Published in Lecture Notes in Computer Science 2633, 1611-3349, 2003
which should be used for any reference to this work

Term Proximity Scoring for Keyword-Based Retrieval Systems

Yves Rasolofo and Jacques Savoy

Université de Neuchâtel, Neuchatel, Switzerland
{yves.rasolofo,jacques.savoy}@unine.ch

Abstract. This paper suggests the use of proximity measurement in combination with the Okapi probabilistic model. First, using the Okapi system, our investigation was carried out in a distributed retrieval framework to calculate the same relevance score as that achieved by a single centralized index. Second, by applying a term-proximity scoring heuristic to the top documents returned by a keyword-based system, our aim is to enhance retrieval performance. Our experiments were conducted using the TREC8, TREC9 and TREC10 test collections, and show that the suggested approach is stable and generally tends to improve retrieval effectiveness especially at the top documents retrieved.

1 Introduction

When Web users submit requests to search engines, they expect to retrieve highly relevant Web pages. This presents quite a challenge, especially when search engine queries are rather short. Studies by Spink *et al.* [16][27] have shown that the average query length is between two and three keywords. Moreover, these requests tend to cover a rather wide variety of information needs, and are often expressed with ambiguous terms. Their study [27] demonstrates also that users expect the system to retrieve relevant documents at the top of the result list. Indeed, more than 50% of Web users tend to consult only the first 2 result pages. Besides of looking for relevant pages, a great number of Web users search for online service location or homepages. As for the other types of queries, the page must to be in the top of the list. A new task handling these kinds of requests was introduced in TREC-10 (Text Retrieval Conference). This search mechanism allows users to submit a request such as "Quantas" in order to retrieve the Quantas Airlines homepage, not several other Web pages referring to this airline company. Previous efforts to resolve this approach based on content only were not really effective [6]. During TREC-10 and as reported in [14], systems performed best when, in addition to information taken from document content, URL texts and/or URL depths were used. Another recent study by Singhal & Kaszkiel [26] involving analogous query types came to similar conclusion. Indeed, as suggested in [7], if one is to improve retrieval effectiveness combining approaches now seems imperative.

We introduce in the first part of this study a simple approach to address the problem of distributed index. The advantage of this approach is its ability to provide retrieval status scores (RSV) as if documents were searched using a single centralized index. Nevertheless, the proposed approach requires that each index use the same scheme. It means that during the document indexing, the term weight is calculated based only on document statistics or local information (e.g. term frequency, document length, …). Collection statistics (e.g. document frequency) are computed only when user's request is submitted to the system. Thus our approach can be used in the digital library context within which various collections are indexed and searched using the same retrieval system. It is also possible to apply our approach locally to a particular commercial search engine, as it normally uses the same indexing and search model when dealing with various inverted files. However, the proposed approach does not apply when metasearching various search engines using different indexing and retrieval techniques [20].

In an effort to improve retrieval effectiveness at the top ranked items, in the second part of this work we propose an enhanced search strategy that can help to resolve both Web-page and online service location searches. We will thus focus on adding a word-pair scoring module to our implementation of the keyword-based Okapi system. This study is based on previous approaches where phrase, term proximity or term distance were used in information retrieval.

Various phrase-finding and indexing methods have been proposed in the past and generally, retrieval performance conclusions on the use of phrases as indexing units were inconsistent. Salton & McGill [23] suggested generating statistical phrases based on word co-occurrence and then incorporating them into document representation as additional index terms. Fagan [10][11] evaluated this method by combining words-based and phrase-based weighting. In his study [10], he considered syntactic phrase information, generated from considering syntactic relations or syntactic structures. This seemed to enhance retrieval performance marginally while statistical phrase discovery approaches seem to produce better results. More recently, leading groups including [1], [3], [9], [15] and [28], who have participated in TREC campaigns, used phrases as indexing units and were able to obtain some improvement. Mitra *et al.* [17] re-examined the use of statistical and syntactic phrases and came to the conclusion that "once a good basic ranking scheme is used, the use of phrases do not have a major effect on precision at high ranks." Finally, Arampatzis *et al.* [2] came up with some possible reasons for the lack of success when using Natural Language Processing (NLP) techniques in information retrieval, particularly when using syntactic phrases. They stated that "first, the currently available NLP techniques suffer from lack of accuracy and efficiency and second, there are doubts if syntactic structure is a good substitute for semantic content."

The work of Hawking & Thistlewaite [12] is more directly related to ours, and they explored the use of proximity scoring within the PADRE system. Their Z-mode method has the advantage of being totally independent of collection statistics and in distributed information retrieval it represents a good solution for merging result lists[1].

[1] Result lists merging or collection merging is one of the critical issues of distributed information retrieval [19].

Clarke *et al.* [5] developed a similar technique and obtained results worthy of consideration. Papka & Allan [18] extracted multiword features from documents using certain proximity measures and obtained some retrieval performance success by using these multiword features for massive query expansions.

The rest of this paper presents our approach based on term proximity scoring. We will use the word-based Okapi system as a baseline for comparisons, and evaluate our approach using TREC-8, TREC-9 and TREC-10 test collections. The main objectives of our work in this paper are the following:

- to propose a framework manipulating the use of distributed indexes in order to cope with index size limitation,

- to determine whether or not simple term proximity scoring can improve retrieval effectiveness: we do not intend to compare existing term proximity approaches with ours. Our baseline will therefore be a word-based system using no proximity measures and we will measure to what extent our term proximity scoring can improve this system in terms of retrieval effectiveness.

In the next section, we will discuss the Okapi probabilistic model and examine its appropriateness within distributed index frameworks. Term pair scoring is presented in Section 3, and finally Section 4 describes our experiments and results.

2 OKAPI and Distributed Index Frameworks

Okapi is an enhanced probabilistic retrieval model based on the binary independence model proposed by Robertson & Spark Jones [21]. By incorporating term frequency and document length considerations, the Okapi model was able to demonstrate interesting retrieval performances during the last TREC campaigns [22]. In this paper, we will use a simplified Okapi weighting function, in which the weight w_i was assigned to a given term t_i in a document d and was computed according to the following formula:

$$w_i = (k_1 + 1) \cdot \frac{tf_i}{K + tf_i} \tag{1}$$

where:

$$K = k \cdot \left[(1 - b) + b \cdot \frac{l}{avdl} \right]$$

l is the document length,
$avdl$ is the average of document length (set to 750),
b is a constant (set to 0.9),
k is a constant (set to 2),
k_1 is a constant (set to 1.2),
tf_i is the occurrence frequency of the term t_i in document d.

The following formula shows the weight given to the same term t_i within a query:

$$qw_i = \frac{qtf_i}{k_3 + qtf_i} \cdot \log\left(\frac{n - df_i}{df_i} \right) \tag{2}$$

where:

qtf_i is the frequency of term t_i in the query,
df_i is the number of documents in the collection containing the term t_i,
n is the number of documents included in the collection.
k_3 is a constant (set to 1000).

The retrieval status value is then calculated as follows:

$$RSV_{Okapi}(d,q) = \sum_i w_i \cdot qw_i$$

During the last three years, the TREC adhoc task (Web track) has been based on test collections containing more than 10 GB of Web pages. Since creating a single inverted file from a collection of this size might be impossible within a 32-bit system (e.g., Linux), we suggest that one way to resolving this problem would be the use of index pruning [4] or distributing inverted files [24]. In both cases there might be some risk of decreasing retrieval performance depending on how the index pruning or the index distributions are implemented. In this paper, our only interest is the second approach, and in this case, we could follow the approach suggested by [24], where result lists obtained from searching different collections are merged. This is achieved by using document scores computed by each collection (collections are searched using the same retrieval scheme). Dumais [8] mentioned however that various statistics are collection dependant (e.g., the idf values) and that these values may vary widely across collections. The resultant document scores might not therefore be directly comparable.

The Okapi weighting function imparts interesting characteristics within distributed collection frameworks. Equation 1 shows how document term weight is based on within-document term frequency and document length only, while the search keyword weight as depicted in Equation 2 uses collection dependant statistics (namely, the *idf* value). Those characteristics facilitate data exchange in a distributed information retrieval (DIR)[2] environment when collections are indexed and searched using the same retrieval scheme. Indeed, when the system indexes documents, it gives term weight independently of this term's collection frequency. Collection frequencies need only be exchanged during query processing. Thus Equation 2 becomes:

$$qw_i = \frac{qtf_i}{k_3 + qtf_i} \cdot \log\left(\frac{N - DF_i}{DF_i}\right) \qquad (3)$$

where:

N is the sum of documents within all collections,

[2] In a distributed information retrieval environment, documents are distributed across various collections which may be searched using the same or different retrieval schemes.

DF_i is the number of documents containing the term t_i within all collections. Each collection sends the local collection frequency (df_i) to the broker[3] on request, qtf_i query term frequency.

As in [13], this process needs to exchange data between the broker and the collections involved. The amount of data exchanged during this process equals $Nt \cdot Nc$, for both the request of collection statistics and the response reception. In this formula, Nt denotes the number of query terms and Nc the number of collections. Therefore in order to perform the search within this proposed framework, the broker must weight query terms using Equation 3 and then send this information to the collections. Following this step, document scores returned by collections are directly comparable because they are based on the same collection statistics [8].

3 Term Proximity Weighting

Efficiency and effectiveness are critical concerns for users, for they expect the search engine to return only what they need and as quickly as possible. With our proposed approach, we expect improvement at the top ranks, in a computationally tractable manner. We thus decided to add the suggested extension to Okapi because we wanted to use an effective keyword-based search model and we hoped to improve its retrieval performance by using term proximity scoring.

Our approach is able to cope with multi-term queries and is based on the assumption that if a document contains sentences having at least two query terms within them, the probability that this document will be relevant must be greater. Moreover, the closer are the query terms, the higher is the relevance probability. Most of the time, search engine users submit a query which is a concept (e.g. "lung cancer"), a proper name (e.g. "Buck Danny"), a place name (e.g. "Pisa Tower") or other types of queries where terms are likely to be found in a narrow context or even adjacent within a relevant document. It is then important to assign more importance to those keywords having a short distance between their occurrences, under the assumption that if the distance increases, the underlying meaning may change.

To achieve this objective we will expand the request using keyword pairs extracted from the query's wording. We will assume that queries are short and that users will only write relevant terms. If stopwords appear in the request's formulation, then they will be automatically removed. Finally, the process described below is applied only for queries having more than one keyword.

First, we establish a set of all possible search keyword pairs. If the query wording consists of $q = (t_i, t_j, t_k)$, we obtain the following set S of term pairs: $\{(t_i, t_j), (t_i, t_k), (t_j, t_k)\}$, with the ordering of terms not being important. During the indexing process, we create an inverted file containing the occurrence positions of each term in each document. The term pair retrieval within a given document is performed by sequen-

[3] A broker is an interface between the user and the collections. It is responsible for receiving the user's request, sending this query to various selected collections, merging the result lists provided by each selected collection and returning a single list to the user.

tially reading the query term positions, and for each instance the term pair (t_i, t_j) within a maximal distance of five (or having a maximal of four terms between the keyword pair), we compute a term pair instance (*tpi*) weight as follows:

$$tpi(t_i, t_j) = \frac{1.0}{d(t_i, t_j)^2}$$

where $d(t_i, t_j)$ is the distance expressed in number of words between search term t_i and t_j.

Our hypothesis is that the closer two search keywords appear together within a document, the higher is the weight attached to the occurrence of this term pair. Based on this formulation, the higher value is 1.0, corresponding to a distance of one (the terms are adjacent), and the lower value is 0.04 corresponding to a distance of 5. For example, based on the request "information retrieval", the resulting *tpi* of an occurrence of the same string "information retrieval" will be 1.0 while the *tpi* of "the retrieval of medical information" will be 1/9 or 0.11.

Of course, a given term pair may appear more than once in a document. Therefore, the weight attached to this given term pair (t_i, t_j) is evaluated by summing all the corresponding term pair instances *tpi*. In a manner similarly to Equation 1, we obtain:

$$w_d(t_i, t_j) = (k_1 + 1) \cdot \frac{\sum_{occ(t_i, t_j)} tpi(t_i, t_j)}{K + \sum_{occ(t_i, t_j)} tpi(t_i, t_j)}$$

Given a document d and a request q, we compute the contribution of all occurring term pairs in that document. This value, denoted *TPRSV*, and based on the set S of query term pairs included in the request q, is evaluated using the following formula:

$$TPRSV(d, q) = \sum_{(t_i, t_j) \in S} w_d(t_i, t_j) \cdot \min(qw_i, qw_j)$$

where qw_i and qw_j are the weights of the query terms t_i and t_j calculated according to eq. 3

This *TPRSV* is only calculated for the top 100 documents returned by the Okapi search model. This choice is motivated by efficiency needs and our main interest is to achieve improvement at top ranks. The final retrieval status value for a given document d, denoted $RSV_{NEW}(d,q)$, is computed as follows:

$$RSV_{NEW}(d, q) = RSV_{Okapi}(d, q) + TPRSV(d, q)$$

This formulation accounts for both the original Okapi score (RSV_{Okapi}) and our proximity scoring function (*TPSRSV*). During this process no new document is retrieved, as it is performed on the top 100 documents retrieved by Okapi. Instead, the scores and therefore the ranks of documents containing at least one query term pair are improved based on the following assumption: The presence of query terms within a document would not always imply a match related to the true meaning of the request,

whereas account for search keyword pairs using some distance constraint may reduce this error. Using our approach and in response to the request "operating system", a document containing theses two terms close each other will be presented to the user before any other documents having these two terms within two different paragraphs.

4 Experiments

4.1 Test Collections

Experiments were conducted based on 150 topics used for the adhoc task for TREC-8, TREC-9 and TREC-10. The TREC-8 collection contains 528,155 documents (representing 1,904 MB) extracted from four different sources, namely *Financial Times* (FT, 210,158 documents), *Federal Register* (FR, 55,630 documents), *Foreign Broadcast Information Service* (FBIS, 130,471 documents) and *Los Angeles Times* (LA Times, 131,896 documents). An assessed set of 50 topics was provided, covering a rather broad range of subjects, including for example "Estonia, economy," "suicides," "airport security," "osteoporosis" and "cosmic events." We decided to split the TREC-8 document collection into four collections according to their sources (FT, FR, FBIS, LA Times), with the number of documents and the collection size varying from one collection to another.

The TREC-9 and TREC-10 test collections contain the same documents, corresponding to Web pages from various sites around the world (1,692,096 Web pages with a total size of 11,033 MB). Thus, their volume is roughly six times greater than TREC-8. Both the TREC-9 and TREC-10 test collections include 50 topics, representing various information needs, and they contain a larger number of spelling mistakes. Topics originate from various domains (e.g., "Parkinson's disease," "hunger," "how e-mail benefits businesses," "Mexican food culture," "Titanic went wrong" and "history of Cambodia"). In order to simulate a distributed environment, we divided this Web page collection into four separate collections, each having roughly the same number of pages and same size.

As our approach deals with search keyword pairs, we removed queries containing a single word. The evaluations were performed on 62 two-words queries, 55 with three search terms, 7 with four and 1 with five terms (see Table 1).

Table 1. Query length statistics after stopword removal

Collection	2 words	3 words	4 words	5 words	Average
TREC-8	25	22	0	0	2.47
TREC-9	18	15	3	0	2.58
TREC-10	19	18	4	1	2.69
Sum	62	55	7	1	2.58

4.2 Evaluations

Table 2 depicts various retrieval performances resulting from the use of our models. In the first column, we indicate the test collection ("TREC8", "TREC9", "TREC10" or "All" when considering the three collections as a whole) and the search model used ("Okapi", and "OkaTP"), with the "OkaTP" rows showing performances achieved by our term proximity scoring. In the second column, we compute the average precision using the TREC-EVAL software and in the following column lists precision levels achieved after retrieving 5 (P@5), 10 (P@10) and 20 (P@20) documents.

In terms of average precision (column two), differences between our enhanced model and Okapi scheme are fairly small. This can be explained as follows: our proximity algorithm takes accounts for each instance of a search keyword pair within a maximal distance of 5. This constraint limits the number of documents that will have their retrieval status value increased. At the same time, this constraint discards a large number of poor term pair candidates where the distance between the two search keywords is too great (under the assumption that in such cases, there is no semantic relationship between the two keywords). Moreover we processed only the top 100 documents, while for some queries it is possible that most relevant documents were located beyond the limit of top 100 documents.

Table 3 serves as a query-by-query average precision analysis, depicting the number of requests for which the OkaTP scheme was better (+), worse (-) or showed similar (=) performance levels as did Okapi. The last column of Table 3 lists the results of Sign tests, with a significance level $\alpha = 0.05$. When considering the TREC-8 corpus (first row), comparisons show that the model (OkaTP) revealed better performance for 29 requests, worse for 15 and identical performance for three queries.

Table 2. Average precision, precision after 5, 10 and 20 documents

Collection	AvPrec	Diff.	P@5	Diff	P@10	Diff.	P@20	Diff
TREC-8-Okapi	0.2465		0.4809		0.4489		0.3979	
TREC-8-OkaTP	0.2525	2.43%	0.5021	4.41%	0.4702	4.74%	0.4000	0.53%
TREC-9-Okapi	0.2399		0.3389		0.2750		0.2014	
TREC-9-OkaTP	0.2447	2.00%	0.3500	3.28%	0.2722	-1.02%	0.2069	2.73%
TREC-10-Okapi	0.1920		0.3333		0.3143		0.2607	
TREC-10-OkaTP	0.1869	-2.66%	0.3952	18.57%	0.3442	9.83%	0.2798	7.33%
All-Okapi	0.2263		0.3904		0.3536		0.2952	
All-OkaTP	0.2282	0.84%	0.4224	8.20%	0.3712	4.98%	0.3040	2.98%

Table 3. Query-by-query average precision analysis: OkaTP vs. Okapi

Collection	+	-	=	Sign test
TREC-8	29	15	3	Okapi < OkaTP
TREC-9	19	12	5	Okapi = OkaTP
TREC-10	23	16	3	Okapi = OkaTP
Sum	71	43	11	Okapi < OkaTP

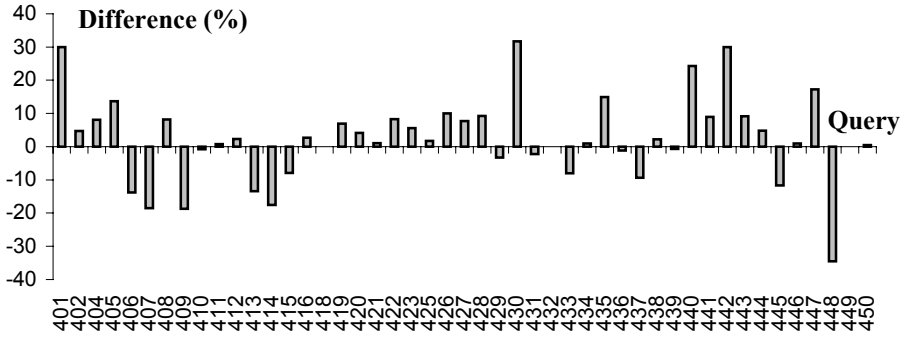

Fig. 1. TREC-8: Query-by-query differences in average precision (OkaTP-Okapi)

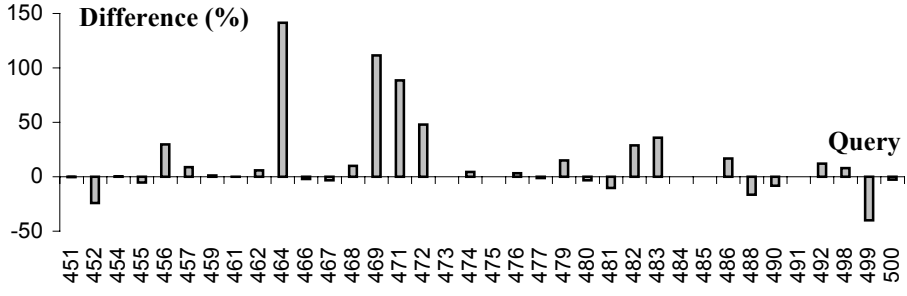

Fig. 2. TREC-9: Query-by-query average precision difference (OkaTP-Okapi)

The Sign test reports the OkaTP (Okapi + term pair scoring) significantly better than Okapi. Taken separately, the TREC-9 and TREC-10 collections do not show any statistically significant differences between the two retrieval models. But based on the last row we can conclude that we encountered significant differences when testing all three corpora, with the OkaTP tending to be better than the Okapi.

As the Sign test does not account for differences in magnitude, Figure 1, Figure 2 and Figure 3 show that for our three corpora, the enhancements in average precision are generally far greater than are the degradations.

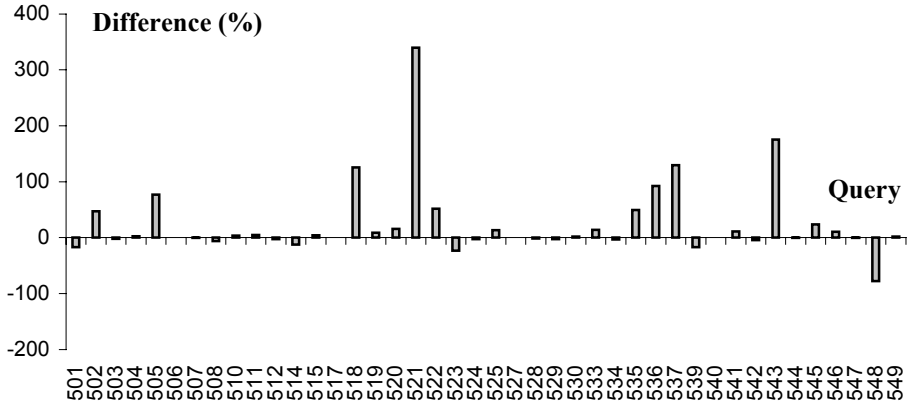

Fig. 3. TREC-10: Query-by-query differences in average precision (OkaTP-Okapi)

Our main concern however is to enhance search performances after retrieving only a few documents. To analyze this aspect, in Table 2 we reported the precision achieved by our two models after retrieving 5, 10 or 20 items. From looking at these evaluations, precision improvements obtained by the proximity measure are more noticeable after retrieving 5 documents, and in this case the overall improvement is around 8.20%. Such results would prove useful for those users looking at the top 5 or 10 documents returned [27]. Moreover, these results would also prove interesting when locating online services [14] or when examining question-answering systems [29] where most of the time, the expected answer is a single URL or a short sentence of only a few words extracted from a document.

5 Conclusion

In this paper we introduced a distributed information retrieval framework based on the Okapi probabilistic model, a framework capable of achieving the same levels of retrieval effectiveness as those achieved by a single centralized index system. Moreover, the impact of a new term proximity algorithm on retrieval effectiveness for a keyword-based system was examined. The approach we suggested seems to improve ranking for documents having query term pairs occurring within a given distance constraint. It seems the term proximity scoring approach that we proposed may potentially improve precision after retrieving a few documents and thus could prove useful for those users looking only at the top ranked items (e.g., when using search engines available on the Web) or in search systems that must provide users with a short and complete answer (e.g., homepage searching, question-answering systems).

Acknowledgement

This research was supported by the SNSF (Swiss National Science Foundation) under grant 21-58'813.99.

References

[1] Allan, J., Ballesteros, L., Callan, J.P., Croft, W.B. and Lu, Z.: Recent experiments with INQUERY. In Proceedings of TREC-4, NIST Special Publication #500-236, 49-63, 1996.

[2] Arampatzis, A., van der Weide, T., Koster, C. and van Bommel, P.: Linguistically motivated information retrieval. Encyclopedia of Library and Information Science, 39, 2000.

[3] Buckley, C., Singhal, A., and Mitra, M.: Using query zoning and correlation within SMART: TREC-5. In Proceedings of TREC-5, NIST Special Publication #500-238, 105-118, 1997.

[4] Carmel, D., Amitay, E., Herscovici, M., Maarek, Y., Petruschka, Y., and Soffer, A.: Juru at TREC-10: Experiments with index pruning. In Proceedings TREC-10, NIST Special Publication #500-250, 228-236, 2002.

[5] Clarke, C.L.A., and Cormack, G.V. and Tudhope E. A.: Relevance Ranking for One to Three Term Queries. Information Processing and Management, 36(2):291-311, 2000.

[6] Craswell, N., Hawking, D., and Robertson, S.E.: Effective site finding using link anchor information. In Proceedings SIGIR-2001, ACM Press, 250-257, 2001.

[7] Croft, W.B.: Combining approaches to information retrieval. In W.B. Croft (Ed.), Advances in information retrieval, Kluwer Academic Publishers, 1-36, 2000.

[8] Dumais, S.T.: Latent semantic indexing (LSI) and TREC-2. In Proceedings of TREC-2, NIST Special Publication, #500-215, 105-115, 1994.

[9] Evans, D.A., Milic-Frayling, N., and Lefferts, R.G.: CLARIT TREC-4 experiments. In Proceedings of TREC-4, NIST Special Publication, #500-236, 305-321, 1996.

[10] Fagan, J.: Experiments in automatic phrase indexing for document retrieval: A comparison of syntactic and non-syntactic methods. PhD thesis, Computer Science Department, Cornell University. 1987.

[11] Fagan, J.: The effectiveness of a nonsytactic approach to automatic phrase indexing for document retrieval. Journal of the American Society for Information Science, 40(2), 115-132, 1989.

[12] Hawkings, D. and Thistlewaite, P.: Proximity operators – So near and yet so far. In Proceedings of TREC-4, NIST Special Publication #500-236, 131-143, 1996.

[13] Hawking, D., and Thistlewaite, P.: Methods for information server selection. ACM Transactions on Information Systems, 17(1), 40-76, 1999.

[14] Hawking, D. and Craswell, N.: Overview of the TREC-2001 Web track. In Proceedings TREC-10, NIST Special Publication #500-250, 61-67, 2002.

[15] Hull, D.A., Grefenstette, G., Schulze, B.M., Gaussier, E., Schutze, H. and Pedersen, J.O.: Xerox TREC-5 site report: Routing, filtering, NLP, and Spanish tracks. In Proceedings of TREC-5, NIST Special Publication #500-238, 167-180, 1997.

[16] Jansen, B.J., Spink, A. and Saracevic, T.: Real life, real users and real needs: A study and analysis of user queries on the Web. Information Processing and Management, 36(2), 207-227, 2000.

[17] Mitra, M., Buckley, C., Singhal, A., and Cardie, C.: An analysis of statistical and syntactic phrases. In Proceedings of RIAO-97, 1997.

[18] Papka, R., and Allan, J.: Document classification using multiword features. In Proceedings of CIKM-98, ACM Press, 124-131. 1998.

[19] Rasolofo, Y., Abbaci, F. and Savoy, J.: Approaches to collection selection and results merging for distributed information ietrieval. In Proceedings of CIKM-2001, ACM Press, 191-198, 2001.

[20] Rasolofo, Y., Hawking, D., Savoy, J.: Result Merging Strategies for a Current News MetaSearcher. Information Processing & Management, 2003 (to appear).

[21] Robertson, S.E., and Spark Jones, K.: Relevance weighting of search terms. Journal of the American Society for Information Science, 27(3), 129-146, 1976.

[22] Robertson, S.E., Walker, S., and Beaulieu, M.: Experimentation as a way of life: Okapi at TREC. Information Processing & Management, 36(1), 95-108, 2000.

[23] Salton G., and McGill, M.J.: Introduction to modern information retrieval. McGraw-Hill, 1983.

[24] Savoy, J., and Rasolofo, Y.: Report on the TREC-10 experiment: Distributed collections and entrypage searching. In Proceedings TREC-10, NIST Special Publication #500-250, 586-595, 2002.

[25] Silverstein, C., Henzinger, M., Marais, H. and Moricz, M.: Analysis of a very large Web search engine query log. ACM SIGIR Forum, 33(1), 6-12, 1999.

[26] Singhal, A., and Kaszkiel, M.: A case study in Web search using TREC algorithms. In Proceedings of WWW'10, Elsevier, 708-716, 2001.

[27] Spink, A. Wolfram, D., Jansen, B.J., and Saracevic, T.: Searching the Web: The public and their queries. Journal of the American Society for Information Science and Technology, 52(3), 226-234, 2001.

[28] Strzalkowski, T., Guthrie, L., Karlgren, J., Leistensnider, J., Lin, F., Perez-Carballo, J., Straszheim, T., Wang, J., and Wilding, J.: Natural language information retrieval: TREC-5 report. In Proceedings TREC-5, NIST Special Publication #500-238, 291-313, 1997.

[29] Voorhees, E.M.: Overview of the TREC 2001 question answering track. In Proceedings TREC-10, NIST Special Publication #500-250, 42-51, 2002.