Challenging Ubiquitous Inverted Files

Arjen P. de Vries[1,2]
[1] CWI, Amsterdam, The Netherlands
[2] University of Twente, Enschede, The Netherlands
arjen@acm.org

Abstract

Stand-alone ranking systems based on highly optimized inverted file structures are generally considered 'the' solution for building search engines. Observing various developments in software and hardware, we argue however that IR research faces a complex engineering problem in the quest for more flexible yet efficient retrieval systems. We propose to base the development of retrieval systems on 'the database approach': mapping high-level declarative specifications of the retrieval process into efficient query plans. We present the Mirror DBMS as a prototype implementation of a retrieval system based on this approach.

1 Introduction

Searching collections by their content (be it text, images, or true multimedia content) has become a common requirement in software systems, as a result of successes achieved in the information retrieval (IR) research field, the demand for (better) search engines on the WWW, and the continuously growing amount of digitized photo footage presented online.

Inverted file structures are widely used in the implementation of stand-alone IR systems, and have proven highly efficient for this purpose [BYRN99]. The inverted file structure consists of two parts: the *indexing vocabulary* and the *postings file*. The indexing vocabulary contains a pointer into the postings file for each term in the collection; the postings file stores the *occurrence lists*: the document identifiers and (usually) the term positions within the documents. At retrieval time, the vocabulary is searched for the query terms, their occurrence lists are retrieved, and each document's rank is computed.

Extensive research and evaluation on real-life test data favors specialized inverted file structures over other approaches like signature files. For further optimization, the occurrence lists may be compressed to reduce the I/O effort of retrieving these lists [WMB94]. Also, terms with high document frequency may be treated especially, to take advantage of early cut-offs and possibly prune the search even further by producing only an approximate ranking (see e.g. [Bro95]).

2 Changing Requirements

We may conclude that inverted files are the best known approach to support the ranking process *in stand-alone IR systems* (see also [ZMR98]). But, it remains an open question whether this ranking process should really take place in a stand-alone fashion. The following developments in software systems challenge the pragmatic engineering practice of coupling otherwise stand-alone retrieval systems (hence inverted files):

- (semi-)structured data collections;
- new (multimedia) data types;
- data mining and machine learning.

The common problem underlying these three developments is that 'conventional' ranking has to be combined with other constraints; based on the structure of text documents, on the information extracted from various media (or various media representations), or through additional information induced in the query process. A well-known example of the latter, though relatively simple, is the incooperation of relevance feedback in the retrieval process; it is the author's strong belief that more complex learning algorithms will become increasingly important in improving the quality of IR systems.

Aside from these changes related to software developments, the more and more complex hardware layout of computer systems adds extra requirements to be met in any resource-consuming application (not just IR):

- parallel and distributed processing;
- cache-aware memory access.

The price of (cheap) personal computers compared to (expensive) professional workstations stimulates distribution of work over 'workstation farms', taking advantage of shared-nothing parallelism. Similarly, server machines with two up to eight processors are not extremely expensive any more, making it more attractive to explore strategies that exploit shared-memory parallelism.

Due to the important role of cache memory in modern CPUs, memory access cannot be thought as random access any more: sequential access patterns can be many times faster than random accesses [RAK99]. Since memory latency is *not* improving with Moore's law (unlike other properties of computer architecture), this problem will gain importance.

3 Implications for IR systems

Summarizing, we identify two classes of changing requirements for information retrieval systems. On the one hand, we notice a widening scope of applications in which information retrieval plays a role; on the other hand, taking full advantage of the computing power available in modern systems becomes ever more complex. For two reasons, these observations force us to rethink the idea of information retrieval as a completely separate, stand-alone component in a system architecture: flexibility and efficiency.

Highly optimized stand-alone systems are, naturally, not very flexible. Experiments, e.g. with new models or adaptive learning strategies, require changes in complex low-level code, with the danger of affecting the correctness of the results. To circumvent such inflexibility, it is common practice to wrap the core IR system in a *black-box*, and implement additional features on top. For example, when processing relevance feedback, systems typically rank the documents with the initial query in a first pass, and then re-rank with an adapted query in a second pass. For this particular case, [JFS98] has shown that the resulting retrieval system is not optimal with respect to efficiency, unless we address buffer management while taking both passes into account. We have to change the inner workings of the original system for optimal performance of the full system. In other words, *we must break open the black-box*. This, obviously, conflicts with our previously stated desire for flexibility.

We suspect however that opening the black-box of the ranking system becomes increasingly important with the development of more complex retrieval systems. In a multimedia retrieval system that ranks its objects using various representations of content (such as the system described in [VDdV00]), the number of independent black-box components that may contribute to the final ranking equals the number of feature spaces used in the system. It seems unlikely that computing these multiple rankings independently (i.e. without taking intermediate results into account) is the most efficient approach.

Considering that we should also optimize our systems for parallel and distributed computing and memory access patterns, usage of black-box abstractions to obtain flexibility becomes ever less desirable: it leads easily to inefficient systems, as we do not *really* understand what happens inside the ranking process. We are caught in a trap: gaining flexibility through abstraction causes an efficiency penalty which is felt most when we exploit this flexibility in new applications of IR or explore improvements upon existing models.

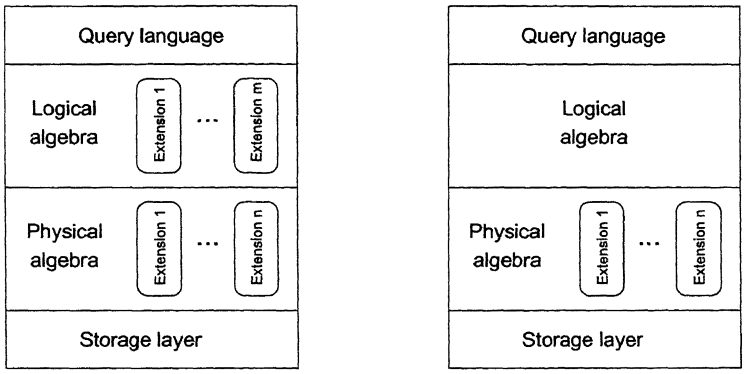

Figure 1: Comparing a Multi-Model DBMS (left) to an Object-Relational DBMS (right).

4 A Database Approach to IR

We seek a solution out of this impasse (between flexibility and efficiency) in 'the database approach'. Database technology provides flexibility by expressing requests in high-level, declarative query languages at the conceptual level, independent from implementation details such as file formats and access structures (thus emphasizing *data independence*). Efficiency is obtained in the mapping process from the declarative specification (describing what should happen) into a query plan at the physical level (describing how it happens). The query optimizer generates a number of equivalent query plans, and selects a (hopefully) efficient plan using some heuristics.

There is not much consensus on how the integration of IR techniques in general-purpose database management systems (DBMSs) should take place. State-of-the-art solutions simply make new functions available in the query language of object-relational database management systems (OR-DBMSs); these functions interface to the otherwise still stand-alone software systems. Therefore, extending an OR-DBMS with an information retrieval module means no more than again 'wrapping' the IR system as a black-box inside the DBMS architecture (see also Figure 1). Apart from seriously handicapping the query optimizer, the IR module must handle parallelism and data distribution by itself. Therefore, adapting an OR-DBMS for the changing requirements identified in Section 2 may even be *more* complex than enhancing the stand-alone IR system.

We propose the architecture of the Mirror DBMS as a better alternative. It has been especially designed to enable the integration of databases and (multimedia) information retrieval [dV99]. As shown in Figure 1, the DBMS can not only be extended at physical level, but also at the logical level. This enables a strong notion of data independence between the logical and physical levels.

We term this architecture with *Multi-Model DBMS*, since the data model used at the logical level can be different from that at the physical level. In the prototype implementation, we use the Moa nested object algebra at the logical level, and the Monet DBMS at the physical level [BWK98]. Domain-specific knowledge about IR is introduced using Moa's extensibility with domain-specific structures. These Moa extensions encode the mapping from high-level operations like 'compute term belief' and 'aggregate term beliefs' into manipulations specified in Monet's binary relational algebra. For more detailed information about this mapping, the interested reader is referred to [dV99].

Defining IR techniques at the logical level of the DBMS provides a desirable separation of concerns. Operations defined at the physical level support facilities for shared-memory parallel processing, and can take full advantage of modern the CPU cache usage plays a key role in obtaining (and losing) efficiency [BMK99]. The mapping from extensions at the logical level into the physical level is the appropriate place to encode domain-specific knowledge to be exploited for shared-nothing parallelism, such as the Zipfian distribution of terms typical for IR.

In the current implementation of the Mirror DBMS, ranked retrieval may already be combined with querying on structured data. Also, the system demonstrated reasonable efficiency on straightforward IR at TREC-8 [dVH99]; this year, a simple dictionary-based approach to cross-lingual IR was implemented for CLEF'00 [dV00]. The system's flexibility has proven very useful in a series of experiments performed to compare the traditional retrieval models to the new language models in IR [HdV00].

5 Discussion

This Section describes two examples to clarify the advantages of a database approach to IR; we do however point out that the optimization process described here is not yet implemented in our system.

To emphasize the differences between query processing in the Mirror DBMS and the standard black-box approach, we zoom in on a specific sub-problem of the topics addressed so far: the efficient processing of queries integrating text retrieval with structured document queries. An iconic example of an information need requiring such query processing is 'recent English newspaper articles about Willem-Alexander dating Maxima'. The terms 'recent', 'English' and 'newspaper article' refer to attributes of a structured document collection, whereas the aboutness-clause is presumably best processed by information retrieval techniques. Assume that we know the user is satisfied with an interpretation in which 'recent' is equivalent to 'published during the last month'.

In the traditional black-box approach, we index the text of the articles with an IR system, manage the structured data in a DBMS, and maintain a connection between the two systems using the document identifiers. Processing of queries combining the two types of data is addressed in a shallow layer on top of these systems. First, we use the IR system to rank the document collection using a query 'Willem-Alexander dating Maxima'. Next, we restrict the result set with the constraints specified on the attributes. Ranking is very efficient if we assume a highly optimized inverted file structure. Since we manage the structured data in a 'normal' DBMS, the selection of document identifiers of recent English newspaper articles is also quite efficient. The final step computes a semi-join between the ranked list and these document identifiers, to throw out the not-so-recent, non-English non-newspaper documents. Probably, ranking the documents in the IR system is the dominant cost in this process.

Of course, the same strategy could be performed in the Mirror DBMS. But, a query optimizer may generate alternative strategies. Because of the high selectivity of 'recent' (only retrieving one month), another strategy is to restrict the index to the last month of English newspaper articles, and then rank the articles with the content query. Assuming (realistically) that a month of newspaper articles fits into memory easily, this may be more efficiently. Notice that such optimizations based on selectivity estimates are usually made at run-time, and do not necessarily have to be defined beforehand.

Now assume a slightly more advanced scenario: we drop 'recent' from the information need, and assume automatic query expansion to be part of the IR process. It is quite likely that 'Willem-Alexander' is only referred to as 'the Dutch crown prince' in some of the English newspaper articles. Similarly, it is likely that some other articles mention 'Willem-Alexander' as well as 'the Dutch crown prince'. Thus, we hypothesize that query expansion with terms from documents containing both 'Willem-Alexander' and 'Maxima' improves recall.[1]

Generalizing this scenario, we define a strategy for queries containing several (more than one) named entities. For these queries, we first retrieve a small number of documents that contain (most of) these named entities *and* rank high using the full query. Next, we perform query expansion with blind feedback using these documents. Finally, we rank the full collection using the expanded query.

This strategy is easily expressed in queries for the Mirror DBMS. In a retrieval system based on black-boxes however, implementing a strategy like this can be rather tricky. Terms from the collection that are tagged as named entities would be stored outside the IR system, probably in a DBMS like structured data. The ranking system cannot usually be instructed to only retrieve documents that contain at least these terms. And, as argued before, the blind feedback process is often implemented on top of the core IR engine, which probably does not cache intermediate results. To process the full query, we travel between the boundaries of systems more than once, which will clearly reduce the efficiency of the system.

6 Conclusions and Future Work

We identified two types of challenges for IR systems, that are difficult to address with the current engineering practice of hard-coding the ranking process in highly optimized inverted file structures. In the current approach, a trade-off exists between flexibility and efficiency, which may be resolved by adopting a 'database approach' to IR. Flexibility is obtained by declarative specification of the retrieval model, and efficiency is addressed through

[1] Disclaimer: this example has been constructed for its educational value; it is not at all clear whether this strategy would indeed improve effectiveness in a real task.

algebraic optimization in the mapping process from specification to query plan. The Mirror DBMS is a prototype system following this approach, and it has already been used in several IR evaluations.

Based on preliminary experiments, we conjecture that our architecture extends to true multimedia retrieval instead of just text retrieval [vDdV00], and is equally suited for efficient query processing on *semi*-structured documents [SKWW00]; the latter seems particularly useful for handling tagged named entities within the documents.

In our future work, we will bring these approaches together under the Mirror umbrella. The second step is to identify optimization strategies in an interactive environment, allowing search with precise queries, browsing based on online clustering of search results, and query refinement using relevance feedback. Also, techniques based on parallel database theory are being developed for handling very large collections. The final goal is to use our previous experience with data-mining in Monet, to learn from the user interaction without explicit relevance feedback. This final step would really take full advantage of the new opportunities created by the integration of databases and information retrieval.

Acknowledgements

Annita Wilschut inspired this research with her work on Moa and GIS. Djoerd Hiemstra has been very supportive and helpful with IR experiments. Also, Henk Ernst Blok has convinced me with his experiments that our effort will pay off *some* day.

References

[BMK99] P.A. Boncz, S. Manegold, and M.L. Kersten. Database architecture optimized for the new bottleneck: Memory access. In *Proceedings of 25th International Conference on Very Large Databases (VLDB '99)*, Edinburgh, Scotland, UK, September 1999. To appear.

[Bro95] E.W. Brown. *Execution performance issues in full-text information retrieval.* PhD thesis, University of Massachusetts, Amherst, October 1995. Also appears as technical report 95-81.

[BWK98] P.A. Boncz, A.N. Wilschut, and M.L. Kersten. Flattening an object algebra to provide performance. In *Fourteenth International Conference on Data Engineering*, pages 568–577, Orlando, Florida, February 1998.

[BYRN99] R. Baeza-Yates and B. Ribeiro-Neto. *Modern information retrieval.* Addison-Wesley, Wokingham, UK, 1999.

[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. In *Proceedings of the Seventh Text Retrieval Conference TREC-8*, Gaithersburg, Maryland, November 1999.

[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.

[JFS98] B.Th. Jónsson, M.J. Franklin, and D. Srivastava. Interaction of query evaluation and buffer management for information retrieval. In Laura M. Haas and Ashutosh Tiwary, editors, *SIGMOD 1998, Proceedings ACM SIGMOD International Conference on Management of Data, June 2-4, 1998, Seattle, Washington, USA*, pages 118–129. ACM Press, 1998.

[SKWW00] A.R. Schmidt, M.L. Kersten, M.A. Windhouwer, and F. Waas. Efficient Relational Storage and Retrieval of XML Documents. In *International Workshop on the Web and Databases (In conjunction with ACM SIGMOD)*, pages 47–52, Dallas, TX, USA, May 2000.

[vDdV00] M.G.L.M. van Doorn and A.P. de Vries. The psychology of multimedia databases. In *Proceedings of the 5th ACM Digital Libraries Conference (DL'00)*, pages 1–9, San Antonio, Texas, USA, June 2000.

[WMB94] I.H. Witten, A. Moffat, and T.C. Bell. *Managing Gigabytes: compressing and indexing documents and images.* Van Nostrand Reinhold, New York, 1994.

[ZMR98] J. Zobel, A. Moffat, and K. Ramamohanarao. Inverted files versus signature files for text indexing. *ACM Transactions On Database Systems*, 23(4):453–490, December 1998.