Queensland University of Technology at TREC 2005

Alan Woodley, Chengye Lu, Tony Sahama, John King and Shlomo Geva

Information Retrieval and Web Intelligent Group, Faculty of Information Technology,
Queensland University of Technology,
PO Box 2434, Brisbane 4001, Queensland, Australia.
{ap.woodley, c4.lu}@student.qut.edu.au | {t.sahama,j5.king,s.geva}@qut.edu.au

Abstract

The Information Retrieval and Web Intelligence (IR-WI) research group is a research team at the Faculty of Information Technology, QUT, Brisbane, Australia. The IR-WI group participated in the Terabyte and Robust track at TREC 2005, both for the first time. For the Robust track we applied our existing information retrieval system that was originally designed for use with structured (XML) retrieval to the domain of document retrieval. For the Terabyte track we experimented with an open source IR system, Zettair and performed two types of experiments. First, we compared Zettair's performance on both a high-powered supercomputer and a distributed system across seven midrange personal computers. Second, we compared Zettair's performance when a standard TREC title is used, compared with a natural language query, and a query expanded with synonyms. We compare the systems both in terms of efficiency and retrieval performance. Our results indicate that the distributed system is faster than the supercomputer, while slightly decreasing retrieval performance, and that natural language queries also slightly decrease retrieval performance, while our query expansion technique significantly decreased performance.

1.0 Introduction

Information Retrieval (IR) is one of the most influential and challenging fields of study in information technology. QUT's Information Retrieval and Web Intelligence (IR-WI) group is a team of researchers investigating IR and other associated technologies such as: data mining; web intelligence; and recommendation systems. In previous years our group has participated in other information retrieval workshops - most notably the Initiative for the Evaluation of XML Retrieval (INEX) - however, 2005 is the first year that we have participated in TREC. We focused our attention on the Terabyte and Robust tracks.

Our approach in the Terabyte track was different to most other TREC participants. Rather than produce our own information retrieval system, we decided to investigate the variation in performance of an open source search engine in two different scenarios, first, when executed on different hardware models, and second, when different queries are used as input. The search engine we used was the Zettair system, developed by the Search Engine Group at the Royal Melbourne Institute of Technology. Zettair is an open source search engine that was also used in the 2004 Terabyte track (Billbereck et al, 2004).

The first of our experiments applied Zettair to two different hardware models. The first hardware model was a high performance supercomputer that produced a single index. The second hardware model was a set of seven midrange personal computers, each of which produced a separate index. This allowed us to compare the performance of a power supercomputer versus a distributed system of standard personal computers, both in terms of efficiency and standard information retrieval metrics.

The second of our experiments was to observe the variation in Zettair's performance when different input was used. We used three input variations. First, as a baseline we used the terms contained in each of the topic's title tags. Second we used a natural language interface to derive keywords from each of the topics' description tags. Third, we augmented the topics' original title with plural/singular variations, stems and synonyms derived from the Porter stemmer (Porter,1980) and Wordnet (Fellbaum, 1998).

The majority of this paper is focused on our participation in the Terabyte track, with the exception of Section 6 that discusses our participation in the Robust track. Section 2 describes Zettair, the open source information retrieval system used for the experiments. Section 3 describes the experiments we performed detailing the two hardware models and the variations in input. Section 4 describes and presents results of the experiments we performed on the 2004 query set. Section 5 describes and presents the results of our 4 runs for the 2005 query set. Section 7 provides concluding remarks as well as a discussion on the future research we intend to perform.

2.0 The Open Source Information Retrieval System (Zettair)

Zettair is an open source IR system available under a BSD license from **http://www.seg.rmit.edu.au/zettair**. It participated in TREC for the first time in 2004 (Billbereck et al, 2004). Zettair can extract text from SGML-based languages. For indexing, it uses and efficient algorithm (Heinz & Zobel, 2003), however, it does not use in-place

merging (Moffett & Bell, 1995) and uses a variable-byte compression scheme instead of Golumb encoding (Scholer et al., 2002). The index saves the full word position, which allows for phrase searches. It uses a single pass indexing algorithm, which generates compressed postings in memory saves them to disk and then merges them together to form a single, continuous list with a B+Tree vocabulary structure. Zettair supports multiple ranking metrics, however, we used the Okpai BM25 metric (Jones et al., 2000) using the formula:

(1) $$bm25(q,d) = \sum_{t \in q} \log\left(\frac{N - f_t + 0.5}{f_t + 0.5}\right) \times \frac{(k_1 + 1) f_{d,t}}{K + f_{d,t}}$$

Where:

t = query terms
N = Number of documents in the collection
f_t = Number of documents that a term t occurs in
$f_{d,t}$ = Number of times that a term t occurs in document d

and

(2) $$K \text{ is } k_1((1-b) + b \times L_d / AL)$$

Where:

k_1 = 1.2
b = 0.75
L_d = Length of document d in bytes,
AL = Average document length over the collection

Note that in formula 1 term contributions for terms occurring in more than half of the documents in the collection are negative; hence, a small positive term contribution is used instead. Next we describe how we implemented Zettair on single index and distributed systems, and the input variations we used.

3.0 Research Methodology

The aim of our research was to observe the variation in performance of an open source information retrieval system when applied to different hardware models and when different sources of input are used. We used two hardware models: a single index created on a high performance supercomputer; and multiple indexes created on a distributed system of midrange personal computers. We used three different sources of input: the topics' title tag that was used as a baseline, keywords derived from the topics' description tag via a natural language interface; and the topics' title tag augmented with plural/singular variations, synonyms and stems. Here we describe our experiments in more detail.

3.1 Single Index System

The aim of the single index experiments was to test the feasibility of searching a (half) terabyte collection on a high performance computer. We used Queensland University of Technology's High Performance Computing (HPC) supercomputer for our experiments. The HPC supercomputer was purchased in 2000 for US$55,000. It consists of 10 nodes, each with two 3.4 Gigahertz processors and 4 Gigabytes of RAM. However, during our experiments we shared the HPC supercomputer with other users and processes, hence, we only had access to 4 of the possessors. We copied the terabyte collection onto the HPC supercomputer, separating it evenly amongst the 10 nodes. Then we executed the Zettair indexer on the collection as a whole and created a single 43.7 Gigabyte index.

3.2 System Distribution

The aim of system distribution experiments was to test the feasibility of searching a (half) terabyte collection on a network of standard personal computers. We tested our system on computers in one of our student computer laboratories. These computers cost about US$1,500 (3 GHz Pentium 4, with 1 Gigabyte RAM), and are a reasonable approximation of a midrange computer system that could be found in a home, school or office. We divided our system randomly into seven sub-collections. Each collection was stored on a 500 Gigabyte external hard drive (Lacie "Big Disk"). The hard drives were connected to the computers via USB 2 connection. Each sub-collection had 39 directories, totally 61 Gigabytes. We separated the collection into sub-collections randomly. Table 3.1 presents the directories contained within each sub-collection.

Sub-Collection	Start Directory	End Directory
1	0	38
2	39	77
3	78	116
4	117	155
5	156	194
6	195	233
7	234	272

Table 3.1. Sub-Collection Distribution

We executed the Zettair indexer to each of the sub-collections. This produced an index size between 4 Gigabytes and 10 Gigabytes per sub-collection. We then used Zettair to search each of indexes and saved the top 10,000 documents per topic per sub-collection to a results file. The time taken to index and search each sub-collection is presented in section 4.2. Finally, we had to merge the results file together to produce a submission file for comparison with the single index. Merging strategies have a well established history in information retrieval (Callan, 2000). We used two relatively simple strategies referred to as Relevance Merge and Round Robin Merge. For Relevance Merge we simply merged together the results generated by each sub-collection and sorted by their relevance score. Note that this score used only local - rather than global - information, therefore, all term weights were defined by statistics derived solely from each sub-collection. For Round Robin Merge we used a two stage ranking strategy. First we grouped together documents according to their rank in the original sub-collection. So all the documents ranked in position 1 were grouped together, followed by all documents ranked in position 2, and so on. Secondly, within each group documents were ranked according to their original (local) relevance score. After the merging, the top 10,000 documents were chosen from each algorithm were chosen to produce the submission, as per specification.

3.3 Natural Language Processing

The second set of experiments investigated the use of natural language processing (NLP). Specifically, the use of natural language queries (NLQs) to derive users' content requirements by using the description tags as input. There has been an extensive amount of research on the use of NLP in IR, both in TREC itself (Strzalkowski & Sparack Jones, 1996, Strzalkowski et al., 1997, Strzalkowski et al., 1998) and in IR in general (Strazlkowski,1999, K. Sparck Jones, 1999, A. F. Smeaton, 1999, D. D. Lewis and K. Sparck Jones, 1996). However, to handle the queries we used our own natural language system, NLPX (Woodley and Geva 2004, Woodley and Geva 2005). NLPX was designed specially for use in the processing of structured (XML) queries, and has previously participated in INEX's NLP track (Geva & Sahama, 2005). To our knowledge this is the first time that a NLP system specifically designed for XML-IR has been used in traditional document-level retrieval.

Handling structured NLQs is more complex than traditional NLQs, since the NLP system must derive both the content and structural need of users. Therefore, all we provide here is a summary of how we used NLPX to derive users' content requirement from NLQs. A more detailed description of how NLPX can be found in our earlier work. For our experiments we treated the TREC descriptions as if they were Content Only queries in INEX. First we augmented the NLQ with their part-of-speech tags using the Brill tagger (Brill, 1994). The Brill tagger is a grammatical rule-based tagger than has a performance comparable to most stochastic taggers (~95%). Then we derived important noun phrases from the NLQ using a set of queries derived from our previous work. The terms contained within these phrases were used as input for the Zettair search engine.

3.4 Query Expansion

Our final set of experiments investigated query expansion, that is, augmenting topics with additional query terms. Three methods of query expansion were investigated: plurals and singular expansion; stemming; and synonym expansion. Plural and singulars were added using lexical-based heuristics to determine the plural form of a singular term (and vice-versa). Wordnet (Fellbaum, 1998) was used to add stems and synonyms to the topics. Wordnet is a linguistic resource inspired by psycholinguistic theories of human lexical memory. Wordnet groups words are into 'synsets', each representing a separate concept. Stemming was performed in two steps. First the Porter stemmer (Porter, 1980) was performed on the existing query terms to derive each of their stems. Then the Wordnet database was searched to find terms with the same stem. Similarly, synonyms were added to the query by searching for terms in the Wordnet database that belonged to the same synset as the query terms. As outlined in Table 4.1, we used several different query expansions.

4.0 2004 Experiments

Prior to submitting official runs for the 2005 Terrabyte track, we experimented using the 2004 query set. By evaluating the runs using the standard TREC evaluation module (trec_eval) and recording timestamps, we were able to measure

how successful our approaches both in terms of system performance and efficiency. Furthermore, by analysing the performance of the 2004 experiments we were able to predict which experiments would be most successful and/or most interesting for submission as official 2005 runs. Here, we describe the experimental process in detail and present their results both in terms of efficiency and performance.

4.1 Experimental Process

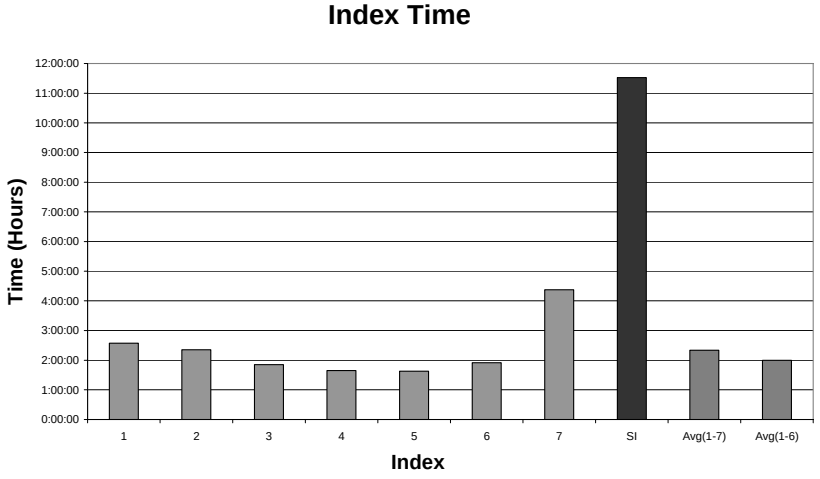
Our experiments can be categorized into two parts: frontend and backend. The frontend consisted of the queries input into Zettair from the natural language processing and query expansion experiments, while the backend consisted of the different hardware models used during the single index and system distribution experiments. To test the frontend we performed a baseline experiment (Base) consisting of the original terms in the topic, followed by an additional five (5) query expansions experiments as outlined in table 4.1. Each was these experiments was repeated for the original title terms and the terms derived from the description using NLPX, giving us a total of 12 query sets input into Zettair. To test the backend we input these queries into both our single index and distributed indexes. Our two different merging algorithms (Rank by Relevance and Round-Robin) were performance at a later stage. Overall this gave us a set of thirty-six (36) experiments performed on the 2004 query set.

Experiment Name	Description
Base	The baseline experiment consisting of the original topic terms
SP	The topic terms augmented with singular/plural derivatives
SP.Stem	The topic terms augmented with singular/plural derivatives and stems
SP.Syn	The topic terms augmented with singular/plural derivatives and synonyms
SP.Syn.Stem	The topic terms augmented with singular/plural derivatives, synonyms and stems
SP.Syn.Stem.SynStem	The topic terms augmented with singular/plural derivatives, synonyms, stems and the stems of the synonyms

Table 4.1. Query Expansion Experiments

4.2 Efficiency Results

Here we present the time it took to perform our experiments. There are two datasets that we focused on, the indexing time and querying time. Graph 4.1 and Table 4.2 present the indexing time, measured in hours, for each of the seven sub-collections as well as the single index (SI) system. Note that the seventh sub-collection took much longer – almost twice as long – to index than the other sub-collections. For this reason we have included two averages, one that is the average for the 7 sub-collections, while the second average excludes the seventh sub-collection. The results indicate that the distributed system is significantly faster than the single index system - around six times as fast. This is not unexpected, since each of the sub-collections only one-seventh of the original collection. However, one must consider that the single index was created on a high-end super computer that costs US$55,000 while the distributed system was created on seven mid-range personal computers that cost $US1,500 each or $US10,500 in total. Hence, the distributed system is much more economically efficient than the supercomputer.

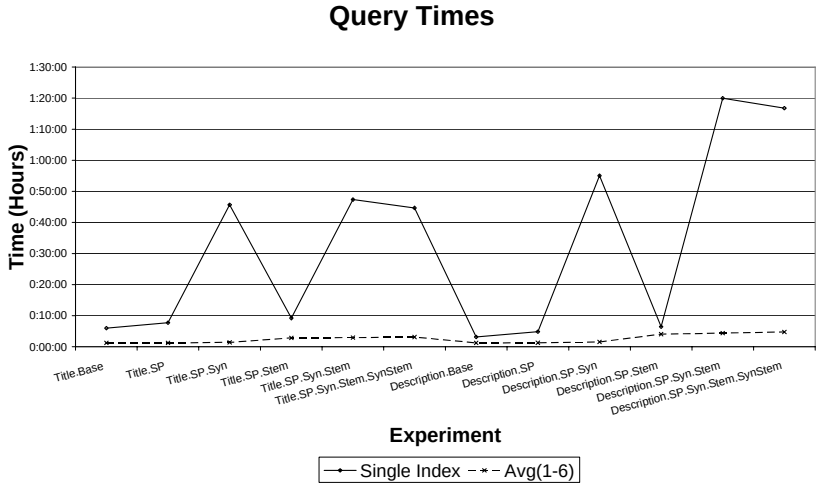


Graph 4.1. Indexing Times

Index	1	2	3	4	5	6	7	SI	Avg(1-7)	Avg(1-6)
Time (Hours)	2:34:15	2:21:02	1:50:55	1:39:03	1:37:38	1:54:56	4:22:28	11:31:30	2:20:02	1:59:38

Table 4.2. Indexing Times

Graph 4.2 and Table 4.3 present the querying time, measured in hours, for each of the seven sub-collections as well as the single index (SI) system. Once again the seventh sub-collection took much longer to query than the other sub-collections. Therefore we have again included two averages. For clarity Graph 4.2 only plots the times for single index and the average of sub-collections 1 - 6. Not surprisingly, the more terms augmented to the query, the longer it took to execute, hence, the queries augmented with synonyms execute slower than the queries without synonyms, a situation magnified for the single index system. To illustrate, Table 4.4 presents the ratio in query time between the supercomputer and the average query time for sub-collections 1 - 6. In queries that contain synonyms, highlighted in grey in Table 4.4, the distributed system was between 14 and 35 times faster than the single index system, while in queries without synonyms its was 'only' 1.5 to 6 times faster.



Graph 4.2. Query Times

Experiment Name	1	2	3	4	5	6	7	SI	Avg (1-7)	Avg (1-6)
Title.Base	01:22	01:23	01:11	01:04	01:07	01:10	02:15	05:59	01:22	01:13
Title.SP	01:26	01:24	01:08	01:07	01:12	01:14	02:31	07:45	01:26	01:15
Title.SP.Syn	01:40	01:33	01:22	01:15	01:17	01:20	02:43	45:40	01:36	01:24
Title.SP.Stem	03:59	03:20	02:20	02:23	02:22	02:43	16:15	09:12	04:46	02:51
Title.SP.Syn.Stem	04:06	03:30	02:28	02:23	02:28	02:55	17:34	47:23	05:03	02:58
Title.SP.Syn.Stem.SynStem	04:19	03:36	02:37	02:33	02:34	03:05	18:48	44:41	05:22	03:07
Description.Base	01:19	01:17	01:13	01:08	01:11	01:09	01:29	03:10	01:15	01:13
Description.SP	01:21	01:20	01:14	01:06	01:17	01:14	01:43	04:52	01:19	01:15
Description.SP.Syn	01:47	01:41	01:30	01:24	01:28	01:26	02:12	55:00	01:38	01:33
Description.SP.Stem	06:38	04:27	03:22	03:04	03:06	03:41	23:05	06:27	06:46	04:03
Description.SP.Syn.Stem	07:08	05:14	03:30	03:15	03:18	04:02	28:44	1:19:58	07:53	04:25
Description.SP.Syn.Stem.SynStem	07:47	05:48	03:38	03:29	03:32	04:14	28:46	1:16:48	08:11	04:45

Table 4.3. Query Times

Experiment Name	Ratio
Title.Base	4.94
Title.SP	6.18
Title.SP.Stem	3.22
Title.SP.Syn	32.43
Title.SP.Syn.Stem	15.94
Title.SP.Syn.Stem.SynStem	14.31
Description.Base	2.60
Description.SP	3.87
Description.SP.Stem	1.59
Description.SP.Syn	35.61
Description.SP.Syn.Stem	18.14
Description.SP.Syn.Stem.SynStem	16.19

Table 4.4. Ratio of Query Time of Single Index vs. Distributed System

4.3 Retrieval Performance Results

Tables 4.5, 4.6 and 4.7 present the performance results of the 2004 query set. We present results of the MAP, Bpref and number of relevant results at 10 documents, since these were the official metrics used in the 2005 Terabyte track. Here we discuss the relevant performance of the experiments across the entire 2004 query set.

Experiment Name	Single Index	Relevance Merge	Round Robin Merge
Title.Base	0.1642	0.1476	0.1473
Title.SP	0.1579	0.1394	0.1389
Title.SP.Syn	0.1038	0.0933	0.0930
Title.SP.Stem	0.1178	0.1149	0.1144
Title.SP.Syn.Stem	0.0877	0.0806	0.0804
Title.SP.Syn.Stem.SynStem	0.0677	0.0739	0.0737
Description.Base	0.1499	0.1412	0.1405
Description.SP	0.1452	0.1313	0.1306
Description.SP.Syn	0.1154	0.0686	0.0683
Description.SP.Stem	0.1040	0.1007	0.1000
Description.SP.Syn.Stem	0.0502	0.0497	0.0495
Description.SP.Syn.Stem.SynStem	0.0446	0.0399	0.0398

Table 4.5. MAP Results - 2004 Query Set

Experiment Name	Single Index	Relevance Merge	Round Robin Merge
Title.Base	0.2500	0.2579	0.2578
Title.SP	0.2709	0.2782	0.2775
Title.SP.Syn	0.2031	0.2221	0.2209
Title.SP.Stem	0.2191	0.2536	0.2507
Title.SP.Syn.Stem	0.178	0.2063	0.2061
Title.SP.Syn.Stem.SynStem	0.1521	0.2004	0.1999
Description.Base	0.2608	0.2718	0.2688
Description.SP	0.2678	0.2734	0.2708
Description.SP.Syn	0.1800	0.2106	0.2086
Description.SP.Stem	0.2104	0.2452	0.2420
Description.SP.Syn.Stem	0.1391	0.1826	0.1823
Description.SP.Syn.Stem.SynStem	0.1216	0.1607	0.1611

Table 4.6. Bpref Results - 2004 Query Set

Experiment Name	Single Index	Relevance Merge	Round Robin Merge
Title.Base	0.3898	0.3245	0.3245
Title.SP	0.3857	0.3694	0.3694
Title.SP.Syn	0.2939	0.2469	0.2469
Title.SP.Stem	0.2898	0.2980	0.2980
Title.SP.Syn.Stem	0.2653	0.2061	0.2061
Title.SP.Syn.Stem.SynStem	0.2347	0.1959	0.1959
Description.Base	0.3469	0.3061	0.3061
Description.SP	0.3249	0.3551	0.3551
Description.SP.Syn	0.2400	0.1857	0.1857
Description.SP.Stem	0.2837	0.2837	0.2837
Description.SP.Syn.Stem	0.1844	0.1408	0.1408
Description.SP.Syn.Stem.SynStem	0.1533	0.1143	0.1143

Table 4.7. P@10 Results - 2004 Query Set

4.3.1 Baseline

Overall, the best performing experiment was the baseline system that had a single index backend, used the title as input and did not perform any query expansion. While this could be described as disappointing, it was not unexpected since several of the techniques used are known to degrade performance.

4.3.2 System Distribution

The distributed systems outperformed the single index systems in the Bpref metric (avg. 15%), but the trend was the opposite for both the MAP metric (avg. 12%) and P@10 metric (16%). Furthermore, there was not much difference between the two types of merging algorithms, however, the Relevance Merge algorithm tended to slightly outperform the Round Robin Merge algorithm on the MAP and Bpref (<1%) metric. Interestingly, the two merging algorithms performed exactly the same under the P@10 metric.

4.3.3 Natural Language Processing

Generally, the standard ad-hoc (title as input) outperformed the natural language processing system (description as input) in the MAP (avg 30%), Bpref (avg 9%) and P@10% (avg 27%) metrics. The degradation was especially severe when natural language processing was combined with synonym query expansion. However, when no query expansion was used the natural language processing system outperformed the standard ad-hoc system in the Bpref metric (avg 4%).

4.3.4 Query Expansion

Query expansion tended to decrease performance dramatically in the MAP (86%) Bpref (30%), and P@10 (49%) metrics, with the exception of singular/plural expansion which outperformed the baseline using the Bpref (4%) and P@10 metrics (7%). The performance decrease of query expansion was far worse when the description was taken as input rather than the title, and when synonyms were added to the query rather than stems.

5.0 2005 Experiments

We conducted our experiments using the 2005 query set in the same manner as we did for the 2004 query set. Once again 36 experiments were conducted; however, only 4 of them were allowed to be submitted as official TREC runs. Here, we present 2005 runs and their results.

5.1 2005 Runs

We based our decision on which runs we could submit for TREC 2005 both on our analysis of our 2004 experiments, and the experiments we thought would provide the most interesting discussion at the workshop. In particular, we wanted to address the three main areas of our research: system distribution; natural language processing; and query expansion. Here we describe the runs, and outline our justification for selecting them.

5.1.1 Baseline (QUT05TBEn)

For out baseline system we had a single index backend, used the title as input and performed no query expansion. This was the best performing experiment in the corresponding 2004 query set and was the logical choice to use as a baseline for the other experiments.

5.1.2 System Distribution (QUT05TBMRel)

The first comparison we wanted to make was between a single index and distributed system. In order to make a valid comparison with our baseline system we used the same input (title without query expansion) and only changed the backend from a single index system to distributed system. We used the Relevance Rank merging method since that slightly outperformed the Round Robin method in our experiments on the 2004 query set.

5.1.3 Natural Language Processing (QUT05DBEn)

Our second comparison was between a system using natural language queries as input and a system using standard keywords input. Like our baseline system, we used a single index backend and performed no query expansion. As input we used the topics' description tags and parsed it our natural language processor – NLPX - to derive important content terms.

5.1.4 Query Expansion (QUT05TSynEn)

Our final comparison was between a system with a query expansion and a system without. Like the baseline system we used a single index backend, however, we performed both plural/singular and synonym expansion on the topic's title element. As with our 2004 experiments we used heuristics for plural/singular expansions and Wordnet to find query term synonyms.

5.2 2005 Results

Here we present the results of our official 2005 runs. Table 5.1 shows the overall MAP, Bpref and P@10 for each of our four runs, as well as the average maximum, minimum and median values of the other 2005 TREC participants.

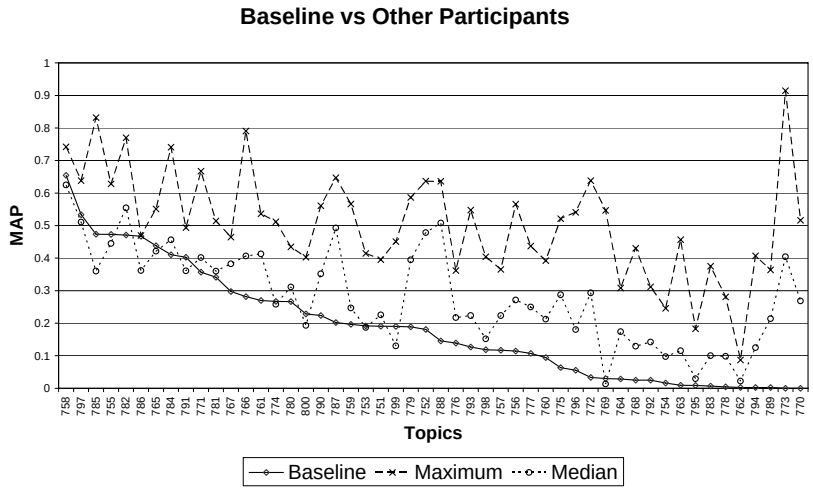
Run	MAP	Bpref	P@10
Baseline	0.1894	0.2100	0.5100
Distributed System	0.1645	0.2148	0.4100
Natural Language	0.1837	0.2071	0.3960
Query Expansion	0.0881	0.1370	0.3160
Maximum	0.5056	0.5236	0.9000
Median	0.2815	0.3030	0.5720
Minimum	0.0109	0.0256	0.0440

Table 5.1. MAP, Bpref, P@10 Results - 2005 Query Set

We also present three graphs that display a topic-by-topic MAP of the runs. Each of the graphs compares the baseline with other systems. We have sorted the graphs' topics according the baseline's MAP value rather than by topic number. Here, we discuss the performance of each system in comparison with the baseline, and outline the queries in which one system performed significantly better.

5.2.1 Baseline (QUT05TBEn)

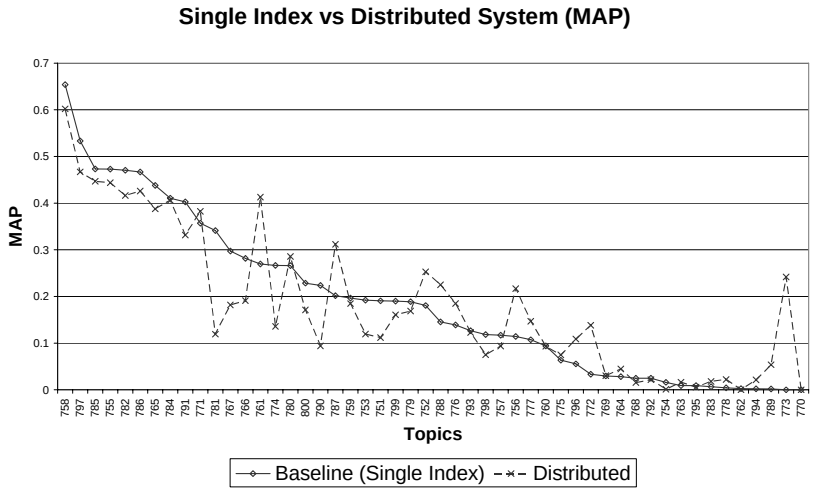
Graph 5.1 presents the MAP results for our baseline system in comparison with the maximum and median values of the other 2005 participants. This provides a guide to see how well the system performed in absolute terms against other participants. Generally, the baseline performed similarly to the median, however, it performed poorly against the maximum. As with our 2004 experiments the baseline system generally performed better than the other runs. However, there were some specific topics where the other runs outperformed the baseline.



Graph 5.1. Topic-by-Topic MAP – Baseline vs Other Participants

5.2.2 System Distribution (QUT05TBMRel)

Graph 5.2 presents the MAP results for our baseline system in comparison with the distributed system. Since the input for both systems was the same (title without any query expansion) this graph provides a means to observe the effect that different hardware models - single index and multiple indexes on a distributed system - have on system performance.

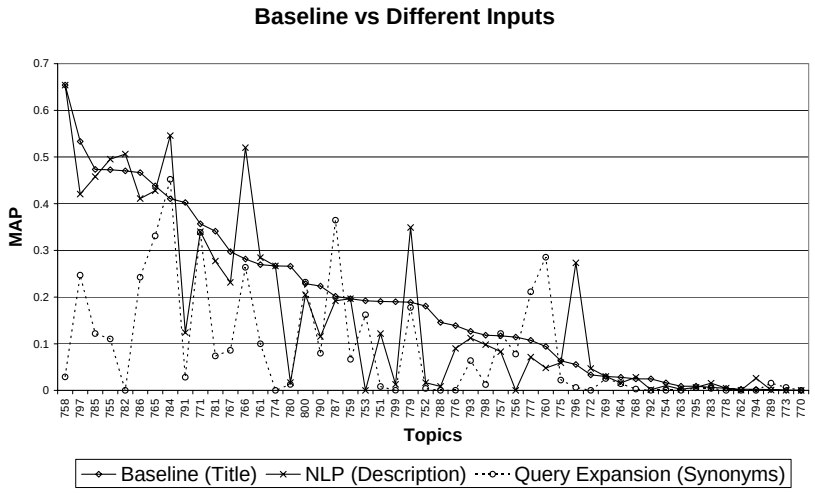


Graph 5.2. Topic-by-Topic MAP – Baseline vs. Distributed System

Overall the distributed system performed very well in comparison with the baseline system, with only a 3% decrease in MAP and 1 % decrease in Bpref; although, it did suffer a degradation of 24% under the P@10 metric. The lack of degradation under the MAP and Bpref was unanticipated given we divided our corpus randomly and that we used a naïve method of collection fusion. However, the high degradation at P@10, could mean that while the distributed system retrieved a similar number of relevant results as the single index system but ranked them lower. Interestingly, the distributed system significantly outperformed the baseline (>10% under MAP and Bpref) in topic numbers 752, 761, 773, 776, 787, and 788 however, it was significantly outperformed by the baseline systems in topic numbers 767, 774, 780, 781 and 800.

5.2.3 Natural Language Processing (QUT05DBEn)

Graph 5.3 presents the MAP results for our baseline system in comparison with the natural language processing and query expanded systems. Since the hardware model for all three systems was the same (single index), this graph provides a means to observe the effect that different inputs - title, keywords derived from a natural language query, and a query expanded with synonyms - have on system performance.



Graph 5.3. Topic-by-Topic MAP – Baseline vs. Different Inputs

Overall, the natural language system also compared favourably to the baseline system, outperforming it by 2% under the Bpref metric. However, under the MAP metric it suffered degradation of 15%, and under the P@10 metric a degradation of 29%. Unfortunately, performance degradation is common when using natural language processing and information retrieval (Smeaton, 1997). However, examples of topics that it significantly outperformed the baseline include 766, 779, 784 and 796 while examples of topics were it was outperformed by the baseline include 752, 780, 788, 791 and 799.

5.1.4 Query Expansion (QUT05TSynEn)

The query expansion run performed the worst in comparison with the other experiments. The baseline system outperformed it by 115% under the MAP metric, 53% under the Bpref metric and 61% under the P@10 metric (Graph 5.3). While these results were disappointing they were not completely unexpected given the naïve method in which synonyms were chosen for the query. If more advanced methods such as part of speech recognition or word sense disambiguation had been used then the performance may have improved. However, the experiment did outperform the baseline in a number of topics such as 760,765, 777 and 787 but was significantly outperformed by the baseline in topics such as 759, 767, 781, 790 and 791.

6.0 Robust Track Participation

In previous years, our group has participated in the Initiative for the Evaluation of XML Retrieval (INEX) with a dedicated XML search engine. This year we participated in the Robust track at TREC and are trying to discover the difference between document-level information and the XML information retrieval. As the TREC collection is "well formatted" XML-like (SGML) documents, we intended to use our indexer and search engine with only minor changes.

6.1 Indexing

The documents were indexed using an inverted file approach that was designed for XML retrieval. Term postings consist of XPath to the containing element, and position within the XPath context. For instance, the posting { /document[10]/body[1]/chapter[3]/section[5]/paragraph[2] , 23 } identifies the precise position of a term in an XML document with some self-explanatory structure. The indexer was originally developed for INEX and it is basically used for indexing XML documents. When indexing, the indexer will record the term, the term position in the context (context position), the term position in the article (global position), the context name (XPATH) and also the article ID. The words were stemmed using porter stemmer and stop words were removed from the index to reduce the size of index file. Our index file is in Microsoft access format. Due to the size limitation of access, each file must be less than 2GB. Thus, we split the whole collection into 5 sub-collections. We eventually ported the system to SQL Server (still distributed)

6.2 Searching

In our XML oriented search- the score of an element (in this case the DOC element) was computed using the following formula:

(3)
$$L = N^{n-1} \sum_{i=1}^{n} \frac{\log(1+t_i)}{\log(1+f_i)}$$

Here n is the number of unique query terms contained within the element. When a phrase is found n is incremented by the number of terms in the phrase, instead of 1. This rewards a phrase more heavily than a non-phrase set of the same term in the element. The term N is an integer - we used the value N=5. The term N^{n-1} scales up the score of elements having multiple distinct query terms, and phrases even more. The system is not sensitive to the value of N – we experimented with N=5 to 50 with little difference in results. The term f_i is the frequency of the i^{th} query term in the collection. The term t_i is the frequency of the i^{th} query term in the element.

The usual approach to phrase searching is based on term proximity. We implemented this in the usual manner. Because our search engine is geared towards finely grained XML documents, we also have a concept of a partial phrase -words that appear in the same context (say sentence or even paragraph) but do not strictly constitute a phrase are regarded as a partial phrase and will be given higher score. We treated partial phrases as phrases in this experiment.

6.3 Searching a distributed collection

The AQUAINT collection was vertically and randomly split into 5 sub-collections, each of which was searched independently. Finally the results were merged together. Results were ranked locally, but without normalizing scores – to allow meaningful comparisons between scores obtained from searching different partitions of the collection. The underlying assumption is that global collection statistics are similar with large sub-collections and the non-thematic vertical split. This assumption is not entirely accurate, and there is a small tradeoff in reduced precision for increased speed which can be very high in a federated collection setting. We leave out the discussion of distributed searches since it is outside the scope of this Robust track investigation.

6.4 2005 results

The official submission performed very poorly. Our MAP is only 0.0294. This was surprising and so after analyzing the results we discovered a bug in the Indexer, introduced when changing from INEX to TREC, and from Microsoft Access to SQL Server. Unfortunately about 20% of the term postings were lost. Regretfully we only just met the submission deadlines and so did not fix the problem. Discussion of these results is meaningless. Therefore, we discuss results that we have subsequently obtained after fixing the indexer and re-indexing the collection, with the official qrels, as shown in table 6.1.

	QUT_Official	QUT_Corrected	TREC Median
MAP	0.0294	0.1613	0.2239
P10	0.1200	0.3840	0.4340

Table 6.1. MAP, P10 Results New index vs. old vs. Median

The intended submission has a much better result, albeit still below the median result over all submissions TREC.

The TREC collection contains many documents that have no fine grained structure – just a single large <text> element. On the other hand, typical XML documents contain fine structure – for instance, the INEX XML collection contains paragraph sized XML leaves. Our node scoring approach to XML documents aims for retrieval of specific nodes, not entire documents. It exploits granularity by rewarding nodes that contain more of the query terms. This is usually done in text retrieval through proximity scoring of one kind or another – which our ranking system does not apply since it assumes fine grained XML elements. Since the TREC collection contains numerous DOC elements that are very large, often a document will contain a few or all of the query terms, but not in the same context. Our XML ranking strategy does not account for this and will therefore find many false positive results. Consider for instance topic 404, with precision details as the table below.

Position	P5	P10	P15	P20	P30	P100	P200	P500	P1000
Precision	0.6000	0.5000	0.4000	0.3500	0.400	0.3700	0.3050	0.2420	0.1570
Documents	3	5	6	7	12	37	61	121	157

Table 6.2: Precision of topic 404

The search engine fails to find many relevant documents until P30. The query for 404 is “talk, peace, Ireland”. Our search engine will give higher rank to documents that have more matches over the terms “peace”, “talks”, and even “peace talks”, but not necessarily in the context of “Ireland”. Since a strict phrase containing all 3 rarely occurs – even in the correct context - our search engine relies on the fine granularity of XML text elements to implicitly identify when the 3 occur in the same context (say sentence or paragraph.). As the terms “talk” and “peace” are very common in the collection many irrelevant documents will be ranked highly and this strategy fails with the structure-less TREC documents.

6.4 Query Expansion

In our experiment, two methods of query expansion were investigated: plural/singular expansion and Porter stemming. Plurals and singulars were added using lexical-based heuristics to determine the plural form of a singular term (and vice-versa.) Porter stemming was performed on the query terms and retrieval was then based on term stems rather than on the query terms. The results are shown as table 6.3.

	No expansion	Plural and Singular	Porter stemming
MAP	0.1448	0.1578	0.1613
P10	0.3740	0.3860	0.3840

Table 6.1. MAP, P10 Results with expansion

Searches with plural/singular expansion and with porter stemming provide similar performance. The Porter stemmer did provide some benefit, but slowed down the search by increasing the number of term postings accessed. Topic by topic examination reveals the usual behaviour – sometimes the stemmer improves precision and sometimes it degrades it. For instance, in topic 408, we are locking for “tropical storms” where the term “tropical” shares a Porter stem with “tropicality”, “tropicalization”, “tropicalize”, “tropically”, “tropicals”. However most of these are irrelevant to our topic. This may or may not lead to precision penalties, but will always lead to performance penalties. In this case (408) the average precision is 0.0710 for porter stemming and 0.0961 in plural/singulars. With more advanced methods such as part of speech analysis or word sense disambiguation performance may be improved by stemming – we intend to study this in future evaluations.

7.0 Conclusion

We performed a set of experiments on an existing open source information retrieval system. We performed two sets of experiments. Our first set of experiments compared the performance of Zettair on a high performance supercomputer with a distributed system of seven midrange personal computers. Our results indicate that the distributed system was more efficient than the supercomputer, both in terms of speed and economics, we have been able to achieve comparable retrieval performance. Our second set of experiments used three different set of inputs: a standard TREC title; a natural language query and an expanded query. Our results indicate that the natural language query had a retrieval performance comparable with the standard title while the expanded query was significantly worse. Interestingly however, all of the experiments outperformed the baseline in some topics. We will continue to research these areas and in particular investigate the topics where the experiments outperformed the baseline.

8.0 Acknowledgments

We would like to thank the staff at the High Performance Computing laboratory, in particular Ashley Wright and Mark Barry, without whom we would not have been able to complete this research. We would also like to thank Daniel Tao who assisted us with laboratory experiments. Finally, we would like to thank the QUT's School of Software Engineering and Data Communications and School of Information Systems for funding part of this project.

References

Billbereck, B., Cannane, A., Chattaraj, A., Lester, N., webber, W., Williams, H. E., Yiannis, Y., Zobel, J., RMIT University at TREC 2004. In. *The Thirteenth Text REtrieval Conference (TREC-13)*. NIST Special Publication 500-261, National Institute of Standards and Technology, Gaithersburg, MD. 2004.

Brill, E., Some Advances in Transformation-Based Part of Speech Tagging. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)*, Seattle, Washington, United States, 1994.

Callan, J., Distributed information retrieval. In W. B. Croft, editor, *Advances in Information Retrieval*, chapter 5, pp. 127-150. Kluwer Academic Publishers, 2000.

Fellbaum, C., *WordNet: An Electronic Lexical Database*. The MIT Press, 1998.

Geva, S., Sahama, T., The NLP task at INEX 2004. In *SIGIR Forum* **39**(1). pp. 50-53 2005

Heinz, S. & Zobel, J., .Efficient single-pass index construction for text databases., In *Journal of the American Society for Information Science and Technology* **54**(8), pp. 713 -729, 2003.

Jones, K. S., Walker, S. & Robertson, S. E., A probabilistic model of information retrieval: development and comparative experiments. Parts 1&2., *Information Processing & Management* **36**(6), pp. 779 – 840, 2000.

Porter, M.F., An algorithm for suffix stripping, In *Program*, **14**(3). pp. 130-137, 1980.

Lewis, D. D. and Sparck Jones, K., Natural Language Processing for Information Retrieval, *Communications of the ACM*, 39(1):92-101, 1996

Moffat, A. & Bell, T. A. H. In-situ generation of compressed inverted files. In *Journal of the American Society of Information Science* **46**(7), pp. 537.550, 1995.

Scholer, F., Williams, H. E., Yiannis, J. & Zobel, J. Compression of inverted indexes for fast query evaluation. In *Proc. ACM-SIGIR International Conference on Research and Development in Information Ret*rieval., Tampere, Finland, pp. 222 - 229. 2002.

Smeaton, A. F. Using NLP or NLP Resources for Information Retrieval Tasks. In Strzalkowski (ed), *Natural Language Information Retrieval*, Kluwer Academic Publisher, Dordrecht, NL, pp. 99-111. 1999.

Smeaton. A. F., Information Retrieval: Still Butting Heads with Natural Language Processing? In M. Pazienza (ed). *Information Extraction – A Multidisciplinary Approach to an Emerging Information Technology*. pp. 115-138, Springer-Verlag, 1997

Sparck Jones, K. What is the role of NLP in text retrieval? In Strzalkowski (ed) *Natural Language Information Retrieval*, Kluwer Academic Publisher, Dordrecht, NL, pp. 1-24. 1999

Strazlkowski, T., editor, *Natural Language Information Retrieval*. Kluwer Academic Publisher, Dordrecht, NL, 1999.

Strzalkowski, T., Lin, F., Carballo, J. P. Natural Language Information Retrieval TREC-6 Report. In V Voorhees E. M. and Harman D. K (editors). *The Sixth Text REtrieval Conference (TREC-6).* NIST Special Publication 500-240, National Institute of Standards and Technology, Gaithersburg, MD. pp. 347-366. 1997

Strzalkowski, T., Sparck Jones, K. NLP Track at TREC-5. In Voorhees E. M. and Harman D. K (editors). *The Fifth Text REtrieval Conference (TREC-5)*. NIST Special Publication 500-238, National Institute of Standards and Technology, Gaithersburg, MD. 1996.

Strzalkowski, T., Stein, G. C., Wise, G. B., Carballo, J. P., Tapanainen, P., Järvinen, T., Voutilainen, A., Karlgren, J.. Natural Language Information Retrieval: TREC-7 Report. In Voorhees E. M. and Harman D. K (editors). *The Seventh Text REtrieval Conference (TREC-7)*. NIST Special Publication 500-242, NIST, MD. pp.164-173. 1998.

Woodley A. and, Geva, S., "NLPX– An XML-IR System with a Natural Language Interface" , In Proceedings of the Australasian Document Computing Symposium, Melbourne, Australia, December 13 2004, pp. 71-74.

Woodley, A. and Geva, S., "NLPX at INEX 2004", In *Advances in XML Information Retrieval: Third International Workshop of the Initiative for the Evaluation of XML Retrieval, INEX 2004*, Dagstuhl, Germany, December 6–8, 2004, Revised Selected Papers, Berlin: Springer (LNCS : 3493). 2005.