The Mirror DBMS at TREC-9

Arjen P. de Vries

CWI, Amsterdam, The Netherlands

Abstract

The Mirror DBMS is a prototype database system especially designed for multimedia and web retrieval. From a database perspective, this year's purpose has been to check whether we can get sufficient efficiency on the larger data set used in TREC-9. From an IR perspective, the experiments are limited to rather primitive web-retrieval, teaching us that web-retrieval is (un?-)fortunately not just retrieving text from a different data source. We report on some limited (and disappointing) experiments in an attempt to benefit from the manually assigned data in the metatags. We further discuss observations with respect to the effectiveness of title-only topics.

1 Introduction

The Mirror DBMS [ ] combines content management and data management in a single system. The main advantage of such integration is the facility to combine IR with traditional data retrieval. Furthermore, IR researchers can experiment more easily with new retrieval models, using and combining various sources of information. The IR retrieval model is completely integrated in the Mirror DBMS database architecture, emphasizing efficient set-oriented query processing. The logical layer of its architecture supports a nested object algebra called Moa; the physical layer uses the Monet main-memory DBMS and its MIL query language [ ]. Experiments performed in last year's evaluation are described in [ ]; its support for IR is presented in detail in [ ] and [ ].

The main goal of this year's participation in TREC has been to migrate from plain text retrieval to retrieving web documents, and simultaneously improve our algorithms to handle the significantly larger collections. The paper is organized as follows. Section details our lab environment. Section interprets our results and discusses our plans for next year with the Mirror DBMS, followed by conclusions.

```
<doc docno='WTX044-B44-57' docoldno='IA044-000798-B032-287'>
<title>bootnet reviews</title>
<description> ... </description>
<keywords> ... </keywords>
<body>
compaq ambitious delivers impressive design performance and features compaq presario so
near yet so far win means the issue of sound blaster compatibility is dead and buried right
wrong ...
</body>
<img>card cage</img>
</doc>
```

Figure 1: The intermediate format produced (XML).

2 Lab Environment

This section discusses the processing of the WT10g data collection used to produce our runs. The hardware platform running the experiments is a (dedicated) dual Pentium III 600 MHz, running Linux, with 1 Gb of main memory and 100 Gb of disk space.

Adapting our existing IR setup to handling Web data caused much more trouble than expected. As a side-effect of this problem, the submitted runs contained some errors, and even fixing does not give us the best runs ever; a lot of work still remains to be done to improve our current platform.

Managing a new document collection and getting it indexed has turned out to be a rather timeconsuming problem. Obviously, the WT10g is 10 times larger than TREC, so we decided to treat it as a collection of 104 subcollections following the layout on the compact discs. But, handling a collection *of this size* was not the real issue; our main problems related to the 'quality' of data gathered from the web.

2.1 Parsing

After some initial naive efforts to hack a home-grown HTML parser, we bailed out and used the readily available Perl package `HTML::Parser`. It is pretty good at 'correcting' bad HTML on-the-fly; the only real problem we bumped into was that it assumes a document to always have at least a `<HEAD>` and a `<BODY>`, which fails on `WTX089-B33`.

We convert the sloppy HTML documents into (rather simple) XML documents that are easier to manage in the subsequent indexing steps. We keep the 'normal' content words, the content of `<IMG>`'s `ALT` attribute, as well as the following meta tags: `keywords`, `description`, `classification`, `abstract`, `author`, `build`. In this first step, we also normalize the textual data to some extent by converting to lower-case and throwing out 'strange characters'; unfortunately, due to working against a very tight schedule (*too* tight), this included the removal of all punctuation and numeric characters (not helping topics referring to a particular year, like topics 456 and 481). An example result file is shown in Figure 1.

What affected our results severely is our assumption that HTML documents are nicely wrapped in `<HTML>` and `</HTML>` tags. Unfortunately, this first 'bug' removed about half of the collection from our index.

2.2 Indexing

The second step reads the intermediate XML files and converts them into load tables for our database system. The contents of the `title`, `body`, and `img` tags are unioned together in 'term' sets, and all other tags in 'meta' sets.[1] Notice that we do not *have* to union these tags together; but, any alternative requires an IR model that can handle fields properly, which is still beyond our skill.

After loading these tables, the complete collection is represented by the following schema:

```
define WT10g_docs as
SET<
  SET<
    TUPLE<
      Atomic<string>         : docno,
      SET< Atomic<string> > : term,
      SET< Atomic<string> > : meta
    >
  >: subCollection
>;
```

The Mirror DBMS supports efficient ranking using the CONTREP *domain-specific* Moa *structures*, specialized in IR. These structures now have to be created from the above representation; this indexing procedure is still implemented in a separate indexing MIL script which is directly fed into Monet, the DBMS. The script performs stopping, stemming, creates the global statistics, and creates the internally used $< d_j,\ t_i,\ tf_{i,j} >$-tuples.

Web-data is quite different from newspaper articles: the strangest terms can be found in the indexing vocabulary after stopping and stemming. Examples vary from 'yippieyayeheee' and the like, to complete sentences lacking spaces between the words. After quick inspection, we decided to prune the vocabulary aggressively: all words longer than 20 characters are plainly removed from the indexing vocabulary, as well as all words containing a sequence of more than four identical characters.[2]

2.3 Retrieval

After running the indexing script, we obtain the following schema that can be used in Moa expressions to perform ranking:

[1] Notice that these 'sets' are really *multi-sets* or *bags*.

[2] We realize that this is a rather drastic ad-hoc approach; it is not likely to survive into the codebase of next year.

```
define WT10g_index as
SET<
  SET<
    TUPLE<
      Atomic<string>: docno,
      CONTREP       : term,
      CONTREP       : meta
    >
  >: subCollection
>;
```

The Monet database containing both the parsed web documents and its index takes 9 Gb of disk space.

Like in TREC-8, we use Hiemstra's LMM retrieval model (see also [Hie00] and our technical report [HdV00]). It builds a simple statistical language model for each document in the collection. The probability that a query $T_1, T_2, \cdots, T_n$ of length n is generated by the language model of the document with identifier D is defined by the following equation:

$$P(T_1 = t_1, \cdots, T_n = t_n | D = d) = \prod_{i=1}^{n} (\alpha_1 \frac{df(t_i)}{\sum_t df(t)} + \alpha_2 \frac{tf(t_i, d)}{\sum_t tf(t, d)}) \tag{1}$$

The getBL-operator (i.e. get beliefs) defined for the CONTREP structure ranks documents according to this retrieval model. We planned two series of experiments: ranking using the raw content collected in the 'term' sets, and ranking using a weighted combination from the first ranking with the ranking based on the annotations extracted from the meta tags collected in the 'meta' sets. These two experiments are (approximately) described by the following Moa expressions:[3]

```
(1) flatten(map[map[TUPLE<THIS.docno,
      sum(getBL(THIS.term, termstat, query))>](THIS)](WT10g_index));

(2) map[TUPLE< THIS.docno, THIS.termBel + 0.1*THIS.metaBel >](
    flatten(map[
        map[ TUPLE< THIS.docno,
          sum(getBL(THIS.term, termstat, query)): termBel,
          sum(getBL(THIS.meta, metastat, query)): metaBel >
        ](THIS)
      ]( WT10g_index )));
```

We do not expect the reader to grasp the full meaning of these queries, but only intend to give an overall impression; the inner map computes the conditional probabilities for

[3]Details like sorting and selecting the top ranked documents have been left out.

documents in a subcollection, which are accessed with the outer `map`; the `flattens` remove nesting. Similar to TREC-8, the query plan generated by the Moa rewriter (in MIL) has been manually edited to loop over the 50 topics, log the computed ranking for each topic, and use two additional tables, one with precomputed normalized inverse document frequencies (a materialized view), and one with the document-specific constants for normalizing the term frequencies. The `flatten` and the outer `map`, which iterate over the 104 subcollections and merge the results, were written directly in MIL as well. Unfortunately, we introduced a second (real) bug in this merging phase, which messed up our main results: we used `slice(1,1000)` whereas this really selects from the 2nd up to the 1001st ranking document; hence throwing out the 104 'best' documents (best according to our model).

3 Discussion

Table 1 presents a summary of the average precision (AP, as reported by `trec_eval`) measured on our runs. The first column has the results in which the top 100 documents are missing; the second column has the fixed results.[4]

run name	description	AP	AP (fixed)
CWI0000	title	0.0176	0.1814
CWI0001	title & description	0.0174	0.1503
CWI0002	title & description & narrative	0.0122	0.1081
CWI0010	title (term & meta)	0.0125	–

Table 1: Result summary.

Very surprising is the fact that using the description and/or narrative has not been helpful at all. This is completely different from our experience with evaluating TREC-6 and TREC-8 topics. Closer examination shows that the description (and sometimes the narrative) help significantly for the following topics:

- 452: 'do beavers live in salt water?'. Here, the description adds more general words such as 'habitat';
- 455: the description specifies 'major league' and the narrative gives finally the desired 'baseball';
- 476: the description adds the scope ('television' and 'movies') to title 'Jennifer Aniston';
- 478: The description adds 'mayor' to 'Baltimore'

[4]The run with combined term and meta data has not been fixed, due to some strange software problems that we have not figured out *yet*.

- 498: The title does not mention 'how many' and 'cost' which reveal the real information need.

Precision drops however in most other cases; especially the very concise title queries 485 ('gps clock'), 497 ('orchid') and 499 ('pool cue') suffer from overspecification in description and narrative. For example, topic 486 shows that the 'casino' from the title is not weighted enough in comparison to generic terms like 'Eldorado'. It warrants further investigation to view if query term weighting would address this counter-intuitive result.

We have not succeeded to make effective use of the information collected in the meta tags. Using *only* the meta tags leads to a poor average precision of 0.0033. Closer investigation of topics 451, 492, 494, for which the meta tags do retrieve relevant documents in the top 10, it turns out that these are also ranked high using the bare document content. Thus, we can only conclude that in our current approach, the meta tags can safely be ignored. Despite of this disappointing experience, we hope still that using this information source may be more benificial for processing blind relevance feedback.

Table 2 shows the results after spell-checking the topics semi-automatically using a combination of `ispell` and common sense, showing that this would have a minor positive effect on the title-only queries. Results for topics 463 (Tartin), 475 (compostion), and 487 (angioplast7) improve significantly; but, splitting 'nativityscenes' in topic 464 causes a loss in precision, because the (Porter) stemmer reduces 'nativity' to 'nativ'. We conclude from the other runs with spelled topics that we should address proper weighting of title terms first.

run name	description	AP
CWI0000s	title	0.1924
CWI0001s	title & description	0.1525
CWI0002s	title & description & narrative	0.1085

Table 2: Results with spell-checked topics.

4 Conclusions

The honest conclusion of this year's evaluation should be that we underestimated the problem of handling Web data. Surprising is the performance of the title-only queries doing better than queries including description or even narrative. It seems that the web-track topics are really different from the previous TREC topics in the ad-hoc task, for which we never weighted title terms different from description or narrative.

For next year, our primary goal will be to improve the current indexing situation. The indexing process can be described declaratively using the notion of *feature grammars* described in [dVWAK00]. Also, we will split the indexing vocabulary in a 'trusted' vocabulary (based on a proper dictionary), a numeric and named entity collection, and a 'trash' dictionary with rare and possibly non-sense terms. A third goal should be evident from the following quote from last year's TREC paper:

> Next year, the Mirror DBMS should be ready to participate in the large WEB track.

In other words: we still intend to tackle the large web track. For our cross-lingual work in CLEF [dV00] we have to address the problems of spelling errors and named entities anyways, which is directly related to some of our TREC problems. Some other ideas include to work on using links, blind relevance feedback, as well as improve our understanding on how to effectively exploit the 'meta' sets.

Acknowledgements

Henk Ernst Blok did a great just-in-time job of getting a late delivery of hardware prepared for running these experiments. Further thanks go to Djoerd Hiemstra and Niels Nes for their support with IR models and Monet respectively.

References

[BK99] P.A. Boncz and M.L. Kersten. MIL primitives for querying a fragmented world. *The VLDB Journal*, 8(2):101–119, 1999.

[dV98] A.P. de Vries. Mirror: Multimedia query processing in extensible databases. In *Proceedings of the fourteenth Twente workshop on language technology (TWLT14): Language Technology in Multimedia Information Retrieval*, pages 37–48, Enschede, The Netherlands, December 1998.

[dV99] A.P. de Vries. *Content and multimedia database management systems.* PhD thesis, University of Twente, Enschede, The Netherlands, December 1999.

[dV00] A.P. de Vries. A poor man's approach to CLEF. In *CLEF 2000: Workshop on cross-language information retrieval and evaluation*, Lisbon, Portugal, September 2000. Working Notes.

[dVH99] A.P. de Vries and D. Hiemstra. The Mirror DBMS at TREC-8. In *Proceedings of the Eighth Text Retrieval Conference TREC-8*, number 500–246 in NIST Special publications, pages 725–734, Gaithersburg, Maryland, November 1999.

[dVW99] A.P. de Vries and A.N. Wilschut. On the integration of IR and databases. In *Database issues in multimedia; short paper proceedings, international conference on database semantics (DS-8)*, pages 16–31, Rotorua, New Zealand, January 1999.

[dVWAK00] A.P. de Vries, M. Windhouwer, P.M.G. Apers, and M. Kersten. Information access in multimedia databases based on feature models. *New Generation Computing*, 18(4):323–339, August 2000.

[HdV00] Djoerd Hiemstra and Arjen de Vries. Relating the new language models of information retrieval to the traditional retrieval models. Technical Report TR–CTIT–00–09, Centre for Telematics and Information Technology, May 2000.

[Hie00] D. Hiemstra. A probabilistic justification for using tf·idf term weighting in information retrieval. *International Journal on Digital Libraries*, 3(2):131–139, 2000.