DOCUMENT REPRESENTATION IN NATURAL LANGUAGE TEXT RETRIEVAL

Tomek Strzalkowski

Courant Institute of Mathematical Sciences
New York University
715 Broadway, rm. 704
New York, NY 10003
tomek@cs.nyu.edu

ABSTRACT

In information retrieval, the content of a document may be represented as a collection of *terms*: words, stems, phrases, or other units derived or inferred from the text of the document. These terms are usually *weighted* to indicate their importance within the document which can then be viewed as a vector in a N-dimensional space. In this paper we demonstrate that a proper term weighting is at least as important as their selection, and that different types of terms (e.g., words, phrases, names), and terms derived by different means (e.g., statistical, linguistic) must be treated differently for a maximum benefit in retrieval. We report some observations made during and after the second Text REtrieval Conference (TREC-2).[1]

1. INTRODUCTION

The task of information retrieval is to extract *relevant* documents from a large collection of documents in response to user queries. When the documents contain primarily unrestricted text (e.g., newspaper articles, legal documents, etc.) the relevance of a document is established through 'full-text' retrieval. This has been usually accomplished by identifying key terms in the documents (the process known as 'indexing') which could then be matched against terms in queries [2]. The effectiveness of any such term-based approach is directly related to the accuracy with which a set of terms represents the content of a document, as well as how well it contrasts a given document with respect to other documents. In other words, we are looking for a representation *R* such that for any text items *D1* and *D2*, *R(D1) = R(D2)* iff *meaning(D1) = meaning(D2)*, at an appropriate level of abstraction (which may depend on the types and character of anticipated queries).

The simplest word-based representations of content are usually inadequate since single words are rarely specific enough for accurate discrimination, and their grouping is often accidental. A better method is to identify groups of words that create meaningful *phrases*, especially if these phrases denote important concepts in the database domain. For example, *joint venture* is an important term in the Wall Street Journal (WSJ henceforth) database, while neither *joint* nor *venture* are important by themselves. In fact, in a 800+ MBytes database, both *joint* and *venture* would often be dropped from the list of terms by the system because their inverted document frequency (*idf*) weights were too low. In large databases comprising hundreds of thousands of documents the use of phrasal terms is not just desirable, it becomes necessary.

An accurate syntactic analysis is an essential prerequisite for selection of phrasal terms. Various statistical methods, e.g., based on word co-occurrences and mutual information, as well as partial parsing techniques, are prone to high error rates (sometimes as high as 50%), turning out many unwanted associations. Therefore a good, fast parser is necessary, but it is by no means sufficient. While syntactic phrases are often better indicators of content than 'statistical phrases' — where words are grouped solely on the basis of physical proximity, e.g., "college junior" is not the same as "junior college" — the creation of compound terms makes the term matching process more complex since in addition to the usual problems of synonymy and subsumption, one must deal with their structure (e.g., "college junior" is the same as "junior in college").

For all kinds of terms that can be assigned to the representation of a document, e.g., words, syntactic phrases, fixed phrases, and proper names, various levels of ''regularization'' are needed to assure that syntactic or lexical variations of input do not obscure underlying semantic uniformity. Without actually doing semantic analysis, this kind of normalization can be achieved through the following processes:[2]

(1) morphological stemming: e.g., *retrieving* is reduced to *retriev*;

(2) lexicon-based word normalization: e.g., *retrieval* is reduced to *retrieve*;

(3) operator-argument representation of phrases: e.g., *information retrieval, retrieving of information*, and *retrieve relevant information* are all assigned the same representation, *retrieve+information*;

(4) context-based term clustering into synonymy classes and subsumption hierarchies: e.g., *takeover* is a kind of *acquisition* (in business), and *Fortran* is a *programming language*.

In traditional full-text indexing, terms are selected from among words and stems and weighted according to their frequencies and distribution among documents. The introduction of terms which are derived primarily by linguistic means into the representation of documents changes the balance of frequency-based weighting and therefore calls for more complex term weighting schemes than

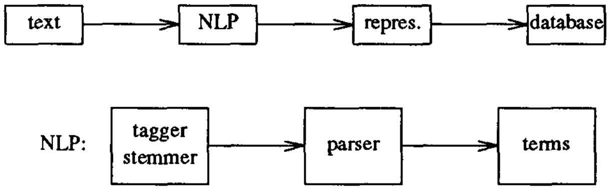
[1] See [1] for a detailed introduction to TREC.

[2] An alternative, but less efficient method is to generate all variants (lexical, syntactic, etc.) of words/phrases in the queries [3].

those devised and tested on single-word representations. The standard tf.idf scheme (term frequency times inverted document frequency), for example, weights terms proportionately to their global scores (idf) and their in-document frequencies (tf), usually normalized by document length. It is appropriate when most uses a term are explicit, that is, appropriate words actually occur in text. This, however, is frequently not the case with proper names or phrases as various anaphors can be used to create implicit term occurrences.

2. OVERALL DESIGN

We have established the general architecture of a NLP-IR system, depicted schematically below, in which an advanced NLP module is inserted between the textual input (new documents, user queries) and the database search engine (in our case, NIST's PRISE system[4]). This design has already shown some promise in producing a better performance than the base statistical system [5,6,7]..



In our system the database text is first processed with a sequence of programs that include a part-of-speech tagger, a lexicon-based morphological stemmer and a fast syntactic parser (TTP).[3] Subsequently certain types of phrases are extracted from the parse trees and used as compound indexing terms in addition to single-word terms. The extracted phrases are statistically analyzed as syntactic contexts in order to discover a variety of similarity links between smaller subphrases and words occurring in them. A further filtering process maps these similarity links onto semantic relations (generalization, specialization, synonymy, etc.) after which they are used to transform a user's request into a search query.

The user's natural language request is also parsed, and all indexing terms occurring in it are identified. Certain highly ambiguous, usually single-word terms may be dropped, provided that they also occur as elements in some compound terms. For example, "natural" may be deleted from a query already containing "natural language" because "natural" occurs in many unrelated contexts: "natural number", "natural logarithm", "natural approach", etc. At the same time, other terms may be added, namely those which are linked to some query term through admissible similarity relations. For example, "unlawful activity" is added to a query (TREC topic 055) containing the compound term "illegal activity" via a synonymy link between "illegal" and "unlawful".

One of the observations made during the course of TREC-2 was to note that removing low-quality terms from the queries is at least as important (and often more so) as adding synonyms and specializations. In some instances (e.g., routing runs) low-quality terms had to be removed (or inhibited) *before* similar terms could be added to the query or else the effect of query expansion was all but drowned out by the increased noise.

After the final query is constructed, the database search follows, and a ranked list of documents is returned. It should be noted that all the processing steps, those performed by the backbone system, and those performed by the natural language processing components, are fully automated, and no human intervention or manual encoding is required.

3. SELECTING PHRASAL TERMS

Syntactic phrases extracted from the parse structures are represented as head-modifier pairs. The head in such a pair is a central element of a phrase (main verb, main noun, etc.), while the modifier is one of the adjuncts or arguments of the head. In the TREC experiments reported here we extracted head-modifier word pairs only, i.e., nested pairs were not used even though this was warranted by the size of the database.[4]

Figure 1 shows all stages of the initial linguistic analysis of a sample sentence from the WSJ database. The reader may note that the parser's output is a predicate-argument structure centered around the main elements of various phrases. For example, BE is the main predicate (modified by HAVE) with 2 arguments (*subject, object*) and 2 adjuncts (*adv, sub_ord*). INVADE is the predicate in the subordinate clause with 2 arguments (*subject, object*). The subject of BE is a noun phrase with PRESIDENT as the head element, two modifiers (FORMER, SOVIET) and a determiner (THE). From this structure, we extract head-modifier pairs that become candidates for compound terms. In general, the following types of pairs are considered: (1) a head noun of a noun phrase and its left adjective or noun adjunct, (2) a head noun and the head of its right adjunct, (3) the main verb of a clause and the head of its object phrase, and (4) the head of the subject phrase and the main verb. These types of pairs account for most of the syntactic variants for relating two words (or simple phrases) into pairs carrying compatible semantic content. For example, the pair *retrieve+information* will be extracted from any of the following fragments: *information retrieval system; retrieval of information from databases*; and *information that can be retrieved by a user-controlled interactive search process*.[5] We also attempted to identify and remove any terms which were explicitly negated in order to prevent matches against their positive counterparts, either in the database or in the queries.

One difficulty in obtaining head-modifier pairs of highest accuracy is the notorious ambiguity of nominal compounds. The pair extractor looks at the distribution statistics of the compound terms to decide whether the association between any two words (nouns and adjectives) in a noun phrase is both syntactically valid and semantically significant. For example, we may accept *language+natural* and *processing+language* from *natural language processing* as correct, however, *case+trading* would make a mediocre term when extracted from *insider trading case*. On the other hand, it is important to extract *trading+insider* to be able to match

[3] For a description of TTP parser, refer to [8,9].

[4] Even with 2-word phrases, compound terms accounted for nearly 88% of all index entries, in other words, including 2-word phrases increased the index size approximately 8 times.

[5] Longer phrases or nested pairs may be more appropriate in some cases, e.g., when *former Soviet president* is broken into *former president* and *Soviet president*, we get something potentially quite different from what the original phrase refers to, and this may have a negative effect on retrieval precision.

INPUT SENTENCE
The former Soviet president has been a local hero ever since a Russian tank invaded Wisconsin.

TAGGED SENTENCE
The/*dt* former/*jj* Soviet/*jj* president/*nn* has/*vbz* been/*vbn* a/*dt* local/*jj* hero/*nn* ever/*rb* since/*in* a/*dt* Russian/*jj* tank/*nn* invaded/*vbd* Wisconsin/*np*

TAGGED & STEMMED SENTENCE
the/*dt* former/*jj* soviet/*jj* president/*nn* have/*vbz* be/*vbn* a/*dt* local/*jj* hero/*nn* ever/*rb* since/*in* a/*dt* russian/*jj* tank/*nn* invade/*vbd* wisconsin/*np* ./*per*

PARSED SENTENCE
```
[assert
 [[perf [HAVE]] [[verb [BE]]
     [subject
       [np [n PRESIDENT] [t_pos THE]
          [adj [FORMER]] [adj [SOVIET]]]]
     [object
       [np [n HERO] [t_pos A] [adj [LOCAL]]]]
     [adv EVER]
     [sub_ord [SINCE
            [[verb [INVADE]]
               [subject [np [n TANK] [t_pos A] [adj [RUSSIAN]]]]
               [object [np [name [WISCONSIN]]]]]]]]]
```

EXTRACTED TERMS & WEIGHTS

president	2.623519	soviet	5.416102
president+former	14.594883	hero	7.896426
invade	8.435012	tank	6.848128
tank+russian	16.030809	russian	7.383342
president+soviet	11.556747	hero+local	14.314775
tank+invade	17.402237	wisconsin	7.785689

Figure 1. Stages of sentence processing.

documents containing phrases *insider trading sanctions act* or *insider trading activity*. In addition, phrases with a significant number of occurrences across different documents, including those for which no clear disambiguation into pairs can be obtained, are included as a third level of index (beside single-word terms, and pairs).[6]

4. TERM WEIGHTING ISSUES

Finding a proper term weighting scheme is critical in term-based retrieval since the rank of a document is determined by the weights of the terms it shares with the query. One popular term weighting scheme, known as tf.idf, weights terms proportionately to their inverted document frequency scores and to their in-document frequencies (tf). The in-document frequency factor is usually normalized by the document length, that is, it is more significant for a term to occur in a short 100-word abstract, than in a 5000-word article.[7]

In our official TREC runs we used the normalized tf.idf weights for all terms alike: single 'ordinary-word' terms, proper names, as well as phrasal terms consisting of 2 or more words.[8] Whenever phrases were included in the term set of a document, the length of this document was increased accordingly. This had the effect of decreasing tf factors for 'regular' single word terms.

A standard tf.idf weighting scheme may be inappropriate for mixed term sets, consisting of ordinary concepts, proper names, and phrases, because:

(1) It favors terms that occur fairly frequently in a document, which supports only general-type queries (e.g., "all you know about 'star wars'"). Such queries were not typical in TREC.

(2) It attaches low weights to infrequent, highly specific terms, such as names and phrases, whose only occurrences in a document are often decisive for relevance. Note that such terms cannot be reliably distinguished using their distribution in the database as the sole factor, and therefore syntactic and lexical information is required.

(3) It does not address the problem of inter-term dependencies arising when phrasal terms and their component single-word terms are all included in a document representation, i.e., *launch+satellite* and *satellite* are not independent, and it is unclear whether they should be counted as two terms.

In our post-TREC-2 experiments we considered (1) and (2) only. We noted that linguistic phrases, that is, phrases derived from text through primarily linguistic means, display a markedly different statistical behaviour than 'statistical phrases', i.e., those obtained using frequency-based or probabilistic formulas such as Mutual Information [11]. For example, while statistical phrases with few occurrences in the corpus could be dismissed as insignificant or 'noise', infrequent linguistic phrases may in fact turn out to be quite important if only we could count all their implicit occurrences, e.g., as anaphors.

Rather than trying to resolve anaphoric references, we changed the weighting scheme so that the phrases (but not the names, which we did not distinguish in TREC-2) were more heavily weighted by their idf scores while the in-document frequency scores were replaced by logarithms multiplied by sufficiently large constants. In addition, the top N highest-idf matching terms (simple or compound) were counted more toward the document score than the remaining terms.

Schematically, these new weights for phrasal and highly specific terms are obtained using the following formula, while weights for most of the single-word terms remain unchanged:

$$weight(T_i)=(C_1*log(tf)+C_2*\alpha(N,i))*idf$$

In the above, $\alpha(N,i)$ is 1 for $i<N$ and is 0 otherwise.[9]

Table 1 illustrates the effect of differential weighting of phrasal terms using topic 101 and a relevant document (WSJ870226-0091)

[6] Longer phrases were not used in TREC-2.

[7] This is not always true, for example when all occurrences of a term are concentrated in a single section or a paragraph rather than spread around the article. See the following section for more discussion.

[8] Specifically, the system used *lnc-ntc* combination of weights which is already one of the most effective options of tf.idf; see [10] for details.

[9] The selection of a weighting formula was partly constrained by the fact that document-length-normalized tf weights were precomputed at the indexing stage and could not be altered without re-indexing of the entire database. The intuitive interpretation of the $\alpha(N,i)$ factor is given in the following section.

as an example. Note that while most of the affected terms have their weights increased, sometimes substantially, for some (e.g., *space+base*) the weight actually decreases. Table 2 shows how ranks of the relevant documents change when phrasal terms are used with the new weighting scheme. Changing the weighting scheme for compound terms has led to an overall increase of precision of more than 20% over our official TREC-2 ad-hoc results. Table 3 summarizes statistics of the runs for queries 101-150 against the WSJ database, both with new weighting scheme and with the standard tf.idf weighting.

5. 'HOT SPOT' RETRIEVAL

Another difficulty with frequency-based term weighting arises when a long document needs to be retrieved on the basis of a few short relevant passages. If the bulk of the document is not directly relevant to the query, then there is a strong possibility that the document will score low in the final ranking, despite some strongly relevant material in it. This problem can be dealt with by subdividing long documents at paragraph breaks, or into approximately equal length fragments and indexing the database with respect to these (e.g., [12]). While such approaches are effective, they also tend to be costly because of increased index size and more complicated access methods.

Topic 101 matches WSJ870226-0091 duplicate terms not shown		
TERM	*TF.IDF*	*NEW WEIGHT*
sdi	1750	1750
eris	3175	3175
star	1072	1072
wars	1670	1670
laser	1456	1456
weapon	1639	1639
missile	872	872
space+base	2641	2105
interceptor	2075	2075
exoatmospheric	1879	3480
system+defense	2846	2219
reentry+vehicle	1879	3480
initiative+defense	1646	2032
system+interceptor	2526	3118
DOC RANK	30	10

Table 1. The effect of differential term weighting.

DOC ID	OLD RANK	NEW RANK
WSJ891004-0119	7	1
WSJ891005-0005	15	4
WSJ890918-0173	2	5
WSJ880608-0121	14	7
WSJ870723-0064	8	8
WSJ870213-0053	10	12
WSJ891009-0009	35	18
WSJ890920-0115	39	26
WSJ891009-0188	73	46
WSJ880609-0061	53	50
WSJ870601-0075	128	52
WSJ890928-0184	40	61
WSJ891005-0001	283	72
WSJ871028-0059	183	93
WSJ880705-0194	97	95

Table 2. Rank changes for relevant documents for Topic 104 when phrasal terms are used in retrieval.

Efficiency considerations have led us to investigate an alternative approach to the *hot spot* retrieval which would not require re-indexing of the existing database or any changes in document access. In our approach, the maximum number of terms on which a query is permitted to match a document is limited to N highest weight terms, where N can be the same for all queries or may vary from one query to another. Note that this is not the same as simply taking the N top terms from each query. Rather, for each document for which there are M matching terms with the query, only min(M,N) of them, namely those which have highest weights, will be considered when computing the document score. Moreover, only the global importance weights for terms are considered (such as idf), while local in-document frequency (eg., tf) is suppressed by either taking a log or replacing it with a constant. The effect of this 'hot spot' retrieval is shown in Table 4 in the ranking of relevant documents within the top 30 retrieved documents for topic 72.

The final ranking is obtained by adding the scores of documents in 'regular' tf.idf ranking and in the hot-spot ranking.. While some of the recall may be sacrificed ('hot spot' retrieval has often lower recall than full query retrieval, and this becomes the lower bound on recall for the combined ranking) the combined ranking precision has been consistently better than in either of the original rankings: an average improvement is 10-12% above the tf.idf run precision (which is often the stronger of the two). The 'hot spot' weighting is represented with the α factor in the term weighting formula given in the previous section.

6. CONCLUSIONS

We presented some detail of our natural language information retrieval system consisting of an advanced NLP module and a 'pure' statistical core engine. While many problems remain to be resolved, including the question of adequacy of term-based representation of document content, we attempted to demonstrate that the architecture described here is nonetheless viable. We demonstrated that natural language processing can now be done on a fairly large scale and that its speed and robustness can match those of traditional statistical programs such as key-word indexing or statistical phrase extraction. We suggest moreover that when properly used natural language processing can be very effective in improving retrieval precision. In particular, we show that in term-based document representation, term weighting is at least as important as their selection. In order to achieve optimal performance terms obtained primarily through the linguistic analysis must be weighted differently than those obtained through traditional frequency-based methods.

On the other hand, we must be aware of the limits of NLP technologies at our disposal. While part-of-speech tagging, lexicon-based stemming, and parsing can be done on large amounts of text (hundreds of millions of words and more), other, more advanced

Run	con1	nyuir2	con2	con2+nlp
Tot number of docs over all queries				
Ret	50000	49876	49999	50000
Rel	3929	3929	3929	3929
RelRet	3129	3274	3332	3401
%chg		+4.6	+6.4	+8.7
Recall	(interp) Precision Averages			
0.00	0.7064	0.7528	0.7469	0.8063
0.10	0.5316	0.5567	0.5726	0.6198
0.20	0.4533	0.4721	0.4970	0.5566
0.30	0.3767	0.4060	0.4193	0.4786
0.40	0.3329	0.3617	0.3747	0.4257
0.50	0.2840	0.3135	0.3271	0.3828
0.60	0.2398	0.2703	0.2783	0.3380
0.70	0.1946	0.2231	0.2267	0.2817
0.80	0.1460	0.1667	0.1670	0.2164
0.90	0.0808	0.0915	0.0959	0.1471
1.00	0.0125	0.0154	0.0168	0.0474
Average precision over all rel docs				
Avg	0.2881	0.3111	0.3210	0.3759
%chg		+8.0	+11.4	+30.5
Precision at				
5 docs	0.5080	0.5360	0.5600	0.6040
10 docs	0.4680	0.4880	0.5020	0.5580
15 docs	0.4440	0.4693	0.4773	0.5253
20 docs	0.4310	0.4390	0.4560	0.4980
30 docs	0.3887	0.4067	0.4100	0.4607
100 docs	0.2840	0.3094	0.3084	0.3346
200 docs	0.2009	0.2139	0.2156	0.2325
500 docs	0.1075	0.1137	0.1162	0.1229
1000 docs	0.0626	0.0655	0.0666	0.0680
R-Precision (after Rel)				
Exact	0.3076	0.3320	0.3455	0.3950
%chg		+8.0	+12.3	+28.4

Table 3. Run statistics for ad-hoc queries 101-150 against WSJ database with 1000 docs per query: (1) *con1* - single-word terms only; (2) *nyuir2* - the official TREC-2 run including phrases with standard tf.idf weighting; (3) *con2* - single-word terms only with low weight terms removed; and (4) *con2+nlp* - single-word terms and phrases with the new weighting scheme. In all cases documents preprocessed with the lexicon-based suffix-trimmer.

DOCUMENT ID	RANK	SCORE
Full tf.idf retrieval - words and phrases		
WSJ901228-0063	2	15957
WSJ910619-0153	3	15843
WSJ910322-0041	4	15063
WSJ880118-0090	7	13816
WSJ910102-0058	11	12803
WSJ870324-0083	12	12720
WSJ910916-0109	17	11014
WSJ910208-0191	18	10912
WSJ871013-0105	19	10745
WSJ910419-0071	21	10540
WSJ901227-0001	27	9928
WSJ900904-0093	28	9685
WSJ910215-0054	30	9609
Hot-spot idf-dominated with N=20		
WSJ910916-0109	1	11822
WSJ910322-0041	2	11822
WSJ920226-0151	4	10016
WSJ901228-0063	6	9917
WSJ901227-0001	11	8704
WSJ870324-0083	12	8704
WSJ880127-0086	13	8704
WSJ910227-0107	14	7571
WSJ901227-0005	48	6754
WSJ900524-0125	51	6754
WSJ880118-0090	59	6754
WSJ911218-0028	61	6754
WSJ910719-0067	67	6754
Merged rankings - new weights		
WSJ910322-0041	1	15975
WSJ901228-0063	2	15060
WSJ910916-0109	3	13951
WSJ910619-0153	4	12745
WSJ870324-0083	6	12577
WSJ880118-0090	9	11732
WSJ920226-0151	11	11518
WSJ910102-0058	13	11225
WSJ901227-0001	16	11181
WSJ880127-0086	18	10871
WSJ910227-0107	23	9821
WSJ910419-0071	24	9811
WSJ871006-0091	37	8768

Table 4. Ranks of the relevant documents in hot-spot retrieval and merged ranking for Topic 72.

processing involving conceptual structuring, logical forms, etc., is still beyond reach, computationally. It may be assumed that these super-advanced techniques will prove even more effective, since they address the problem of representation-level limits; however the experimental evidence is sparse and necessarily limited to rather small scale tests (e.g., [13]).

Acknowledgements

We would like to thank Donna Harman of NIST for making her PRISE system available to us. We would also like to thank Ralph Weischedel and Heidi Fox of BBN for providing and assisting in the use of the part of speech tagger. Thanks to Ralph Grishman for his comments on an earlier version of this paper. This paper is based upon work supported by the Advanced Research Project Agency under Contract N00014-90-J-1851 from the Office of Naval Research, under Contract N00600-88-D-3717 from PRC Inc., and the National Science Foundation under Grant IRI-93-02615. We also acknowledge support from the Canadian Institute for Robotics and Intelligent Systems (IRIS).

References

1. Harman, Donna (ed.). 1993. *First Text REtrieval Conference.* NIST special publication 500-207.
2. Salton, Gerard. 1989. *Automatic Text Processing: the transformation, analysis, and retrieval of information by computer.* Addison-Wesley, Reading, MA.
3. Sparck Jones, K. and J. I. Tait. 1984. "Automatic search term variant generation." *Journal of Documentation,* 40(1), pp. 50-66.
4. Harman, Donna and Gerald Candela. 1989. "Retrieving Records from a Gigabyte of text on a Minicomputer Using Statistical Ranking." *Journal of the American Society for Information Science,* 41(8), pp. 581-589.
5. Strzalkowski, Tomek. 1993. "Natural Language Processing in Large-Scale Text Retrieval Tasks." Proceedings of the First Text REtrieval Conference (TREC-1), NIST Special Publication 500-207, pp. 173-187.
6. Strzalkowski, Tomek. 1993. "Robust Text Processing in Automated Information Retrieval." Proc. of ACL-sponsored workshop on Very Large Corpora. Ohio State Univ. Columbus, June 22.
7. Strzalkowski, Tomek and Barbara Vauthey. 1992. "Information Retrieval Using Robust Natural Language Processing." Proc. of the 30th ACL Meeting, Newark, DE, June-July. pp. 104-111.
8. Strzalkowski, Tomek. 1992. "TTP: A Fast and Robust Parser for Natural Language." Proceedings of the 14th International Conference on Computational Linguistics (COLING), Nantes, France, July 1992. pp. 198-204.
9. Strzalkowski, Tomek, and Peter Scheyen. 1993. "An Evaluation of TTP Parser: a preliminary report." Proceedings of International Workshop on Parsing Technologies (IWPT-93), Tilburg, Netherlands and Durbuy, Belgium, August 10-13.
10. Buckley, Chris. 1993. "The Importance of Proper Weighting Methods." Human Language Technology, Proceedings of the workshop, Princeton, NJ. Morgan-Kaufmann, pp. 349-352.
11. Lewis, David D. and W. Bruce Croft. 1990. "Term Clustering of Syntactic Phrases". Proceedings of ACM SIGIR-90, pp. 385-405.
12. Kwok, K.L., L. Papadopoulos and Kathy Y.Y. Kwan. 1993. "Retrieval Experiments with a Large Collection using PIRCS." Proceedings of TREC-1 conference, NIST special publication 500-207, pp. 153-172.
13. Mauldin, Michael. 1991. "Retrieval Performance in Ferret: A Conceptual Information Retrieval System" Proceedings of ACM SIGIR-91, pp. 347-355.