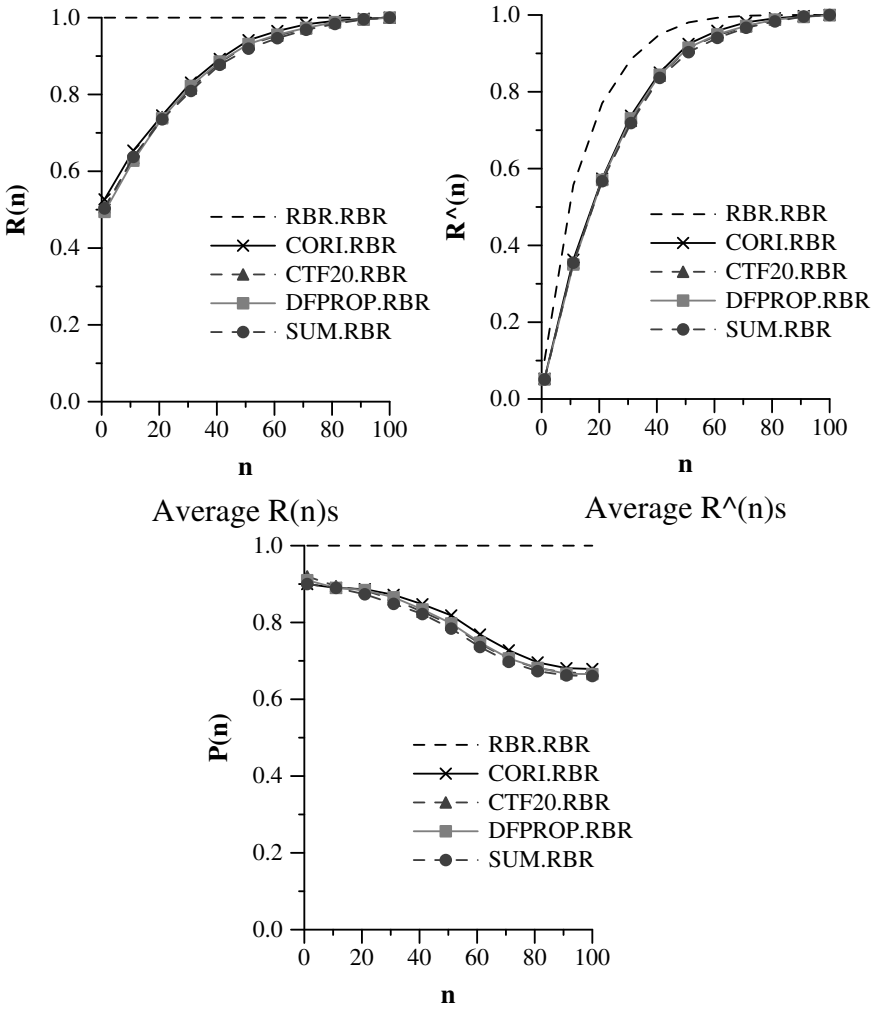

Figure 3: Performance of CTF20

n	RBR	CORI	DFPROP	CTF20	SUM
1	0.101	0.053	0.051	0.052	0.051
11	0.557	0.363	0.350	0.354	0.355
21	0.770	0.574	0.570	0.572	0.567
31	0.885	0.738	0.731	0.724	0.719
41	0.950	0.850	0.844	0.842	0.836
51	0.981	0.925	0.915	0.913	0.903
61	0.993	0.959	0.950	0.947	0.940
71	0.998	0.981	0.972	0.970	0.967

n	RBR	CORI	DFPROP	CTF20	SUM
81	1.000	0.991	0.988	0.988	0.983
91	1.000	0.997	0.995	0.996	0.996
100	1.000	1.000	1.000	1.000	1.000

Table 1: $\hat{R}_n$ values for CTF20

CORI uses an icf (inverse collection frequency) component to do database selection. The icf component is a form of global information (information over all databases). DFPROP also uses a form of global information — the ($df_{1j} + df_{2j} + ... + df_{Nj}$) component used to calculate $dfproportion_{ij}$. In our next experiment, we studied the effect of adding an icf component to DFPROP, thus emphasizing the global information component. The DFPROP-ICF algorithm estimates the merit of database DB_i as $\sum_{j=1}^{q} \{tf_j \times \langle dfproportion_{ij} \times icf_j \rangle\}$

where icf_j is the inverse collection frequency of term t_j. The *collection frequency* of term t_j is denoted by cf_j, and is the number of collections (databases) in which t_j occurs. If the number of collections is N, $icf_j = (\log(N+1))/(cf_j)$.

Figure 4 shows the database selection performance of RBR, CORI, DFPROP, DFPROP-ICF and SUM. The results show that the performance of DFPROP-ICF is worse than that of DFPROP.

Of all the selection algorithms studied above, CORI performs the best, followed by DFPROP. The performance of Ideal(0) is consistently worse than that of DFPROP. In order to test the significance of these performance differences, we performed a paired Wilcoxon signed rank test on the R_n values. The results showed that CORI was significantly better than DFPROP at 79 out of the 100 observation points, and that DFPROP was significantly better than Ideal(0) at 99 observation points.

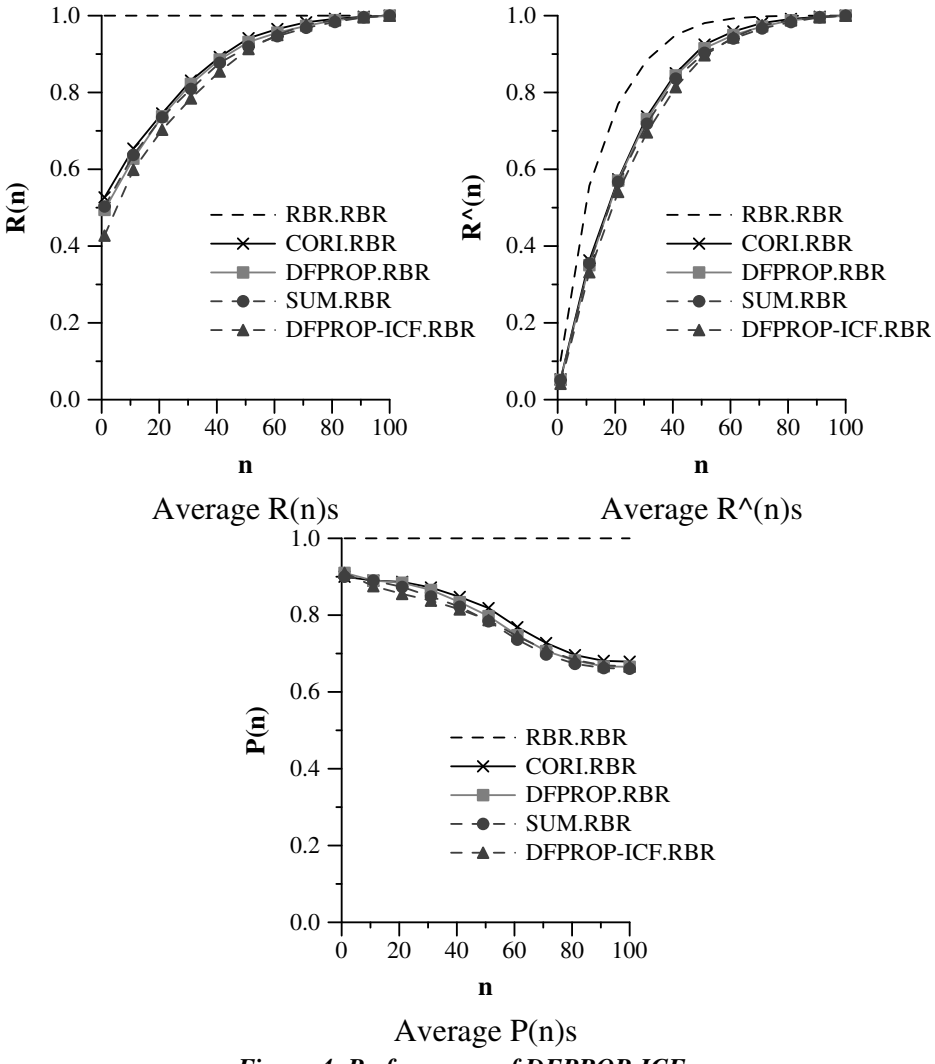

Figure 4: Performance of DFPROP-ICF

5 Performance on the SYM-236 Testbed

The SYM-236 testbed contains 236 databases organized by source, year and month. Figure 5 shows the database selection performance of RBR, CORI, DFPROP, Ideal(0) and SBR. Again, DFPROP performs better than Ideal(0), but worse than CORI. The Wilcoxon significance tests on the R_n values show that CORI is significantly better than DFPROP at 96 out of 236 observation points. DFPROP is significantly better than Ideal(0) at 111 observation points, and Ideal(0) is significantly better than DFPROP at 2 observation points.

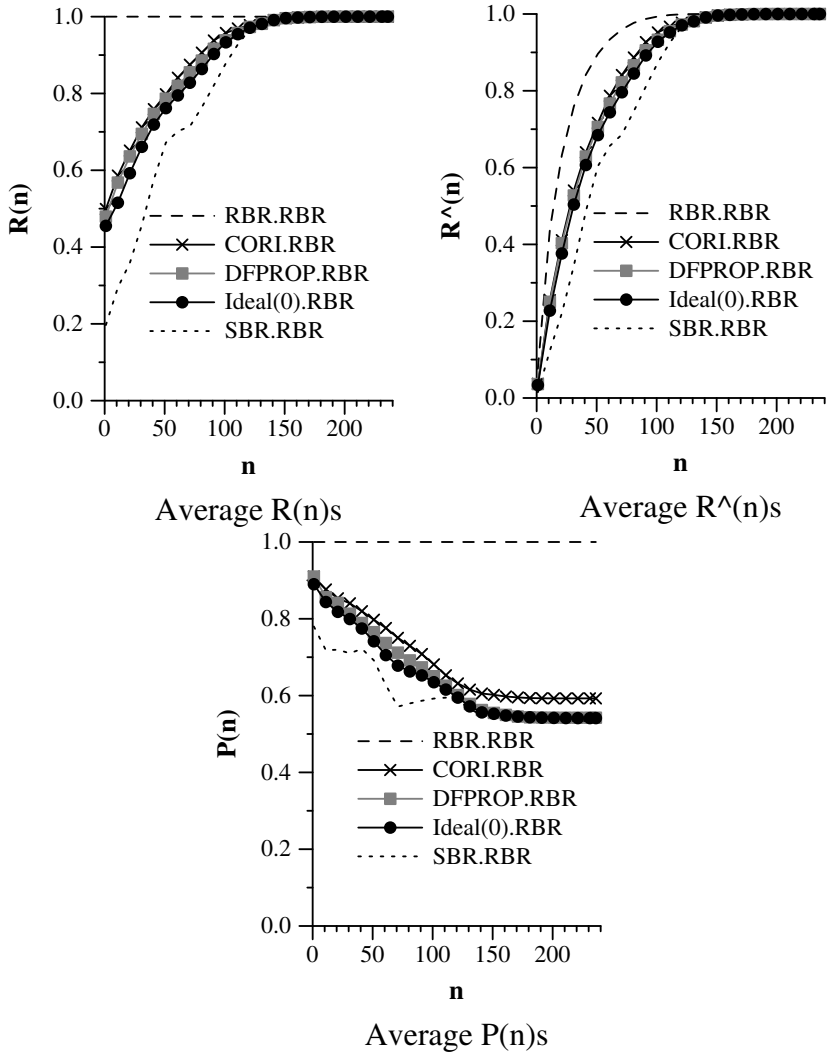

Figure 5: Performance of DFPROP on SYM-236

6 Performance on the UDC-236 Testbed

The UDC-236 testbed contains 236 databases with approximately the same number of documents. Figure 6 shows the database selection performance of RBR, CORI, DFPROP and Ideal(0). DFPROP performs worse than CORI, and better than Ideal(0), except when the number of databases selected is very small, where Ideal(0) performs better than DFPROP. The Wilcoxon significance tests on the R_n values show that CORI is significantly better than DFPROP at 206 out of 236 observation points. DFPROP is significantly better than Ideal(0) at 183 observation points, and Ideal(0) is significantly better than DFPROP at 1 observation point.

We hypothesize that the reason for the difference in performance between DFPROP and CORI (especially when a small number of databases are selected) is that DFPROP has a bias towards databases that contain long documents. The UDC-236 testbed includes two databases — PATN.01 and PATN.02 — that are composed of very long documents. A long document is likely to have a bigger vocabulary than a shorter one, and consequently, the probability of a given term occurring in the long document is higher than the probability of the term occurring in the shorter document. We hypothesize that longer documents are likely to contain query terms without actually being relevant to the query. Such a scenario would cause DFPROP to rank databases with long documents higher, while CORI might avoid this pitfall because of its *cw* (collection words) component. We examined the DFPROP and CORI results, and found that for 28% of the queries, DFPROP ranked PATN.01 among the top 5 databases, even though PATN.01 had no documents that were relevant to the query. In contrast, CORI ranked PATN.01 among the top 5 databases for only 4% of the queries. For 29% of the queries, DFPROP ranked PATN.02 among the top 5 databases, even though PATN.02 had no documents that were relevant to the query. CORI ranked PATN.02 among the top 5 databases for only 3% of the queries. In the case of CORI, only 6% of the queries were affected by having PATN.01 or PATN.02 or both, among the top 5 databases. In the case of DFPROP, 34% of the queries were affected by having PATN.01 or PATN.02 or both, among the top 5 databases. These results seem to support our hypothesis that DFPROP performs worse than CORI because of a size bias towards databases with longer documents.

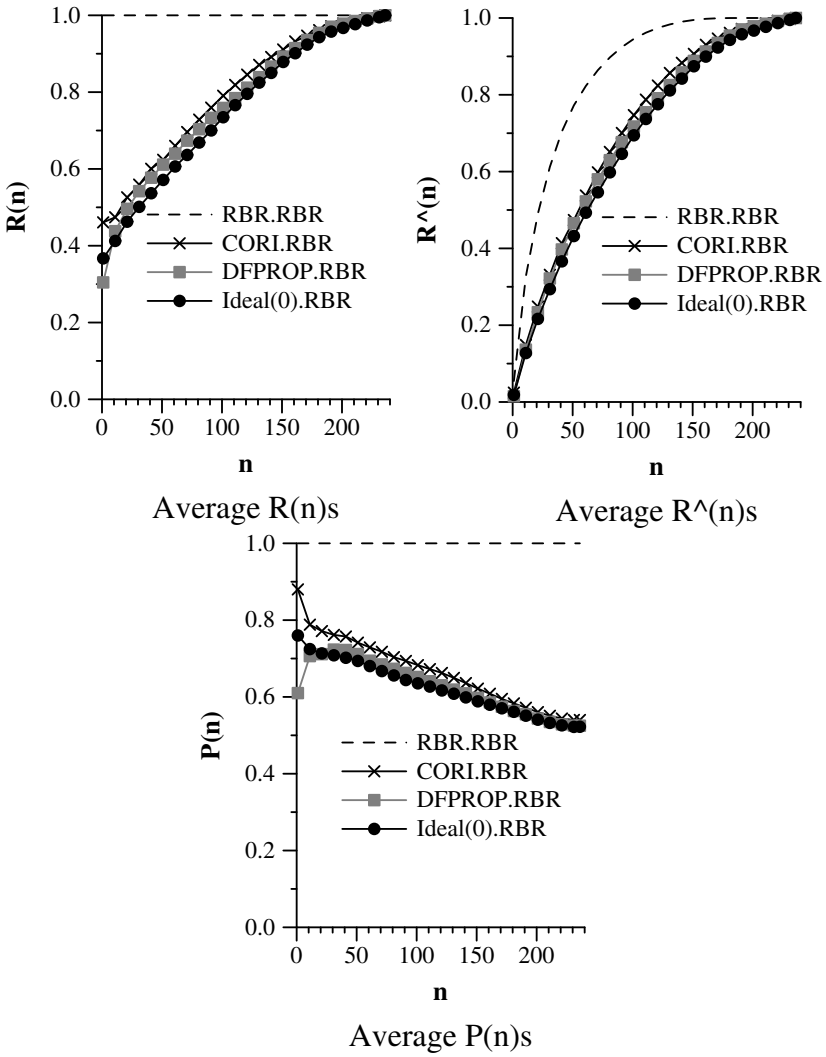

Figure 6: Performance of DFPROP on UDC-236

7 Conclusions

We have introduced a new database selection algorithm — DFPROP — that uses only document frequency information in the form of *df* proportions. DFPROP uses less information than Ideal(0), but performs better on all three TREC testbeds that we used. Unlike gGlOSS, the information used by DFPROP does not depend on the indexing strategy of the individual databases. DFPROP also uses less information than CORI, and achieves a large portion of the performance benefit of CORI.

We have demonstrated that our simple DFPROP algorithm works well for three different types of testbeds. We hypothesize that the gap between the performance of CORI and DFPROP is because of DFPROP's bias towards databases with long documents. In future work, we hope to discover the cause of this performance gap, and to explore the effects of adding information that will improve the selection effectiveness of our algorithm.

8 References

[1] C. Baumgarten, A Probabilistic Model for Distributed Information Retrieval, *Proceedings of the 20th International Conference on Research and Development in Information Retrieval*, pg. 258-266, 1997.

[2] C. Baumgarten, A Probabilistic Solution to the Selection and Fusion Problem in Distributed Information Retrieval, *Proceedings of the 22nd International Conference on Research and Development in Information Retrieval*, pg. 246-253, 1999.

[3] J. P. Callan, Z. Lu and W. B. Croft, Searching Distributed Collections with Inference Networks, *Proceedings of the 18th International Conference on Research and Development in Information Retrieval*, pg 21-29, 1995.

[4] N. Craswell, P. Bailey and D. Hawking, Server Selection on the World Wide Web, *Proceedings of the Fifth ACM Conference on Digital Libraries,* pg. 37-46, 2000.

[5] J. C. French and A. L. Powell, Metrics for Evaluating Database Selection Techniques, *World Wide Web*, 3(3), 2000.

[6] J. C. French, A. L. Powell, J. Callan, C. L. Viles, T. Emmitt, K. J. Prey and Y. Mou, Comparing the Performance of Database Selection Algorithms, *Proceedings of the 22nd International Conference on Research and Development in Information Retrieval,* pg. 238-245, 1999.

[7] J. C. French, A. L. Powell, C. L. Viles, T. Emmitt and K. J. Prey, Evaluating Database Selection Techniques: A Testbed and Experiment, *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pg 121-129, August 1998.

[8] N. Fuhr, A Decision-Theoretic Approach to Database Selection in Networked IR, *ACM Transactions on Information Systems*, 17(3):229-249, 1999.

[9] L. Gravano and H. Garcia-Molina, Generalizing GlOSS to Vector-Space Databases and Broker Hierarchies, *Proceedings of the 21st International Conference on Very Large Databases (VLDB)*, 1995.

[10] L. Gravano, H. Garcia-Molina and A. Tomasic, The Effectiveness of GlOSS for the Text Database Discovery Problem, *ACM SIGMOD 94*, pg 126-137, 1994.

[11] D. Harman, Overview of the 4th Text Retrieval Conference (TREC-4), *Proceedings of TREC-4*, 1996.

[12] D.Hawking and P. Thistlewaite, Methods for Information Server Selection, *ACM Transactions on Information Systems*, 17(1):40-76, 1999.

[13] W. Meng, K.-L. Liu, C. Yu, X. Wang, Y. Chang and N. Rishe, Determining Text Databases to Search in the Internet, *Proceedings of the 24th VLDB Conference*, pg. 14-25, 1998.

[14] {W. Meng, K.-L. Liu, C. Yu, W. Wu and N. Rishe, Estimating the Usefulness of Search Engines, *15th International Conference on Data Engineering*, pg. 146-153, 1999.

[15] A. L. Powell, J. C. French, J. Callan, M. Connell and C. L. Viles, The Impact of Database Selection on Distributed Searching, *Proceedings of the 23rd International Conference on Research and Development in Information Retrieval*, pg. 232-239, 2000.

[16] J. Xu and J. Callan, Effective Retrieval with Distributed Collections, *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pg. 112-120, August 1998.

[17] J. Xu and W. B. Croft, Cluster-based Language Models for Distributed Retrieval, *Proceedings of the 22nd International Conference on Research and Development in Information Retrieval*, pg. 254-261, 1999.

[18] C. Yu, K.-L. Liu, W. Wu, W. Meng and N. Rishe, Finding the Most Similar Documents across Multiple Text Databases, *Proceedings of the IEEE Conference on Advances in Digital Libraries (ADL'99)*, pg. 150-162, 1999.

[19] C. Yu, W. Meng, K.-L. Liu W. Wu and N. Rishe, Efficient and Effective MetaSearch for a Large Number of Text Databases, Proceedings of the Eighth International Conference on Information and Knowledge Management, pg. 217-224, 1999.

[20] B. Yuwono and D. L. Lee, Server Ranking for Distributed Text Retrieval Systems on Internet, Proceedings of the Fifth International Conference on Database Systems for Advanced Applications, pg. 41-49, 1997.