Document Retrieval Using The MPS Information Server (A Report on the TREC-7 Experiment)

François Schiettecatte
(francois@fsconsult.com)
FS Consulting, Inc.
20709 Delta Drive, Laytonsville, MD, 20882
http://www.fsconsult.com/

1 Introduction

This paper summarizes the results of the work conducted by FS Consulting, Inc. as part of the Seventh Text Retrieval Experiment Conference (TREC-7). We participated in two components of TREC: (1) Category C, running the Ad Hoc experiments, producing two sets of official results (fsclt7m and fsclt7a) and (2) the Very Large Collection (VLC) track, running the entire experiment, producing the required official results as well as a set of unofficial results.

Our long-term research interest is in building information retrieval systems that help users find information to solve real-world problems. Our TREC runs employ two models: for the manual experiments, we have developed a model information systems end-user described more fully in section 5. For the automatic experiments, we use a title-word retrieval model.

2 Overview of FS Consulting TREC-7 Experiments

Our TREC-7 participation centered on two goals: comparing the effectiveness of manually generated queries versus automatically generated queries and evaluating the effectiveness and scalability of our search engine when working with very large collections of documents. In the TREC-7 experiments we set out to answer two questions:

- Are manually generated queries more effective than automatically generated queries?

For our first experiment, we produced two runs (fsclt7a and fsclt7m). For the first run (automatic), the system automatically constructed the queries from the topic title, producing the set of results (fsclt7a). In this run, the title words were connected together, using AND Boolean operators, into a query and submitted to the search engine. This approach reflects an assumption (admittedly false for some fields) that a title contains words that signal the content of the document. For the second run (manual), we employed a set of manually-produced query statements (fsclt7m). Our goal in this experiment is to deliver the highest percentage of relevant documents within the first 20 documents. This reflects our belief that in a 'live' search environment, the searcher would not scan more than @ 20 search hits before modifying or abandoning the search.

- What is the effectiveness and scalability of our search engine when dealing with very large collections?

For our second experiment, we indexed the VLC corpus into the three databases required by the track guidelines (Base-1, Base-10, and VLC100.) Because of its size and the capabilities of the equipment we used, the VLC2 corpus (100GB) was split up into four equally sized ‘buckets’ of data and indexed into four separate databases. These four databases were searched separately (in parallel) and the results were merged into a single result set.

The results merging algorithm has been developed and refined using the Ad Hoc corpora over the course of the TREC-4, 5 and 6 conferences. This algorithm has proved to work well in practice as was demonstrated in TREC-5 [2] and TREC-6 [3]. What differentiates our merging algorithm from many others is that it is a 'shared-nothing' algorithm where no statistics are shared between collections, neither are any global collection statistic gathered or generated prior to running any searches.

In addition to the required runs, which employed the automatic model described above, we ran our manual model against the same 3 databases. The purpose of this was to explore whether end users might need to adopt alternate search strategies when searching very large databases.

3 System Configuration

The MPS Information Server is a commercial full-text retrieval system that runs on a large number of Unix™ based platforms and on Windows NT™. Given a user query, the MPS system returns a list of relevance-ranked documents from a database. The system is capable of performing simple or complex term searching, phrase searching and proximity searching using parentheses, wildcards and Boolean operators. Soundex, typographical variation[1]. Fielded searches and numerical range searches are also supported. The system is designed to favor precision over recall when performing searches. Because it supports a number of different protocols, including WAIS-88, Z39.50-V2 (WAIS-V2 profile), STARTS and Gopher as well as two internal protocols, LWPS and Direct[2], the MPS Information Server is capable of responding to search requests from a wide variety of client applications.

The TREC-7 experiments employed version 4.5 of the MPS Information Server running on an UltraSparc 1/200 with 256 MB of memory and 160 GB of disk space. Four GB of disk space were set aside for the TREC data and index and 90 GB of disk space were set aside for the VLC2 data and index.

For the purposes of the experiments, we used a driver application that had been built for previous TREC experiments. Running on the UltraSparc, the driver application communicated with the MPS Information Server using the LWPS protocol. This driver application was designed to read TREC topic files, build a query by extracting a specific field, or fields, from the individual topic entries[3], run the queries against the MPS Information Server and save the query results in the TREC result format to a specified file. The result files could then be processed by the Trec Eval program to obtain the precision-recall values for that run.

The parser that was built to index TREC databases for the previous TREC experiments was reused with one modification, namely that we suppressed data present in a greater number of 'noise' fields. The rest of the documents were indexed as plain text, with the SGML tags extracted from the text, and the words stemmed using a plural stemmer. Word positions were extracted from the text to allow phrase and proximity searching if desired by the searcher. No other information was extracted from the text. All keywords in the news articles were suppressed as required by the guidelines.

[1] For example, missing letters, 'color' would also pick up 'colour', and juxtaposition of letters, 'animal' would also pick up 'ainmal'.
[2] LWPS is an inter-server communications protocol and Direct is a protocol which allows for rapid integration into front-end development application tools such as Perl or Tk/Tcl.
[3] In fact the queries formulated by the user are embedded into the TREC topic file and are marked up with SGML tags.

While the MPS Information Server's indexing application starts up with a default stop-word list (containing 547 words), it can be made to convert a word to a stop word if that word's total occurrence in the database reaches a specific threshold value. This threshold value would typically be set anywhere in the range of 20,000 to 500,000 occurrences and is site/collection dependent. For the TREC-7 experiments, we opted to turn off this threshold so that no additional words beyond the default list of 547 would be turned into stop words.

Four databases were created, namely the Trec-7 database, the Base-1, Base-10 and VLC100 database. Database sizes and build times were are follows:

Database:	**Size:**	**Build time:**
Trec-7	457MB	8 Hours
Base-1	237MB	1 Hour
Base-10	2.2GB	10 Hours
VLC100	20GB	96 Hours

It should be noted that the four databases making up the VLC100 database were searched by individual MPS Information Servers which were accessed by the MPS Information Server Gateway. This gateway presented the various physical databases as a single logical database to the driver application. The driver application was unaware of the fact that multiple physical databases were being searched and that multiple result sets were being merged, or where these databases were located. As explained in section 2 above, this is a 'shared-nothing' system so each MPS Information Server is searching a single collection.

During the year prior to running the TREC-7 experiments, we have been tuning the search algorithms using the TREC-6 data and relevance assessment in an effort to continually improve the search engine performance.

4 TREC-7 Results for FS Consulting Experiments

4.1 Searcher Model and Guidelines

One person constructed all query statements for the experiments. The initial parameters of the searcher 'model' were defined as follows:

- s/he regularly uses Internet search engines and/or library catalogs;
- s/he regularly uses search engines in the work setting;
- s/he may have some search training, but is not a professional searcher;
- s/he dislikes reviewing large search outputs;
- s/he is seeking information to solve a real-life problem;
- s/he may not be a content expert in the topic area of a given question.

The following instructions guided query formulation:

- prepare a simple search statement that will capture the most relevant documents for a given topic;
- use single or multiple terms, employing wild-card capability to capture multiple versions of a word, and/or quotes around several words (e.g., "cardiac arrest") to create a fixed phrase;
- apply Boolean logic as desired, using AND, OR or NOT operators. Create nested statements using parentheses if desired;
- no other databases or thesauri are available for consultation;

· the total time taken to prepare a single query should not exceed 5 minutes

4.2 First Experiment: Ad Hoc Automatic Run

4.2.1 Query formulations

The query formulation strategy for the Ad Hoc Automatic (fsclt7a) run created a set of title terms connected by Boolean AND operators. In the TREC-6 [3] experiments, we employed a strategy of using OR to connect the title terms; this new approach produced better results.

The following are some examples of the query formulations:

Topic 352: british AND chunnel AND impact

Topic 369: anorexia AND nervosa AND bulimia

Topic 386: teaching AND disabled AND children

4.2.2 Server Performance

The results for fsclt7a produced the following precision/recall figures over all of the topics:

```
Queryid (Num):       all  fsclt7a
Total number of documents over all queries
    Retrieved:     9471
    Relevant:      4674
    Rel_ret:        950
Interpolated Recall - Precision Averages:
    at 0.00       0.6078
    at 0.10       0.3026
    at 0.20       0.2080
    at 0.30       0.1367
    at 0.40       0.1001
    at 0.50       0.0639
    at 0.60       0.0339
    at 0.70       0.0236
    at 0.80       0.0228
    at 0.90       0.0208
    at 1.00       0.0132
Average precision (non-interpolated) over all rel docs
                  0.1146
Precision:
  At    5 docs:   0.3720
  At   10 docs:   0.3380
  At   15 docs:   0.3027
  At   20 docs:   0.2820
  At   30 docs:   0.2433
  At  100 docs:   0.1184
  At  200 docs:   0.0715
  At  500 docs:   0.0354
  At 1000 docs:   0.0190
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.1678
```

Overall, for all topics, 20% of the relevant documents were retrieved from the database and only 10% of the documents retrieved were relevant in fsclt7a.

4.3 Second Experiment: Ad Hoc Manual Run

This searcher created manual questions for our TREC 4, 5, and 6 experiments. We reviewed results of earlier TREC manual runs and made slight adjustments to the approach. Earlier experiments showed that using more than 3 or 4 topics in a search statement often resulted in few hits. For TREC-7, the searcher was conscientious about keeping the topic phrases below 4. For TREC-7, the searcher was allowed to review a preliminary list of the 'top 20' hits before settling on a final query statement.

4.3.1 Query formulations

The following examples are typical formulations:

Topic 352: (chunnel OR "channel tunnel")

Topic 369: anorexi* OR bulimi*

Topic 386: (child OR Children OR youth) (disabl* OR disabilit* OR handicap*) (teach* OR educat*)

Most query formulations employed parentheses, wildcards and the AND and OR Boolean operators. As the examples indicate, not all capabilities of the system were employed (e.g., field searching, proximity searching beyond the user of phrases, soundex, typographical variation and "NOT" operators were not used for example). The bounded phrase was the most common feature used.

Past experience influenced the searcher's query formulation behavior in the following ways:

- she preferred to use the wild-card capability selectively to increase recall, rather than entering multiple forms of a word;
- in some situations, she continued to use multiple synonym sets, believing that it would increase recall in a selective fashion and/or counter the impact of the stop-word list;
- in topics whose subject included a unique or unusual word, she used a single term rather than including all concepts in the topics.

4.3.2 Server Performance

The results for the Ad Hoc Manual run, fsclt7m, produced the following precision/recall figures over all of the topics:

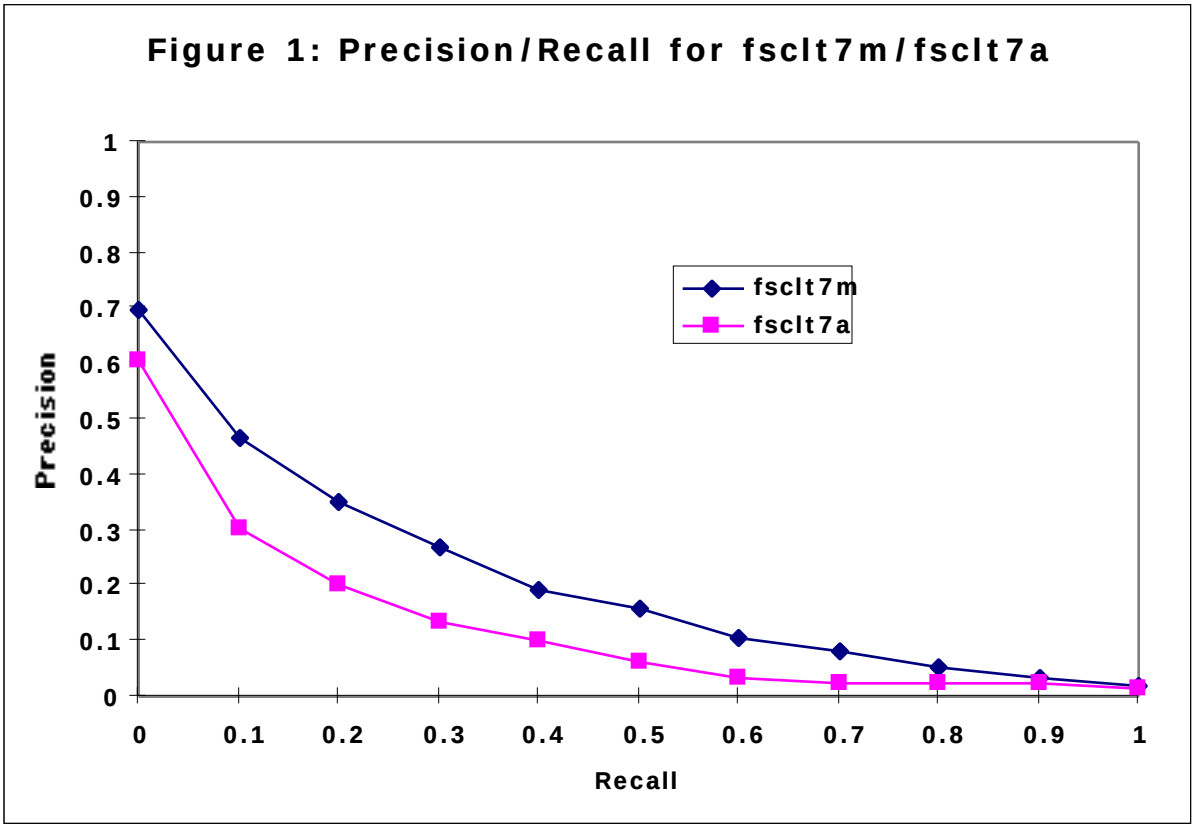
```
Queryid (Num):       all  fsclt7m
Total number of documents over all queries
    Retrieved:    18968
    Relevant:      4674
    Rel_ret:       1849
Interpolated Recall - Precision Averages:
    at 0.00       0.6951
    at 0.10       0.4667
    at 0.20       0.3526
    at 0.30       0.2709
    at 0.40       0.1944
    at 0.50       0.1582
    at 0.60       0.1061
    at 0.70       0.0806
    at 0.80       0.0527
    at 0.90       0.0314
    at 1.00       0.0179
```

```
Average precision (non-interpolated) over all rel docs
                0.1961
Precision:
  At    5 docs:   0.4600
  At   10 docs:   0.4160
  At   15 docs:   0.3947
  At   20 docs:   0.3810
  At   30 docs:   0.3333
  At  100 docs:   0.1962
  At  200 docs:   0.1270
  At  500 docs:   0.0648
  At 1000 docs:   0.0370
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.2562
```

Overall, for all topics, 39% of the relevant documents were retrieved from the database and only 10% of the documents retrieved were relevant in fsclt7m. If we compare these results with our TREC 6 [3] results, we see a marked improvement of recall (up from 28%) and a very small drop in precision (down from 11%). The precision at 20 documents also went up from .3120 to 0.3810.

4.4 Discussion of Ad Hoc Runs

Figure 1 below shows the precision-recall curve for fsclt7m and fsclt7a.



These results indicate that using the title words alone does not work as well as using a manually-constructed query, even if the words in the title are ANDed together. However if we compare these results with our TREC-6 results [3], we find that ANDing the title words together works better than ORing them.

5 Second Experiment: Very Large Database Track (VLC2)

In the second experiment, we ran all the official runs required by the VLC track, namely automatic runs on the Base-1, Base-10 and VLC100 databases. In addition to these we also ran a set of unofficial runs using the manual queries created in the first experiment. As noted earlier, our purpose here was to explore the need for end users to adopt different search strategies when searching large databases.

The official runs were labeled as follows:

	Base-1	Base-10	VLC100
Automatic run	fsclt7a-v1	fsclt7a-v10	fsclt7a-v100
Manual run	fsclt7m-v1	fsclt7m-v10	fsclt7m-v100

Note that in the official runs, only the top 20 document were retrieved, limiting the total number of documents retrieved to 1000 over all 50 runs. Unfortunately a small, but problematic, operator error in setting up the driver application resulted in truncating the maximum number of documents retrieved to 19 as opposed to 20 which explains why only 950 documents were retrieved for each run as opposed to 1000. The analysis below centers only on fsclt7a-v100 and fsclt7m-v100.

A second set of unofficial runs was also submitted; in these runs, the top 100 documents were retrieved (making them a little more comparable to the Ad Hoc runs). These runs were labeled as follows:

	Base-1	Base-10	VLC100
Automatic run	fsclt7a-v1a	fsclt7a-v10a	fsclt7a-v100a
Manual run	fsclt7m-v1a	fsclt7m-v10a	fsclt7m-v100a

These unofficial runs were submitted to the VLC track relevance assessors in the hope that they might be judged, but it was unknown if they had been assessed at the time this paper was sent into NIST. No extended analysis of these runs is reported here, but we do summarize the results in section 5.3.

5.1 Results for VLC2

As with the Ad Hoc runs, we submitted results from an automatic and a manual run, using the same query formulation models.

5.1.1 First Experiment: Ad Hoc Manual Run

The results for fsclt7m-v100, the manually-constructed queries, produced the following precision / recall figures for all the topics:

```
Queryid (Num):       fsclt7m-v100
Total number of documents over all queries
    Retrieved:      950
    Relevant:      4440
    Rel_ret:        255
Interpolated Recall - Precision Averages:
    at 0.00       0.5142
    at 0.10       0.0967
    at 0.20       0.0188
    at 0.30       0.0188
    at 0.40       0.0188
```

```
    at 0.50       0.0000
    at 0.60       0.0000
    at 0.70       0.0000
    at 0.80       0.0000
    at 0.90       0.0000
    at 1.00       0.0000
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.0387
Precision:
  At    5 docs:   0.3080
  At   10 docs:   0.2880
  At   15 docs:   0.2720
  At   20 docs:   0.2550
  At   30 docs:   0.1700
  At  100 docs:   0.0510
  At  200 docs:   0.0255
  At  500 docs:   0.0102
  At 1000 docs:   0.0051
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.0629
```

Overall, for all topics in the fsclt7m-v100 run, only 5% of the relevant documents were retrieved from the database and 23.7% of the documents retrieved were relevant.

5.1.2 First Experiment: Ad Hoc Automatic Run

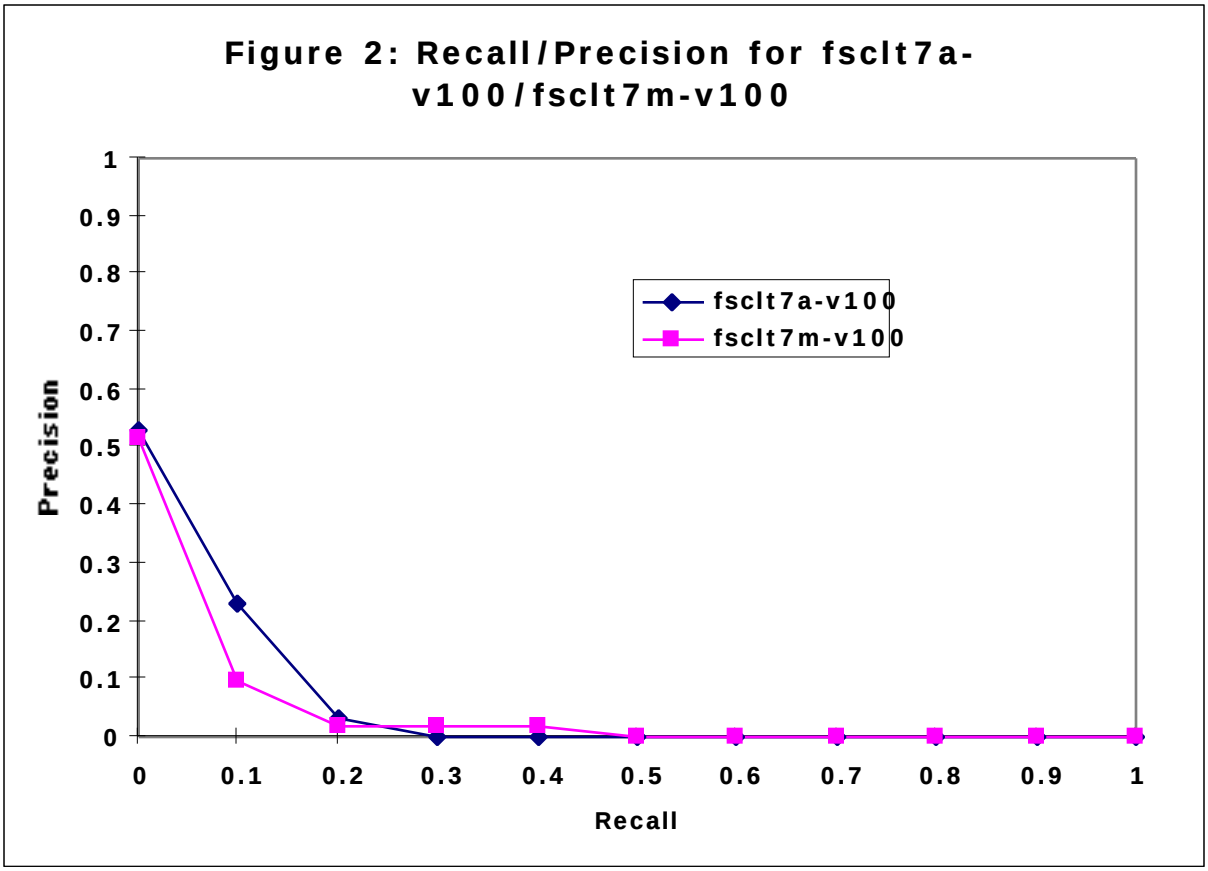
The results for fsclt7a-v100, the run employing automatically-constructed queries, produced the following precision / recall figures for all the topics:

```
Queryid (Num):       fsclt7a-v100
Total number of documents over all queries
    Retrieved:      941
    Relevant:      4440
    Rel_ret:        345
Interpolated Recall - Precision Averages:
    at 0.00       0.5284
    at 0.10       0.2288
    at 0.20       0.0303
    at 0.30       0.0000
    at 0.40       0.0000
    at 0.50       0.0000
    at 0.60       0.0000
    at 0.70       0.0000
    at 0.80       0.0000
    at 0.90       0.0000
    at 1.00       0.0000
Average precision (non-interpolated) for all rel docs(averaged over queries)
                  0.0472
Precision:
  At    5 docs:   0.3760
  At   10 docs:   0.3760
  At   15 docs:   0.3800
  At   20 docs:   0.3450
  At   30 docs:   0.2300
  At  100 docs:   0.0690
  At  200 docs:   0.0345
  At  500 docs:   0.0138
  At 1000 docs:   0.0069
R-Precision (precision after R (= num_rel for a query) docs retrieved):
    Exact:        0.0763
```

Overall, for all topics, only 7% of the relevant documents were retrieved from the database and 37% of the documents retrieved were relevant.

5.2 Discussion of VLC2 Track Results

Figure 2 below show the precision-recall curve for fsclt7a-v100 and fsclt7m-v100.



What is interesting to note is that these results are the opposite those obtained in Ad Hoc experiments. In those runs, the manually-constructed queries obtained better results; in this experiment, the opposite is true. In particular, the number of relevant documents retrieved is lower at every cluster of documents (5, 10, etc.) One possible explanation is because the queries were geared towards recall rather than precision and those kinds of queries might do well with smaller databases (such as 2GB), but break down when dealing with very large amounts of data (such as 100GB). From these results it seems clear that a different search strategy is needed when searching very large databases. We expected to get better results than these and we will be investigating why they are so disappointing over the next few months.

5.3 Additional results

In addition to the searches required by the VLC track, we also ran a number of unofficial searches as summarized below:

Run	Number of topics	Retrieved	Relevant	Relevant retrieved	P@20
fsclt7a-v1	47	692	4315	128	0.1362

fsclt7a-v10	50	885	4440	268	0.2680
fsclt7a-v100	50	941	4440	345	0.3450
fsclt7a-v1a	47	1859	4315	186	0.1340
fsclt7a-v10a	50	4275	4440	566	0.2590
fsclt7a-v100a	50	4746	4440	938	0.2850

These results show that raising the maximum number of documents retrieved with each query to 100 document increased the recall as one might well expect, but the numbers are still disappointing.

6 Life with great exponents

6.1 Data, data everywhere

TREC provides us with 2GB databases with which to experiment. While this could prove to be challenging 3 or 4 years ago, the current availability of cheap large capacity disks have negated this challenge. The 100GB database provided by the VLC track bring back most, if not all, these challenges. Even compressed, the data still require 35GB of disk space to store. Five 9.1 GB disks were needed for the data, five 9.1 GB disks for the indices and three 9.1 GB disks for temporary space required for index creation.

6.1 Time immemorial

Another challenge was the time required performing searches. The VLC100 database was bigger than any database we had searched before and we needed to do some software optimizations to attain the time required to meet the challenge set out by the VLC 2 track and hope to get the 'gold' medal.

The first task was to profile the software. Bottlenecks in software have a tendency to 'move' from function to function (or method to method depending on the language one uses) in any application when that application is subjected to a wide variety of tests. This was the case with our search engine when the amount of data that had to be searched was increased fifty-fold. Functions that used to be fast when dealing with small amounts of data can turn into major bottlenecks when dealing with very large amounts of data.

Prior to tuning the code, the VLC100 automatic run required 43 minutes to complete, after tuning the time required had fallen to 25 minutes.

6.2 I/O, I/O, it's off to work we go

Disk I/O is quite possibly the slowest link in the computing chain. Looking at the I/O statistics that were generated by the Sun Microsystems performance measurement program 'iostat'[4] showed that the system was doing a lot of swapping while running the queries. This is to be expected when searching such a large database on a system with a modest amount of memory (256MB)[5].

[4] 'iostat' is a Sun Microsystems tool which is shipped as part of the Solaris operating system, it allows one to monitor and measure the transaction rate of various devices in the computer. It is especially useful to identify bottlenecks on a computer.

[5] We realize that 256MB may not seem to be a modest amount of memory for most applications, the searching of very large databases benefit from having a much as memory as possible. It is now quite common to find machines with very large amounts of memory in them.

A trick gleaned from Adrian Cockcroft's excellent book on performance tuning [4], is to split the swap space required by the operating system across two separate disk drives rather than putting it all on a single disk drive as is the convention on Unix systems. The optimal solution would be to put the swap space on two separate disk drives each managed by separate disk controllers. The optimal solution was impractical in our setup so we settled for splitting the swap space in two equal parts across two separate disk drives, both working off the same disk controller. Doing that reduced the time it took to complete the run from 25 minutes to 20 minutes, better but still not good enough for the 'gold' medal.

6.3 Thanks for the memories

We then monitored the machine using the SE performance monitoring tool built by Adrian Cockcroft [4] and it showed that the machine was suffering from memory starvation once it got to the fourth or fifth query in the run. In technical terms, the pager was stealing memory pages too rapidly. In layman's terms, memory holding data that had just been read was being released prematurely because there was not enough physical memory to accommodate it in the amounts and speeds with which it was being read from the disks as the run was being executed. Upgrading the memory in the machine from 256MB to 768MB further reduced the time it took to complete the run from 20 minutes to 10 minutes, still not within reach of the 'gold' medal.

6.4 Together we stand

Shortly before the TREC conference, we were able to reorganize disks by striping[6] a number of them together to create a single virtual device using a Sun product called DiskSuite[7]. We created two virtual devices, one for the data (52GB) and one for the indices (78GB). This allowed us to create a single VLC100 database rather than create four separate ones. The MPS Information Server has generally been faster when searching a single large database than searching multiple small databases on a single CPU machine, so we were hopeful that this would allow us to further reduce the time. We were also hopeful that any performance impact that would be incurred by using DiskSuite would be offset by performance gains achieved by searching a single large database. We were proved wrong in our assumption, the time to complete the run increased from 10 minutes to 13 minutes.

6.5 So, where do we stand?

It seems clear it will be difficult to get many more performance improvements from the machines we are using. During the optimization process described above, the code was profiled and tuned several times, memory has been added, a single virtual database as been created. Thus, while the time to complete a run was reduced from 43 minutes to 10 minutes which is in itself no mean feat, we still don't qualify for the 'gold' medal in terms of search time.

[6] Striping disks together allows files to be laid out in stripes across a number of disks rather than contiguously on a single disk. The theory behind this idea is one can read the file faster as one is reading data from multiple disks as opposed to a single one, so one benefits from a certain amount of parallelism.

[7] DiskSuite is a software RAID solution from Sun Microsystems. It supports some basic RAID features such as device concatenation, device stripping and device mirroring. Sun is very non-committal about the performance impact of this product, only suggesting that performance might be improved in some situations and worsen in other while not providing any firm guidelines on how best to set up the devices based on their use.

The next step would be to migrate to a multi-processor setup and split the database across a number of disks working off separate disk controllers. Obvious solutions here would be to use a cluster of machines (such as fast PCs with fast disks and a fast interconnection network) or a multi-CPU system (with fast disks and multiple disk controllers). It also seems clear that using a software RAID solution (such as DiskSuite) does not really provide any performance gains at all thereby confirming Sun's assertion that performance gains may vary based on the application. The alternative would be to use a hardware based RAID system, though we don't have the means to do that.

7 Discussion of FS Consulting TREC-7 Results

The MPS Information Server is designed to operate in an interactive setting, where quick response and high precision are generally preferable to high recall. These results are close to our TREC-6 results but the recall is somewhat improved because of the queries.

For the Ad Hoc track, using manually generated queries worked better than using the title words alone. The VLC track results where just the opposite, the title words generated better results than the manually generated queries. As we mentioned above, we are not quite sure why this was the case, but we will be investigating why this was the case over the next few months. Additionally we are not sure why the results were so disappointing, but we will be investigating that as well.

8 Future Work

We will be working on the following areas prior to our participation in TREC 8:

- See if we can improve the precision of the Ad Hoc results.
- Investigate why the VLC results are so poor.
- See if we can further optimize the search engine to reduce the time required to run the searches.

9 References

[1] Schiettecatte, François and Florance, Valerie, "Document Retrieval Using the MPS Information Server". In Harman D. (Ed) The Fourth Text Retrieval Conference (TREC-4). National Institute of Standards and Technology Special Publication 500-236, Gaithersburg, Md. 20899.

[2] Schiettecatte, François, "Document Retrieval Using the MPS Information Server". In Harman D. (Ed) The Fifth Text Retrieval Conference (TREC-5). National Institute of Standards and Technology Special Publication,
Gaithersburg, Md. 20899.

[3] Schiettecatte, François, "Document Retrieval Using the MPS Information Server". In Harman D. (Ed) The Sixth Text Retrieval Conference (TREC-6). National Institute of Standards and Technology Special Publication,
Gaithersburg, Md. 20899.

[4] Sun Performance and Tuning, Adrian Cockcroft, Richard Pettit, Prentice Hall, 1998.