Okapi at TREC–3

S E Robertson S Walker S Jones M M Hancock-Beaulieu
M Gatford

Centre for Interactive Systems Research
Department of Information Science
City University
Northampton Square
London EC1V 0HB
UK

Advisers: E Michael Keen (University of Wales, Aberystwyth), Karen Sparck Jones (Cambridge University), Peter Willett (University of Sheffield)

1 Introduction

The sequence of TREC conferences has seen the City University Okapi IR system evolve in several ways. Before TREC–1 it was a very traditional probabilistic system comprising closely integrated search engine and interface, designed for casual use by searchers of bibliographic reference databases.

City at TREC–1

During the course of TREC–1 the low-level search functions were split off into a separate Basic Search System (BSS) [2], but retrieval and ranking of documents was still done using the "classical" probabilistic model of Robertson and Sparck Jones[7] with no account taken of document length or term frequency within document or query. Four runs were submitted to NIST for evaluation: automatic ad hoc, automatic routing, manual ad hoc and manual ad hoc with feedback. The results were undistinguished, although not among the worst. Of the ad hoc runs, the manual was better than the automatic (in which only the CONCEPTS fields of the topics were used), and feedback appeared beneficial.[1]

[1] We have only recently noticed that our TREC–1 (and probably also TREC–2) results would have been considerably worse had it not been that the system at that time could not handle documents longer than 64K, and so the longest few hundred documents in the database were truncated. The TREC–1 automatic ad hoc run redone on the full database (with cutoff at 200 documents) gives an 11–pt average of 0.10 (0.12), precision at 5 documents 0.37 (0.50); and at 30 documents 0.36 (0.42) (TREC–1 results in parentheses). This appears to be because the simple weighting scheme tends to favour long documents, particularly FR, few of which are relevant.

City at TREC–2

For TREC–2 the simple inverse collection frequency (ICF) term-weighting scheme was elaborated to embody within-document frequency and document length components, as well as within-query frequency, and a large number of weighting functions were investigated. Because of hardware failures few of the runs were ready in time, and City's official results were very poor. However, later automatic ad hoc and routing results, reported in [4, 5], were similar to the best official results from other participants. There were also some inconclusive experiments on adding adjacent pairs from the topic statements, and on automatic query expansion using the top-weighted terms extracted from the top-ranked documents from a trial search. Again, there was an interactive manual ad hoc run with feedback, but the results were far less good than City's best (unofficial) automatic run.

TREC–3

The emphasis in TREC–3 has been on

- further refinement of term-weighting functions
- an investigation of run-time passage determination and searching
- expansion of ad hoc queries by terms extracted from the top documents retrieved by a trial search
- new methods for choosing query expansion terms after relevance feedback, now split into:
  - methods of ranking terms prior to selection
  - subsequent selection procedures
- and the development of a user interface and search procedure within the new TREC interactive search framework.

The two successes have been in query expansion and in routing term selection. The modified term-weighting functions and passage retrieval have had small beneficial effects. For TREC–3 there were to be topics without the CONCEPTS fields, which had proved to be by far the most useful source of query terms. Query expansion, passage retrieval and the modified weighting functions, used together, have gone a long way towards compensating for this loss.

2 The system

Software

The Okapi software used for TREC–3 was similar to that used in previous TRECs, comprising a low level basic search system (BSS) and a user interface for the manual search experiments (section 7), together with data conversion and inversion utilities. There were also various scripts and programs for generating query terms, running batches of trials and performing evaluation. The main code is written in C, with additional material in awk and perl. The evaluation program is from Chris Buckley at Cornell.

Hardware

A single-processor Sun SS10 with 64 MB of core and about 12 GB of disk was used as the main development machine and file server. Batch processing was also done on two other Suns, a 4/330 with 40 MB and an IPX with 16. The SS10 is considerably faster than machines used for previous TRECs, particularly on disk I/O; this was important because the search-time passage determination procedure (section 4) was very greedy. In contrast to TREC–2, this time there were no very serious hardware problems.

Databases

Two databases were used: disks 1 & 2, and disk 3. In TRECs 1 and 2 all line and paragraph information was discarded. This time paragraph information had to be retained, and both for this reason and to improve readability for users of the interactive system most of the formatting of the source data was kept. (Some reformatting was done on the long lines of disk 1 WSJ.)

A 3-field structure was used, common to all source datasets. The first field was always the DOCNO and the third field contained all the searchable text, mainly the TEXT portions but also headline or title-like material for some datasets and documents. The second field was unindexed (and unsearchable) and so only (possibly) useful for display to users of the interactive system. It was empty except in the case of SJM, when it contained the DESCRIPT field; and the Ziff JOURNAL, AUTHOR and DESCRIPTORS fields.

3 Probabilistic model and basic procedures

3.1 Some notation

N: Number of items (documents) in the collection
n: Collection frequency: number of items containing a specific term
R: Number of items known to be relevant to a specific topic
r: Number of these containing the term
tf: Frequency of occurrence of the term within a specific document
qtf: Frequency of occurrence of the term within a specific query
dl: Document length (arbitrary units)
$avdl$: Average document length
BMxx: Best-match weighting function implemented in Okapi (see below)
k_i, b: Constants used in various BM functions (see below)

3.2 Weight functions

As in previous TRECs, the weighting functions used are based on the Robertson–Sparck Jones weight [7]:

$$w^{(1)} = \log \frac{(r+0.5)/(R-r+0.5)}{(n-r+0.5)/(N-n-R+r+0.5)}, \quad (1)$$

which reduces to an inverse collection frequency weight without relevance information ($R = r = 0$). This is the BM1 function used in TREC–1.

In TREC–2 and the work that followed [4, 5, 6], we demonstrated the effectiveness of the following two functions:

$$w = s_1 s_3 \times \frac{tf}{(k_1 + tf)} \times w^{(1)} \times \frac{qtf}{(k_3 + qtf)}$$
$$\text{and} \quad k_2 \times nq \frac{(avdl - dl)}{(avdl + dl)} \quad \text{(BM15)}$$
$$w = s_1 s_3 \times \frac{tf}{(\frac{k_1 \times dl}{avdl} + tf)} \times w^{(1)} \times \frac{qtf}{(k_3 + qtf)}$$
$$\text{and} \quad k_2 \times nq \frac{(avdl - dl)}{(avdl + dl)} \quad \text{(BM11)}$$

s_i are scaling constants related to k_i (see [6]). nq is the number of query terms, and the "and" in front of the last component indicates that this document length correction factor is "global": it is added at the end, after the weights for the individual terms have been summed, and is independent of which terms match.

In the course of investigating variant functions for TREC–3, we in effect combined BM11 and BM15 into a single function BM25, which allowed for a number of variations. The term frequency component is implemented as

$$\frac{tf^c}{K^c + tf^c} \tag{2}$$

with $K = k_1((1-b) + b\frac{dl}{avdl})$. Thus if $c = 1$, $b = 1$ gives BM11 and $b = 0$ gives BM15; different values of b give a mix of the two. The basis for BM11 was one of two possible models of document length (the "verbosity" hypothesis, [4]) which might be expected to exaggerate the document length effect; this is the justification for considering the mix.

A formula like equation 2, with $c > 1$, was suggested in [4], to give an s-shape to the function, as under some conditions the 2–Poisson model generates such a shape. Examination of a number of such curves generated by the 2–Poisson model suggested that c was related to K, and the formula $c = 1 + mK$, $m \geq 0$ was used in the experiments (in the event, m was largely ignored: see below). A scaling factor $s_1 = k_1 + 1$ was used, and the "global" document length correction was included. Also $s_3 = k_3 + 1$ was used, and where k_3 is given as ∞, the factor $s_3 \times qtf/(k_3 + qtf)$ is implemented as qtf on its own.

BM25 is referred to as BM25(k_1, k_2, k_3, b). It is always to be assumed that $m = 0$ unless stated.

The modified weight function seems able to give slightly improved results, at the cost of another parameter to be guessed. Non-zero m was not helpful. $b < 1$ can give some improvement. Values around 0.75 were usually used, sometimes with a higher k_1 than for BM11. Evaluation results for BM25 with various parameter values are not explicitly given in this paper.

3.3 Term ordering for feedback

In query expansion after relevance feedback in Okapi, terms from the relevant items are ranked according to some selection value which is intended to measure how useful they would be if added to the query. The formula usually used for this purpose (and in particular, the one used in TREC–1 and TREC–2) is the Robertson Selection Value (RSV), based on the argument in [8]. The formula given in that reference is $w(p-q)$ where w is the weight to be assigned to the term, p is the probability of the term occurring in a relevant document, and q is the probability that it occurs in a non-relevant document. For RSV, w is interpreted as the usual Robertson–Sparck Jones relevance weight $w^{(1)}$, p is estimated as r/R, and q is assumed to be negligible.

Given that the weighting function is now more complex, it seemed appropriate to consider some alternative interpretations. In particular, since within-document term frequency now figures in the weighting function, it should probably be part of the selection value (a good term is not just one which tends to occur in relevant documents, but one which tends to occur more frequently in relevant than in non-relevant documents). Although it is no longer obvious how to interpret the w in the $w(p-q)$ formula (since there is no longer a single weight for the term), a possible measure would keep the w as before, but reward terms that occur frequently in relevant documents by replacing r/R by $\sum_{\text{reldocs}} tf/R$ or rtf/R. This formula is referred to below as RSV2.

RSV2 seems to assume that the weight is a linear function of tf, as it would be with large k_1. However, as we have found that a smaller value of k_1 gives better performance, it seems likely that RSV2 is over-valuing large tf values. So we have also tried the (unweighted) average of RSV and RSV2, referred to as ARSV. Also, following earlier work by Efthimiadis [9], we have tried using r on its own as a selection value (referred to as the **r** criterion).

In the event, RSV2 and ARSV have not shown any advantage over RSV. The **r** criterion appears less good than the others.

In the past, this ranking of terms has been used to select the top n terms, where n is fixed (TREC–1) or variable between topics (TREC–2) (see section 6). Further development of these ideas, together with some results from early runs for TREC–3, suggested a more elaborate, stepwise term selection procedure.

3.4 Term selection and optimization

Theoretically, an alternative to term selection based on a ranking method such as those just described would be to try every possible combination of terms on training set, and use some performance evaluation measure to determine which combination is best. This is almost certainly not a practical proposition, but we have attempted a small step towards such optimization.

The principle was to take the terms in one of the rank orders indicated, and then to make a select/reject decision on each term in turn. This decision was based on one of the standard evaluation measures applied to the resulting retrieval: that is, a term was selected if its inclusion improved performance over that achieved at the previous step.

Although such a procedure is likely to be computationally heavy, it is not out of the question for a routing task. Full details of the procedure adopted are given in section 6.

4 Passage determination and searching

Some TREC participants have experimented with passage retrieval (e.g. [10]), with some success. In much previous passage retrieval work, however, passages are prespecified. The object of the City experiment described here was to investigate search-time *determination* of "good" passage(s) in each document by examining all, or many, of the possible sequences of text "atoms" (paragraphs, for example, or sentences).

There are at least three ways one could consider using passage retrieval.

- The retrieval status value of a whole document may be based on the score(s) of its best subdocument(s).
- In interactive searching the user could be presented (initially, or on request) with the best portions of a long document.
- In relevance feedback only the good portions need be used for feedback.

Only the first of these three uses has been tried in the present experiments.

Since the number of passages is nearly proportional to the square of the number $\mathcal{A}$ of text atoms in a document (and the total time to weight all passages is of order $\mathcal{A}^3$ unless the code is very carefully optimized[2]), it is not practical to use atoms which are too short in comparison with the length of a document. It was decided that the TREC atom should be a paragraph.[3] The Okapi database model was modified to incorporate paragraph and sentence information, and the TREC source disks reconverted in conformity with the new model. Paragraph identification was algorithmic, using indentation and/or blank lines in the source. Some of the more elaborate text structures, some of the FR documents for example, were not very accurately parsed; also, one-line paragraphs tended to become joined to the succeeding paragraph. Paragraph information for a document included length and offset, and the number of sentences in each paragraph. The mean length of a paragraph turned out to be not much more than 230 characters, with about 11 paragraphs in an average document and mean document length 2600 for both databases.

With this information it becomes possible to search any passage or sub-document which consists of an integral number of consecutive paragraphs. The system was set up so that the following could be varied:

- minimum number of atoms (paragraphs) in a passage (default 1)
- maximum number of atoms in a passage (default 20)
- number of atoms to "step" between passages (default 1).
- the weight functions depend on a notional "average document length" *avdl*; the true *avdl* (about 2600) is far too high for true weighting of short passages, so this parameter was sometimes reduced for the weighting of proper subdocuments only.

So as to avoid "passaging" documents with little chance of attaining a best passage weight in the top 1000, the first passage considered was the whole document. If this failed to come up to a certain threshold weight no further processing was done. By experiment, it was found that this threshold could be set to the weight of the 10000th whole document, where this was known, without losing more than a very small number of long documents with a good passage embedded somewhere. This reduced the number of documents considered by a factor of ten or more at the cost of a preliminary "straight" search for each topic. Finally, as a safety measure, it was also possible to set a maximum number of passages to be considered for a document. This was sometimes used, and it may have affected the final weights of up to about a dozen documents for some topics and conditions.

Results

A very large number of trials were done using topics 101–150 on the complete disk 1 & 2 database, first on single topics, then on topics 101–110, and finally on 101–150. Looking at individual documents suggested that the procedure behaved sensibly, but it proved difficult to obtain more than a small improvement over whole-document searching. Table 1 summarizes some results; see also table 3 for the effect of passage searching in combination with query expansion and table 7 for routing results. A minimum passage length of four paragraphs was a good compromise between speed and performance. Neither unlimited maximum passage length nor a fine granularity or large overlap gave more than a minimal improvement.[4]

In conjunction with query expansion, however, the improvement was considerably greater (see Section 5 and Table 3); it is not at all obvious why this should be so. For all the passage retrieval results given, the document weight was taken as the maximum of the weight of the best proper subdocument and the weight of the whole document.[5] We also tried linear combinations of

[2] If a maximum passage length is set this becomes $\mathcal{A}^2$

[3] The document with the most paragraphs is probably FR89119-0111, with about 8700. This can make about 3.9×10^7 passages of mean length 4350 paragraphs.

[4] In interactive searching it is unlikely that users would benefit from being offered passages longer than two or three screens.

[5] Where a maximum passage length has been set the whole document may not have been considered in the passage weighting.

Table 1: Automatic ad hoc results, passage retrieval, unexpanded queries: topics 151–200 TND only, disks 1 & 2

Passage min	Passage step	Passage max	*avdl*	AveP	P5	P30	P100	R-Prec	Rcl
1	1	8	1800	0.349	0.720	0.587	0.437	0.398	0.701
4	2	4	1800	0.346	0.720	0.586	0.439	0.391	0.692
4	2	4	2200	0.345	0.720	0.582	0.438	0.392	0.695
4	4	4	1800	0.344	0.724	0.583	0.437	0.386	0.691
4	2	12	1800	0.345	0.720	0.585	0.440	0.392	0.692
4	2	20	1800	0.345	0.716	0.585	0.440	0.392	0.692
4	2	24	1800	0.344	0.716	0.584	0.440	0.392	0.692
8	4	24	1800	0.342	0.728	0.589	0.434	0.387	0.687
8	4	8	1800	0.342	0.728	0.590	0.434	0.387	0.688
Non-passage result for comparison (official citya2)									
none				0.337	0.732	0.590	0.431	0.382	0.681
All runs BM25(2.0, 0.0, ∞, 0.75)									

best passage weight and document weight (as reported also in [10]); the best results from this were similar to those in Table 1, but achieved with different parameters.

5 Query expansion without relevance information

One of the experiments we did for the TREC–2 ad hoc was to attempt query expansion or modification without precise relevance information [4]. Query *modification* was done by reweighting the original query terms extracted from the topic statement on the basis of their distribution in the top-ranked documents retrieved in a trial search. There were no positive results from reweighting. For query *expansion*, the top documents from the trial search were used as the sole source of terms. Terms were selected in RSV sequence, with a limit on the number of non-topic terms. Any selected topic terms which occurred more than once in the topic statement were given a query term frequency component, with a value of 8 for k_3 (section 3.2).

A possibly–similar procedure appears to have been used with some success by at least one other TREC participant [11], but our best TREC–2 results showed only marginal and probably not significant improvement over the best from unmodified queries. Nevertheless, spurred by the relatively poor ad hoc results from topics with no CONCEPTS field, we decided to give it another try for TREC–3. This was unexpectedly successful.

Even if it has been considered, it may have been weighted with an *avdl* less than the true average document length, so the weight of the whole document considered as a passage may be less than its weight considered as a document.

Trial search and term selection

For all runs the trial search used the TITLE, NARRATIVE and DESCRIPTION fields of the topics with BM25(2.0, 0, ∞, 0.75). The top R documents were output and all terms other than stop and semi-stop terms extracted. These were F4-weighted on the basis of their occurrence in r of the R documents and the query term frequency adjustment applied. The resulting weight was multiplied by r/R to give an RSV value. The top T terms were then selected from the RSV-ordered list, subject to $RSV > 0$ and $r \geq 5$. Table 2 shows an example. (Table 4 illustrates a routing query for the same topic.)

Query expansion results

Table 3 gives a selection of ad hoc results. The official citya2 run is included for comparison: this used the same weighting method as citya1 but without expansion or passage-searching. Citya1 did better than citya2 on 35 of the topics. The results are fairly flat around 20–30 feedback documents and maxima of 20–40 additional terms. Passage searching improves results noticeably.

6 Automatic routing

Query term sources and weights

As in previous TRECs, for the official runs we used all the known relevant documents in the training collection (disks 1 & 2) as the sole source of query terms, ignoring the topic statements. After the official runs we repeated some runs restricting the term source to the subset of the officially relevant documents which appeared in the top 1000 documents retrieved by an ad hoc search. There have also been some experiments in which terms extracted from relevant documents have

Table 2: Topic 120, query expansion without relevance information: top 20 terms, $R = 30$, $k_3 = 8$

term	source	qtf	n	r	wt	RSV
terror	tit	6	8375	28	386	360
intern	tit	4	103519	25	144	120
privat	nar	3	37425	14	98	46
bomb	doc	0	9664	15	62	31
government...	nar	1	122323	23	40	31
threat	doc	0	15433	15	56	28
attack	doc	0	26544	16	49	26
state	doc	0	138351	22	35	26
militari	doc	0	33510	16	46	25
countri	doc	0	72086	19	40	25
act	nar	1	73037	19	40	25
offici	doc	0	117653	21	36	25
group	doc	0	124078	21	35	25
america...	doc	0	102703	20	36	24
consequ	nar	3	14062	7	98	23
libya	doc	0	1760	9	75	23
counterterror	doc	0	148	6	104	21
sponsor	doc	0	12950	12	53	21
iran...	doc	0	13455	12	52	21
econom	tit	3	64882	10	59	20
"Doc" means term does not occur in the topic statement.						

Table 3: Automatic ad hoc results, query expansion without relevance information: topics 151–200, disks 1 & 2

Feedback docs	Feedback max terms	r	passages	Mean terms	AveP	P5	P30	P100	R-Prec	Rcl
30	40	> 4	lgth 1–8	51	0.401	0.752	0.617	0.478	0.421	0.741
30	40	> 4	lgth 4–24 by 1	51	0.401	0.740	0.625	0.476	0.422	0.739
(the above row is the official citya1)										
20	40	> 4	none	47	0.389	0.768	0.620	0.473	0.418	0.723
30	40	> 4	none	51	0.388	0.752	0.615	0.471	0.410	0.725
20	20	> 4	none	32	0.381	0.760	0.606	0.464	0.409	0.720
100	50	> 4	none	67	0.361	0.680	0.575	0.453	0.389	0.706
100	50	> 9	none	63	0.359	0.676	0.569	0.452	0.387	0.705
Unexpanded run for comparison (official citya2)										
0	0		none	34	0.337	0.732	0.590	0.431	0.382	0.681
All runs BM25(2.0, 0.0, 8, 0.75).										

been given additional weight if they occurred twice or more in the topic statement.

All non–stop and non–semi–stop terms were extracted, and given the normal $w^{(1)}$ weights (equation 1). Where a bonus was given for terms which occurred more than once in the topic statement this was done by multiplying the $w^{(1)}$ weight by $(k_3 + 1)\frac{qtf}{k_3+qtf}$ (see section 3).

Term ordering

Potential terms were first ordered according to some criterion based on their occurrence in relevant and nonrelevant documents and in the collection as a whole. The four criteria tried are described in section 3.3.

Obviously, for most topics there was a very large number of potential query terms. In previous TRECs we tried two methods for term selection from the ordered termlists. Both involved selecting the top T terms; either T was the same for all topics, or a value was chosen for each topic. Retrospective runs were done in which T was varied, from 3 upwards. Not surprisingly it was found that better results were obtained by choosing the best value for each topic, rather than the single value which gave the best average precision (for example) over all topics (see Table 4 in [4]). The former method (T optimized for each topic) was used for the cityr2 run in TREC–3 (Table 6), where T is between 3 and 100.

Table 4 illustrates the observation that, for a given topic, performance generally does not vary smoothly with the number of terms. This is part of the motivation for trying to discover more effective term ordering criteria. But the figures in Table 5 suggest that there is not much to choose between the criteria, at least when the same number of terms is used for each query. Further, when the same number of terms is used for each topic, there is very little difference in the averaged results as T increases from about 15 to 100 or more.

Table 6 shows that retrospective results can be improved by "individualizing" the number of terms selected for each topic.

"Optimizing" the queries

Since none of the term ordering criteria seems particularly effective, being swamped by the vagaries of individual terms in individual topics, it was decided to try some approach to the optimization of the term set for each topic with respect to some retrospective evaluation statistic, specifically a stepwise select-or-reject procedure as discussed in section 3.4. The procedure evolved after a number of informal trials (specifically to ensure that it would run in reasonable time, say an hour or two per topic) was as follows:

- termweights were not varied
- the top three terms were used, unconditionally, to start building the termset
- terms were considered one at a time, with no backtracking, in the sequence given by one of the ordering criteria
- after the first three terms, each successive term was added to the query and the query run (with a cutoff of 1000 documents) and evaluated against the training set; if the evaluation result satisfied some acceptance criterion relative to the result of the previous iteration the new term was retained, otherwise it was rejected
- the procedure ran until some stopping rule was satisfied (see below).

The stopping rule was triggered when one of the following conditions was satisfied:

- the number of terms in the set reached $maxterms$
- $maxbad$ successive terms had been rejected
- the $lastterm$th term had been considered
- elapsed time exceeded $maxtime$

Acceptance criteria tried were increases in average precision or r-precision or recall. The most successful runs used average precision, with ties resolved on r-precision. Recall gave much more variability between topics, doing well on some and spectacularly badly on others. $maxterms$ was initially set at 20, but since a majority of queries came out with the full 20 terms some later runs were done using a value of 30. $Maxbad$ was always 8. $lastterm$ was set so high (150) that it never caused the stopping rule to be triggered. $Maxtime$ depended on the machine (and the time available), usually one or two hours per topic, although some runs with $maxterms = 30$ were given a higher value.

Automatic routing results

The "optimized" queries are much better than the other two types. Predictive and a few retrospective results for optimized queries are shown in Table 7. The procedure is computationally very demanding, sometimes taking several hours to produce a query on a Sun SS10 (excluding the time to extract and weight terms from the relevant documnets). At the time of writing work is in progress on a more efficient and perhaps sounder method of optimization, but no experiments have been done yet. The figures also suggest that there may be little difference in effectiveness between three of the four term-ordering criteria, but that the **r** criterion is less good.

Table 4: Topic 120, query terms extracted from relevant documents: effect of adding successive terms in ARSV order

Terms	wt	ARSV	AveP	P5	P30	P100	R-Prec	Rcl
terror	140	381						
airline	55	139						
secure	37	134	0.068	0.200	0.133	0.120	0.126	0.568
carrier	46	81	0.102	0.400	0.233	0.160	0.168	0.568
travel	48	71	0.126	0.600	0.267	0.200	0.210	0.632
intern	44	67	0.152	0.600	0.300	0.190	0.200	0.674
air	32	66	0.130	0.400	0.333	0.200	0.200	0.579
iran...	54	65	0.152	0.600	0.367	0.230	0.242	0.642
foreign	37	65	0.145	0.400	0.300	0.220	0.232	0.642
america...	35	65	0.145	0.400	0.333	0.230	0.242	0.632
libya	74	64						
bomb	59	56	0.125	0.400	0.233	0.190	0.190	0.653
flight	53	54						
faa	50	53						
passeng	58	53	0.102	0.400	0.233	0.230	0.232	0.611
pan	65	51						
airport...	51	48						
aviat	50	46	0.067	0.400	0.167	0.160	0.168	0.505
libyan	77	44						
europ...	39	42	0.088	0.400	0.200	0.180	0.179	0.579
(25 terms)			0.087	0.400	0.200	0.170	0.179	0.579
(30 terms)			0.088	0.400	0.133	0.190	0.200	0.579
(40 terms)			0.104	0.200	0.233	0.240	0.242	0.547
(50 terms)			0.117	0.400	0.267	0.220	0.232	0.600
(60 terms)			0.141	0.400	0.367	0.260	0.274	0.547
(75 terms)			0.121	0.600	0.300	0.210	0.221	0.537
(100 terms)			0.112	0.600	0.300	0.190	0.190	0.516
(125 terms)			0.129	0.600	0.300	0.220	0.210	0.526
(150 terms)			0.010	0.000	0.033	0.030	0.032	0.305

Table 5: Best routing results (retrospective) with same number of query terms for all topics

# terms	Criterion	AveP	P5	P30	P100	R-Prec	Rcl
18	RSV2	0.347	0.740	0.643	0.498	0.393	0.684
20	RSV2	0.351	0.724	0.656	0.497	0.398	0.681
30	RSV2	0.347	0.776	0.647	0.496	0.399	0.669
40	RSV2	0.353	0.756	0.653	0.510	0.400	0.674
50	RSV2	0.355	0.756	0.652	0.515	0.402	0.670
60	RSV2	0.356	0.764	0.655	0.516	0.400	0.667
75	RSV2	0.354	0.788	0.652	0.514	0.400	0.661
100	RSV2	0.346	0.788	0.651	0.509	0.394	0.652
40	RSV	0.351	0.780	0.655	0.511	0.396	0.669
50	RSV	0.356	0.784	0.667	0.516	0.397	0.668
60	RSV	0.354	0.800	0.654	0.513	0.395	0.664
75	RSV	0.350	0.824	0.663	0.510	0.393	0.657
15	ARSV	0.352	0.772	0.641	0.499	0.392	0.697
18	ARSV	0.346	0.724	0.630	0.497	0.398	0.684
20	ARSV	0.349	0.732	0.642	0.498	0.399	0.684
30	ARSV	0.346	0.756	0.655	0.498	0.397	0.666
40	ARSV	0.355	0.760	0.658	0.513	0.403	0.672
50	ARSV	0.354	0.752	0.649	0.513	0.397	0.670
60	ARSV	0.355	0.780	0.660	0.515	0.401	0.666
75	ARSV	0.355	0.800	0.659	0.511	0.399	0.660

Table 6: Best routing results using top T terms, T chosen to maximize AveP for a topic

Criterion	AveP	P5	P30	P100	R-Prec	Rcl
Predictive						
ARSV	0.371	0.660	0.584	0.457	0.393	0.752
RSV2	0.363	0.704	0.578	0.447	0.388	0.744
RSV	0.362	0.648	0.553	0.451	0.392	0.747
(the above row is the official cityr2 run)						
Retrospective						
RSV2	0.414	0.848	0.719	0.560	0.445	0.724
ARSV	0.410	0.840	0.715	0.561	0.442	0.723
RSV	0.409	0.856	0.707	0.562	0.441	0.719

$Maxterms = 30$ gives better results than $maxterms = 20$, and possibly a further small improvement might be obtained by setting $maxterms$ still higher. A small topic term weight bonus ($k3 > 0$) appears to be beneficial. There was little difference between the weighting functions $BM25(2.0, 0.0, -, 0.75)$ and $BM25(0.8, -1.0, -, 1.0)$ ($=BM11$) (not shown in the table). Passage searching improves the results still further. Perhaps more interestingly, reducing the amount of training information by about 25% by using only the relevant records retrieved by one of the better ad hoc methods does not affect the results as much as might be expected; looking at individual topics, a few do substantially worse but some actually produce better results than with the full relevant set. (These are the rows marked "own rels" in table 7.)

7 Interactive routing

In comparison with TREC 1 and 2 where interactive searching was undertaken for ad hoc queries, TREC-3 routing queries constituted quite a different task and required different experimental conditions. The searchers were members of the City Okapi research team, who played the role of intermediaries. The official relevance judgements for the training document set served to simulate end-user relevance judgements in a realistic routing task.

The Appendix gives a factual description of the interactive system itself, the experimental conditions and the search process, as an addendum to the official system description provided elsewhere in these proceedings.[6]

7.1 The task and interactive process

The aim of the exercise was to generate an optimal query based on (a) information given with the topics (i.e. narrative, concepts and descriptions), and (b) terms extracted from relevant documents. The searchers made their own relevance judgements whilst interacting with the system using knowledge about the official relevance judgements. The interface was designed to facilitate query formulation rather than the creation of a set of relevant documents and searchers made use of the different information presented during the interaction to meet that end. One major improvement was that they no longer felt inhibited about examining documents at any stage in the search process, as they had previously under the 'frozen ranks' regime. A second was the ability to treat phrases as search terms, which were weighted as single terms and retained throughout the search, provided the relative weights were high enough. Thirdly, searchers were able to remove terms from term sets produced by automatic query expansion in order to eliminate 'noise' in the system generated term sets, e.g. numbers, proper names or other rare terms, which might be considered to have a disproportionately high weight.

Initial query formulation

Search sessions consisted of three iterative phases. Firstly, in the initial query formulation phase the searcher could define different aspects of the topic with separate term sets and then join the sets to generate an initial query, from which a document set would be retrieved. The different commands and operators (define, join, adj) provided fine control over the elements of the search and to some extent enabled the searcher to structure the query. The 'adj' operator was used extensively to generate phrases: 232 times compared with the default operator BM11 (153 times).

Viewing results

The second phase, viewing results, involved the display of brief and full records. The brief record display gave a breakdown of the occurrence of query terms in the individual records and indicated the document source. The information on term occurrence was useful for multi-faceted queries, where the co-occurrence of two or more terms might be deemed important. However in most cases it simply provided a summary view since relevant terms could be combined in so many different ways. Likewise the document source served as background information but did not generally influence which full records were chosen for display.

In 75% of the searches the display of both the brief and full records was confined to the top 50 documents generated by the query; in only one instance did the scan go down to the 300+ level. As might be expected, the searchers' main objective was to achieve a reasonable precision amongst the top documents, rather than a high recall overall. However on occasions searchers did jump further down the ranking to check for more relevant documents, if the total number of officially judged relevant documents was known to be high.

Relevance judgements

Relevance judgements were made after viewing the full record, at which point the official relevance judgements were made available. However no distinction was made between documents not seen by the assessors and those definitely judged as not relevant, consequently documents tended to be read thoroughly even if marked with a 'no'.

[6] The weighting function used in the interactive system was BM11, as this was the best available at the time that system was implemented.

Table 7: Some routing results with "optimized" queries

Conditions	AveP	P5	P30	P100	R-Prec	Rcl
Predictive (topics 101–150 on disk 3)						
$maxterms = 30$, $k_3 = 2$, passages (lgth 1–8)	0.430	0.728	0.615	0.480	0.449	0.791
$maxterms = 30$, $k_3 = 2$, passages (lgth 1–8), own rels	0.419	0.716	0.606	0.473	0.434	0.784
$maxterms = 30$, $k_3 = 2$, passages (lgth 4, o'lap 2)	0.426	0.724	0.611	0.480	0.449	0.788
$maxterms = 30$, $k_3 = 2$	0.425	0.724	0.603	0.483	0.447	0.788
$maxterms = 30$, $k_3 = 2$, own rels	0.417	0.716	0.610	0.475	0.436	0.775
$k_3 = 2$	0.412	0.692	0.605	0.474	0.436	0.779
$maxterms = 30$	0.414	0.744	0.621	0.482	0.443	0.762
$maxterms = 30$, own rels	0.405	0.696	0.602	0.467	0.428	0.754
passages (lgth 1–20)	0.415	0.716	0.621	0.477	0.439	0.779
	0.407	0.716	0.612	0.475	0.435	0.765
(the above row is the official cityr1 run)						
own rels	0.401	0.684	0.598	0.467	0.425	0.753
ARSV	0.406	0.692	0.599	0.476	0.438	0.770
RSV2	0.401	0.696	0.604	0.466	0.425	0.748
r	0.366	0.676	0.582	0.453	0.401	0.738
Retrospective (topics 101–150 on disks 1 & 2)						
passages (lgth 1–20)	0.500	0.944	0.789	0.618	0.502	0.794
	0.492	0.956	0.795	0.609	0.495	0.772
ARSV	0.481	0.928	0.769	0.609	0.490	0.768
RSV2	0.478	0.916	0.773	0.600	0.487	0.761
r	0.448	0.908	0.745	0.584	0.465	0.745
Ordering criterion RSV, $k_3 = 0$ and $maxterms = 20$ unless stated						
All relevant documents used (mean 239 per topic) except where "own rels" stated (176 per topic).						

Since official judgements were available, searchers were required to concentrate on selection of documents likely to be useful for term extraction. This reduced the conflicts experienced under TREC-2. The differences between official and searcher judgements are shown in Table 8.

Much of this difference can be accounted for by the fact that judgements were being made for a different purpose, but there were instances where the assessors appeared to have missed documents containing relevant sections interspersed with other material, or to have judged on the basis of simple term occurrence rather than query relevance.

Final query

In the third phase, usually after identification of 10 to 12 relevant documents, a new set of terms was generated by the system from the relevance feedback information. In most cases this constituted the final optimal expanded query, although the extracted term sets for 35 out of the 50 queries were modified by the searchers. After two or more iterations it became difficult to decide between similar term-sets. For 18 out of 50 topics, the last term set extracted was not the one chosen to form the final query. In two cases (topics 132 and 140) the initial user generated query produced satisfactory results without the need for term extraction and on at least one occasion the initial query was chosen as the final query in preference to the extracted term set.

7.2 Human intervention and the probabilistic models

In this round of TREC two features were introduced to provide more flexibility for interactive searching. The first allowed searchers to define phrases as query terms, which were treated as single terms in the term extraction process. The second provided searchers with the facility to delete candidate terms from an extracted term set. What effect this type of human intervention has on the probabilistic models is unknown. Some words occurred in derived term sets both as phrase components and as single terms, without any weight adjustment. In some instances searchers removed the single terms. Although searchers intuitively appeared to prefer to use phrases in formulating queries, the implications for the weighting functions need further consideration.

Similarly searchers were not aware of the full consequences of the deletion of individual terms from an extracted term set. One effect of extraction was to bring out more specific terms, including proper names. Searchers were sometimes doubtful about the potential

Table 8: Official vs Searcher Relevance Judgements

	Searcher			
Official	Y	N	?	Total
Y	573	78	0	651
N	154	446	2	602
Not Seen	48	86	0	134
Total	775	610	2	1387

value of such terms in routing queries, and tended to delete them in favour of more general ones. This highlights the artificiality of the task and the conflict of attempting to generate an optimal routing query which would be effective in another database and the very specific, often topical nature of some of the queries. Searchers were uncertain about whether to retain time-dependent names, events, and places which had been successful in a current context.

Another aspect of the weighting function which influenced human/system interaction relates to document length. The algorithm used this time brought short documents to the top of the list, with AP and WSJ sources being the most common. Such documents tended to be more homogeneous than those from other sources. This appeared to be a helpful property for both relevance judgement and term extraction.

7.3 Results

Output from the interactive system were queries like those shown in table 9.

Table 10 shows the results of applying these searches predictively and retrospectively. The predictive result should not be compared with the routing results (Table 7) because the routing queries were derived using a very large amount of relevance information, whereas the interactive queries had the benefit only of those few relevant documents found by the searchers. It is probably more fruitful to compare the result of applying the interactive searches retrospectively (i.e. the output which the searches would have obtained had they executed the final searches) with automatic ad hoc results. Since the searchers had access to complete topic statements the best comparison is with an automatic run using all topic fields.

8 Conclusions

8.1 Overview

In the course of participating in three rounds of TREC, the Okapi team has made very substantial progress. Internally, the system has been developed from an interactive search program into a sophisticated distributed tool for a wide variety of experiments. In terms of generally applicable research results, we have shown the benefits of continuing to work within the framework of the classical probabilistic model of Robertson and Sparck Jones. While the field of information retrieval continues to be strongly empirically driven (a tendency reinforced by the entire TREC programme), and any practical system has to make use of methods and techniques based on very different theories, arguments or observations, it remains possible for an effective system design to be guided by a single theoretical framework. Furthermore, even without such developments as regression analysis, the classical approach is capable of achieving performance levels comparable with the best systems in the world today.

8.2 Main conclusions from TREC–3 experiments

Term-weighting functions

The basic "rough model" methods developed for TREC–2, whose benefits were not apparent in the official results submitted to TREC–2 but emerged in subsequent experiments, have now been shown to be effective under the full rigour of the official TREC procedures. These methods allow the inclusion of within-document and within-query term frequency and document length into the Robertson–Sparck Jones relevance weighting model, and are applicable either with or without relevance feedback.

However, attempts at somewhat less rough models have shown only small benefit.

Passages

Run-time passage determination is feasible, if computationally expensive. In common with other investigators, we have shown some benefits for document retrieval, though not very large ones, from considering best-matching passages. It is likely that the most important uses of passage determination will be in interactive systems and in connection with feedback for query expansion.

Table 9: Topic 120: query from interactive search after relevance feedback

Term(s)	op	weight
guerilla		125
thailand		84
iraq...		66
holidai		63
iran...		58
econom consequ	adj	113
intern terror	adj	112
trade restrict	adj	110
travel		51
trade polici	adj	98
econom effect	adj	98
properti damag	sames	93
russia...		45
europ...		41
econom impact	adj	70
busi		23

Table 10: Interactive results

Conditions	AveP	P5	P30	P100	R-Prec	Rcl
Predictive						
BM11	0.250	0.560	0.445	0.345	0.302	0.648
(the above row is the official cityi1)						
Retrospective						
BM11	0.283	0.704	0.569	0.438	0.337	0.620
best automatic for comparison						
BM11, TCND topic fds	0.366	0.660	0.577	0.492	0.411	0.754
BM11, TND topic fds	0.294	0.600	0.517	0.435	0.350	0.659

Query expansion without relevance information

Somewhat to our surprise, query expansion based on the top ranked documents from an initial search, irrespective of relevance, proved to be of benefit with the shorter queries now in use. Furthermore, this technique combined effectively with passage retrieval.

Ordering and selection of expansion terms

We have not managed to improve on the term-ordering measures used in previous experiments. However, the stepwise selection or rejection of terms from the ranked list, although computationally expensive, proved very effective. This represents a return to our old friend, term dependencies.

Interactive searching

The reconciliation of the demands of interactive searching with the kind of controlled experiment represented by TREC has a long way to go. Although we have made a serious attempt at evaluating an interactive method within TREC rules, we do not believe that it is yet appropriate to try to compare interactive with non-interactive procedures.

8.3 Futures

The automatic methods developed for Okapi for TREC–3 depart somewhat from the principles on which Okapi was originally based, in that they involved some computationally heavy procedures (specifically those involved in query expansion for routing and in passage retrieval) which may not be feasible, as they stand, in a live-use system. One future line of work (within or outside TREC) will be to try to achieve similar levels of performance with simpler methods.

The scope for further performance improvements is debatable. It is possible that we and other TREC participants are approaching some limit of performance, or at least a point of diminishing returns. However, the real progress made over three years of TREC (after thirty years and more of research in information retrieval) does encourage the view that not all ideas have yet been exhausted. We have every expectation of further improvements in successive rounds of TREC and elsewhere.

In common with most other TREC participants, we have done much too little work on analysing individual or selected groups of instances (topics, documents, terms), to try to understand in more detail the circumstances under which our methods work or do not work. The time pressures of TREC participation and the scale of the operation do tend to discourage such analysis; at the same time, the TREC material provides a very great deal of scope for it, and there could be considerable benefits from it.

In many ways the most interesting and currently puzzling area is that of interactive searching. The apparent huge performance advantage of automatic over interactive methods may in various ways be an artifact of the methodology, but it most certainly deserves substantial further investigation. Given that most IR system use in the world today is interactive, the importance of achieving a better understanding of the phenomenon is hard to exaggerate.

References

[1] *The First Text REtrieval Conference (TREC–1).* Edited by D K Harman. Gaithersburg, MD: NIST, 1993.

[2] Robertson S E *et al.* Okapi at TREC. In: [1]. p21–30.

[3] *The Second Text REtrieval Conference (TREC–2).* Edited by D K Harman. Gaithersburg, MD: NIST, 1994.

[4] Robertson S E *et al.* Okapi at TREC–2. In: [3]. p21–34.

[5] Robertson S E and Walker S. Some simple effective approximations to the 2–Poisson model for probabilistic weighted retrieval. In: Croft W B and van Rijsbergen C J (eds). *Proceedings of the 17th Annual International ACM–SIGIR Conference on Research and Development in Information Retrieval, Dublin 1994.* Springer-Verlag 1994. (p232–241)

[6] Robertson S E, Walker S and Hancock-Beaulieu, M. Large test collection experiments on an operational, interactive system: Okapi at TREC. *Information Processing and Management* (to appear).

[7] Robertson S E and Sparck Jones K. Relevance weighting of search terms. *Journal of the American Society for Information Science 27* May–June 1976 p129–146.

[8] Robertson S E. On term selection for query expansion. *Journal of Documentation 46* Dec 1990 p359–364.

[9] Efthimiadis E N. A user-centred evaluation of ranking algorithms for interactive query expansion. *SIGIR Forum (USA), spec. issue.*, 1993, p146–59, (Paper given at: Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Pittsburgh, PA, USA, 27 June-1 July 1993)

[10] Callan J P. Passage-level evidence in document retrieval. In: Croft W B and van Rijsbergen C J (eds). *Proceedings of the 17th Annual International ACM–SIGIR Conference on Research and Development in Information Retrieval, Dublin 1994.* Springer-Verlag 1994. (p302–310)

[11] Evans D and Lefferts R. Design and evaluation of the CLARIT–TREC–2 system. In: [3]. p137–150.

Appendix: Addendum to System Description for Interactive Experiments

A System Description

A.1 Summary

Figure 1 (attached) shows a screen dump from a running system. The most important new functions in the TREC-3 interface were those for:

- User-controlled definition and manipulation of *term-sets*, reflecting the fact that our objective was to generate routing queries rather than sets of documents,
- Display of *brief records* giving an overview of a document-set and relevance judgements upon it, allowing searchers to assess the performance of the current query,
- Automatic retention and re-use of user-defined *phrases*, following new term extraction from relevant documents.

A.2 Interface style

The basic interaction was command-driven, but the interface was designed to run in an X-windows environment. One window was used for entering commands and receiving summary responses, another to show lists of brief records comprising a document set, and another to display a complete document. Brief record lists, and complete document displays, were piped through the Unix *less* utility, enabling repeated scrolling and rudimentary within-document searching. In displays of retrieved documents, any query terms are capitalized and surrounded by asterisks.

A.3 Usable features of the interface

The most important commands are described below, in roughly their expected order of use during a search session.

A.3.1 Define

Define a term-set using one or more key-words and operators. The default operator is standard Okapi BM11. Two other useful operators are:

- ADJ: adjacency, used to generate phrases. The presence of intervening stop-words is ignored.
- SAMES: words must occur in the same sentence.

Boolean operators AND, OR and NOT are available, but unlikely to be useful in the current context. The "define" command causes a numbered term-set to be created, whose details are retained by the interface client software. It reports back on the number of documents matched by the term-set, but does *not* generate a permanent document-set.

A.3.2 Join

Join two or more term-sets. This enables the creation of complex queries, comprising, for instance, two or more ADJ expressions. Again, no permanent document-set is created.

A.3.3 Docset

Generate a document-set by submitting a term-set as a query to the Okapi search engine. Information about this set is retained by the server.

A.3.4 Brief

Show brief records from a document-set. For each record, the following information is displayed:

- Document set-number,
- Sequence number of record in set (used by subsequent requests for full-record displays),
- Source (e.g. ZF, AP, FR, etc.),
- Weight ($w^{(1)}$),
- Summary of query terms occurring in the document,
- Both the "official" and the searcher's previous relevance judgements. N.B. These appear only after a full display of the relevant record has been requested.

A.3.5 Show

Show a full record. The text is piped through the unix *less* utility, enabling the user to scroll and search the document. Following this command, the system requires a relevance judgement—in the current context this should reflect the searcher's estimate as to whether the document contains terms which will be useful at the query expansion phase. A running total is kept of the number of relevant and non-relevant records seen, to assist searchers in deciding when to attempt new term extraction.

A.3.6 Extract

Create a new term-set by extracting terms with a high frequency of occurrence in relevant documents. The top 50 such terms are identified; the top 20 are displayed in weight order. Existing user-defined phrases are submitted to the term extraction process and included in system-generated sets, if their occurrence in relevant documents warrants it.

A.3.7 Remove

Remove terms from a term-set. This operation can be applied to any term-set, but is most likely to be used on one generated by automatic query expansion. Its main purpose is to allow removal of "noise" terms from generated sets, e.g. numbers, typos, and other peculiarities which have a high weight because of their low frequency.

Following term removal, the remaining terms are promoted upwards by one place and the top 20 are again displayed. It is possible to remove a range of terms, including all those not currently displayed, so that a final query formulation is confined to terms actually seen by the searcher.

A.3.8 Results

Produce the final search output, i.e. the term-set which is to serve as the final query formulation.

B Experimental Conditions

B.1 Searcher Characteristics

The five (female) participants comprised two members of academic staff, one member of the research staff, and two postgraduate students. Their ages ranged from mid-20s to early 50s. Each searcher was allocated a batch of ten contiguous queries from the overall list, enabling some comparisons to be made about their search behaviour.

None of the searchers had any prior familiarity with the retrieval topics. They saw themselves as intermediaries, carrying out searches on behalf of end-users who were in the position to deliver relevance judgements. Three had existing experience of Boolean searching; one had a very detailed knowledge of the statistical principles underlying the okapi probabilistic search algorithm.

B.2 Task description / training

All but one of the searchers had participated in TREC–2 and were familiar with the objectives of the experiment—in fact the new interface for TREC–3 was based largely upon their proposals. In preparation for their task, they were given a demonstration of the interface, and undertook some dry runs with previous queries which were not part of the official training set. The system description (see section A) above was treated as a basic user guide, and on-line help was available to give the full syntax of the command language.

C Search process

C.1 Clock time

The figures given below are for on-line clock times only. On average about 5 minutes was spent off-line in thinking about the initial query; most work was done (as it should be in an interactive situation) by examining the effect of using search terms and functions with the database.

Mean	Median	Variance	Range
39.32	39.00	468.47	8–84

C.2 Number of documents viewed

In this context "Viewing" a document means displaying and reading its full text using the *show* command, and making a relevance judgement on it. Since brief record entries were listed 50 at a time, it was not practical to count them individually.

Mean	Median	Variance	Range
27.78	23.00	184.42	10–72

C.3 Number of iterations

At the start of the exercise, two possible forms of search "iteration" were identified:

- A *major* iteration was considered to involve all stages of the search from initial to final query formulation. A straightforward search was expected to require only one such iteration, where the initial query yielded enough relevant records for use by the later processes. A second or third major iteration would be counted when it was necessary to go back and reformulate the query in the light of documents examined.

- A *minor* iteration would involve a sub-series of actions, i.e. create a document-set, make relevance judgements, extract new terms from relevant documents, and create a new document set from the expanded query. A search might include two or three such iterations — repeated until the searcher was satisfied that the current query was finding a good proportion of relevant documents.

Based on these original criteria, 11 searches consisted of two or more major iterations, in that new definitions were entered *after* term-extraction from relevant documents. In practice, however, new definitions involved addition of a few extra terms to existing queries rather than complete re-starts, so it is probably more accurate to say that there were *no* major iterations.

For reporting purposes, the use of the *extract* function was treated as the boundary between one minor iteration and another. Summary figures for the use of this command are as follows:

Mean	Median	Variance	Range
1.5	1.0	0.62	0–3

Following is a more detailed breakdown of number of queries by number of term extractions. In two cases, (topics 132 and 140), the initial query was considered to produce satisfactory results without the need for any extraction at all.

Queries	Extractions
2	0
28	1
14	2
6	3

C.4 Number of terms used

In this context the "initial query" is considered to be the term-set used by the first *docset* command; the "final query" is the one output following a *results* command. Note that overall 262 "terms" defined by users were in fact *phrases* specified with adjacency operators. System-derived terms were all single words, except for a few ad hoc phrases in the Okapi GSL.

	Mean	Median	Variance	Range
initial	8.06	7	16.47	2–20
final	16.86	20	34.37	3–28

C.5 Use of system features — summary table

Command	Mean	Median	Variance	Range
Define	8.58	7.00	13.84	4–18
Join	3.22	2.00	7.03	1–13
Docset	4.00	3.00	7.47	1–15
Brief	4.74	4.00	14.65	0–16
Show	27.78	23.00	184.42	10–72
Extract	1.50	1.00	0.62	0–3
Remove	3.22	2.00	16.01	0–15

C.6 Number of user errors

No data were collected under this heading.

C.7 Search narrative for query 122

Two attempts were made on this topic because a system failure occurred half way through the first one. On the second occasion the searcher entered fewer terms initially before creating and examining document sets, having found that some of her original candidates (e.g. *evaluation, marketing*) were taking the search in the wrong direction. However it is unlikely that the second attempt was any more successful than the first would have been.

The initial query term consisted of the words: *cancer drug develop test* which were all required to occur in the same sentence. Of the 12 top documents examined from this search, only 3 were officially judged relevant, although the searcher included 2 others as potentially useful for term extraction. As the following quotation illustrates, all the right terms may co-occur in a sentence, without really matching the query:

"The Food and **Drug** Administration has approved a **test** that can detect the sexually transmitted virus believed to be linked to the **development** of cervical **cancer**, a Baltimore newspaper reported Saturday."

The second query term consisted of the words *anti cancer laboratory*, which once again were required to be in the same sentence. This yielded 3 more officially relevant documents, and 2 others deemed useful for term extraction. The search went through two extraction / retrieval cycles, during which one further term *leukemia* was entered.

Extracted terms output for the final query were: *patients, tumors, cells, therapy, immune, agent, chemotherapy, surgery, Kaposi's, transplant, treatment, bacteria, approved, research.* Some extracted terms deleted by the searcher were: *area, New York, food, Kettering, aid, architecture, protein, infected.* One of the difficulties when examining documents was to sift out those mainly concerned with cancer from those mainly concerned with Aids, since these topics were often closely intermixed.

The logged search took just under 24 minutes, although another 10 minutes should probably be added to the overall time to account for the abortive first attempt. Altogether 48 documents were examined, of which only 9 were officially relevant but 23 were selected for term extraction by the searcher. Relevant documents which were found referred to the laboratory stage of anti-cancer drug development: evaluation and marketing were barely touched on, as reflected by the terms output for the final query. This was a disappointing result, which contrasted with others which appeared to be more successful. Looking back over the log, the searcher could see many points at which her strategy could have been improved.

Following is a breakdown of command usage on this search:

Define	Join	Docset	Brief	Show	Extract	Remove
5	2	2	1	48	2	13

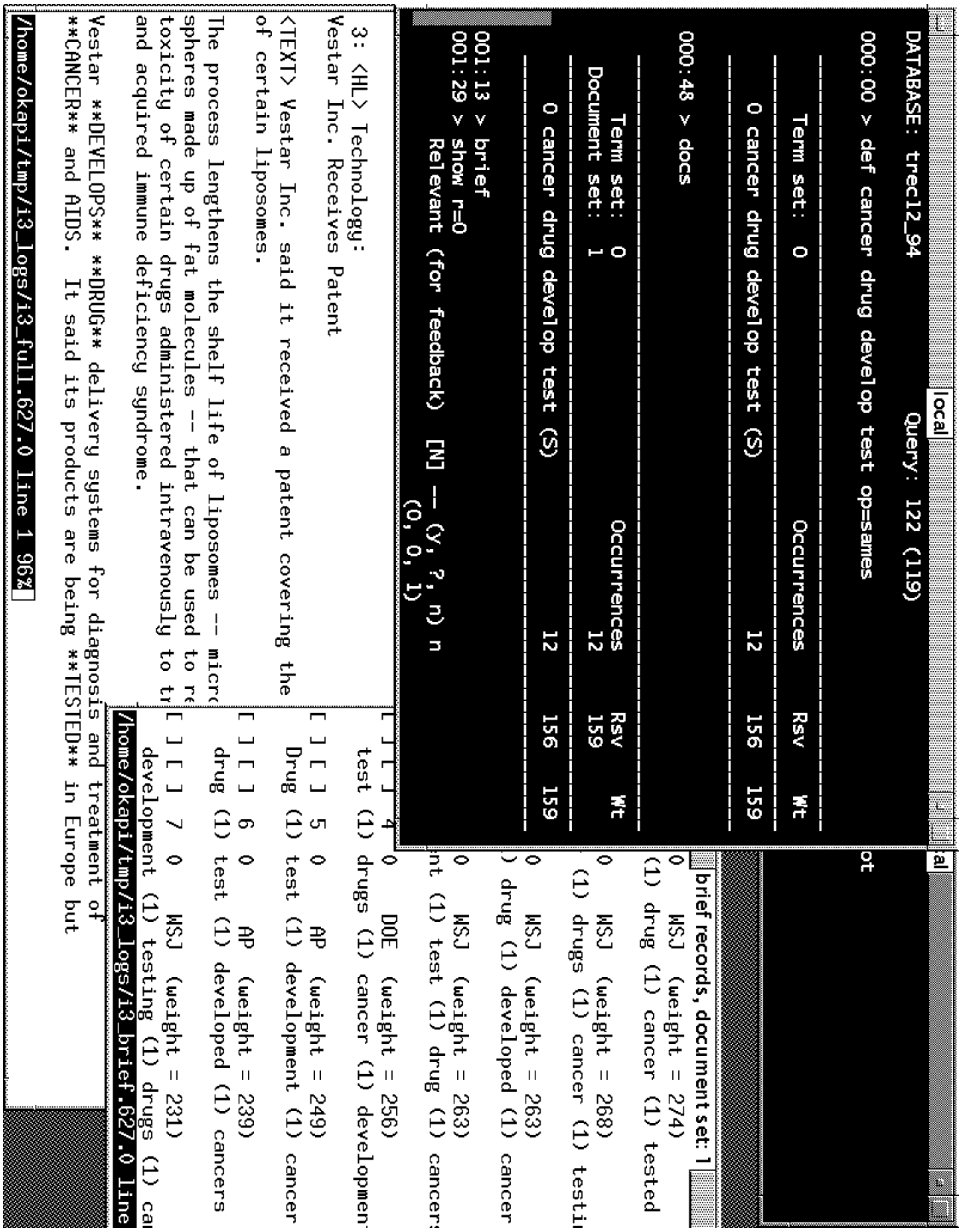

Figure 1: Interactive interface screen