Database Research at CITRI

Ron Sacks-Davis and Kotagiri Ramamohanarao
Collaborative Information Technology Research Institute
The University of Melbourne and RMIT

Introduction

The University of Melbourne and RMIT (Royal Melbourne Institute of Technology) are two of the leading universities in Australia and are located close together in Melbourne, the capital city of the State of Victoria. CITRI (Collaborative Information technology Research Institute) was established by these two Universities in 1990 with the support of funding from the Victorian Government. The goal of CITRI was to create a research institute of significant size (over 200 researchers and postgraduate students) to work with Australian industry in the areas of computing and communications technologies. A modern research facility was established between The University of Melbourne and RMIT to permit co-location of researchers from the two institutions.

This report describes the activities of the database research group at CITRI. This group consists of approximately 40 researchers and postgraduate students, under the direction of Professors Ron Sacks-Davis and Kotagiri Ramamohanarao. The main areas of research are hypermedia systems, document databases, and deductive databases. A partial list of recent publications is given at the end of this paper.

Hypermedia systems research

The research of the Hypermedia group is focused on storage, management, and retrieval for large document databases. The volume of text available online is growing exponentially, with the growth in use of word processors and imaging systems; research in this area is required to meet the demands of this growth. Systems for management of large volumes of text and documents will have wide application for libraries, catalogues, collections of law, documentation, and any large organisation.

The group's research can be divided into three major areas: large hypertext systems (or hyperbases), information retrieval, and database systems.

Hypertext has become widely accepted as a methodology for describing and browsing documents. In hypertext, documents are stored as collections of nodes, with links between nodes to represent relationships such as sharing of common topics, or citation of one document by another. In this paradigm, discovery of knowledge is made by browsing the data. However, current hypertext systems are designed for small volumes of text that have been coded into hypertext by hand, and are not suitable for large document collections. In particular, they do not include methodologies for selecting documents from the collection, nor is it feasible to hand code large volumes of text into hypertext.

The group is developing techniques to make large hyperbases practical. These techniques include efficient storage, retrieval, and indexing methods for large databases, automatic link generation, and the use of document structure—such as that given by markup languages like SGML—to aid query processing. The group is also extending information retrieval techniques in several ways. These include investigating algorithms to allow ranking over very large collections in reasonable lengths of time, and applying natural language and artificial intelligence techniques to aid document understanding.

To support large hyperbases efficiently, database systems should support complex objects such as documents. We have undertaken research into nested relational database systems in order to demonstrate the suitability of these systems for the support of complex applications involving text, images, and other multimedia data. Further research is being carried out to make these systems extensible so that they can incorporate new algorithms easily and to enable application software to be migrated into the database kernel over time. The Atlas database system is a research prototype that has been developed to incorporate these features.

A commercial system that incorporates some of our research is the Titan Information Management System, owned by The University of Melbourne and RMIT and supported and marketed by Knowledge Engineering. Titan has been installed in over 150 sites in Australia, New Zealand, Japan, Hong Kong and Canada.

Research Projects

Atlas: A Database System for Multimedia Applications

Principal Investigators: Prof. Ron Sacks-Davis, Dr Alan Kent, Dr Justin Zobel.

The general-purpose database systems in use today are well-suited to management of data that has a simple model and repetitive form. They are often not appropriate, however, to advanced applications that require management of more complex data, such as hyperbase systems and they do not provide operators for the direct support of text, image and other multimedia data.

The Atlas database system [12] was developed to overcome these limitations. Atlas provides both database functionality and high level support for applications involving large volumes of text. Using a data model based on nested relations, Atlas is capable of directly supporting objects with complex structures such as documents or office memos [3, 15, 18].

The design of Atlas is based on the client-server model. Applications access Atlas databases using a language called TQL [16], a nested relational extension of SQL. TQL is rich in text searching constructs and supports ranked information retrieval, whereby matching database records can be ordered according to their relevance to the query. Atlas includes powerful indexing methods based on signature files that provide efficient access to very large databases.

Several applications built using Atlas include hypertext systems, legal information systems and image databases.

Large Hypertext Databases

Principal Investigators: Mr Michael Fuller, Dr Ross Wilkinson, Dr Justin Zobel, Prof. Ron Sacks-Davis, Mr Eric Mackie (University of Waterloo).

The aim of this project is to explore the design and architecture of large hyperbases so that browsing-based systems can be combined with information retrieval techniques. Such combined systems will allow browsing users to locate particular areas of interest quickly, and give querying users the opportunity to browse interesting areas of data.

This project has resulted in a hypertext system, based on Atlas, that uses SGML to standardise and automate data capture, markup, storage, display, and exchange [2]. The project involves collaboration with the University of Waterloo, who developed a user interface with Lector, an SGML-based display tool. In this system the text and links of the database are represented visually on the screen in a style reminiscent of the printed page. Links from phrases in the text to related material are visually emphasised. Activation of a link using a mouse or the keyboard takes the user to the destination of the corresponding link.

The database can be searched without the need to learn a query language. The user simply describes the topic of her search, and a natural language algorithm translates the user's description into a ranked query for the node layer to perform. A list of links to the matching documents is then constructed in order of their relevance to the search topic [14].

Indexing and Storage Techniques

Principal Investigators: Prof. Ron Sacks-Davis, Prof. Kotagiri Ramamohanarao, Dr Justin Zobel, Dr Alistair Moffat, Dr Alan Kent, Mr Evan Harris.

A number of techniques have been developed for efficient storage and retrieval of information from large databases containing both text and structured data. Indexing methods developed within the group include signature file methods, compressed inverted files, and multikey attribute hashing.

A generalized signature file architecture has been developed to support access to both small and very large text databases [6, 8, 13]. These methods incorporate storage representation based on bit slice techniques, multi-level indexing methods where signatures are formed for both blocks of records as well as single records. These methods do not require a vocabulary of indexed terms to be maintained and provide support for indexing on word parts and word pairs.

Inverted file methods maintain a vocabulary of indexed terms and for each indexed term, an inverted list containing the record identifiers of matching records is maintained. Our compression techniques reduce the storage overheads on the inverted files and provide efficient retrieval through fast decompression techniques. These methods are more storage efficient than signature file methods and provide for the ranking of matching records in order of user relevance as well as for boolean query processing [9, 11, 17].

We are developing multikey attribute hashing techniques. These techniques are extensions of dynamic hashing techniques for primary key retrieval and are designed to support partial match retrieval. They are applicable to formatted data, as typically found in relational databases and provide for automatic clustering of records. The clustering of records can be exploited to realize efficient support for join queries and range queries [4].

We are also developing indexing techniques to support retrieval of hierarchically structured data, as typically required for retrieval from document databases.

Tethys: A Language Designed for Implementation of an Extensible Database System

Principal Investigators: Dr Alan Kent, Prof. Ron Sacks-Davis, Dr Justin Zobel.

Tethys is an object oriented language designed for the implementation of extensible database systems. Tethys is based on the Eiffel programming language and provides support for data types required in database systems. For example data of type integer that can also take NULL values is supported [5, 7].

The object oriented features of Tethys make it suitable for the development of modular and extensible database systems. A design goal of Tethys is to support both compiled and interpreted code so that application code can easily be migrated into the database kernel. The system also provides automatic storage allocation and garbage collection in order to reduce coding errors.

The Tethys compiler generates C code.

Compression for Text and Image Databases

Principal Investigators: Dr Alistair Moffat, Dr Justin Zobel, Mr Neil Sharman.

Current compression techniques are designed to compress a stream of data, and are optimised for minimal compression plus decompression time. These techniques are not well-suited to databases, in which the data arrives piecemeal, can be retrieved in any order, and compression time is relatively unimportant. We are investigating techniques suitable for document databases and have shown that the text stored in a document database can be reduced to about 30retrieval faster, because of the reduction in volume of disc traffic [1, 10].

We have also developed techniques for the progressive compression of image data, the compression of images consisting of black and white text as well as a number of other image formats.

References

[1] T.C. Bell, A. Moffat, C.G. Nevill, I.H. Witten, and J. Zobel. Data compression in full-text retrieval systems. *Journ. American Society for Inf. Sci.* To appear.

[2] A. Botham, M. Fuller, E. Mackie, R. Sacks-Davis, and R. Wilkinson. An SGML-based hypertext information retrieval system. In *Proc. 15th Aust. Comp. Sci. Conf.*, pages 111–124, Hobart, Australia, 1992.

[3] E. Harris, A. Kent, and R. Sacks-Davis. An environment for building graphical user interfaces for nested relational databases. *Information Systems*, 17(1):107–115, 1992.

[4] E. P. Harris and K. Ramamohanarao. Optimal storage management of relations for join operations. Technical Report 92/2, Dept. of Comp. Sci., Univ. of Melb., Parkville, Victoria 3052, Australia, 1992.

[5] A. Kent. The tethys programming language. Technical Report TR-91-1, Collaborative Information Technology Research Institute, Melbourne, Australia, 1991.

[6] A. Kent, K. Ramamohanarao, and R. Sacks-Davis. A superimposed coding scheme based on multiple block descriptor files for indexing very large data bases. In *Proc. 14th VLDB*, pages 351–359, Los Angeles, California, 1988.

[7] A. Kent, R. Sacks-Davis, and J. Zobel. Database systems for advanced applications. In *Proc. 15th Aust. Comp. Sci. Conf.*, Hobart, Australia, 1992.

[8] A.J. Kent, R. Sacks-Davis, and K. Ramamohanarao. A signature file scheme based on multiple organisations for indexing very large text databases. *Journ. American Society for Inf. Sci.*, 41(7):508–534, 1990.

[9] A. Moffat. Economical inversion of large text files. *Computing Systems*, 5(2), 1992.

[10] A. Moffat and J. Zobel. Coding for compression in full-text retrieval systems. In *Proc. IEEE Data Compression Conference*, pages 72–81, Snowbird, Utah, 1992.

[11] A. Moffat and J. Zobel. Parameterised compression for sparse bitmaps. In N. Belkin, P. Ingwersen, and A.M. Pejtersen, editors, *Proc. 15th SIGIR*, pages 274–285, Copenhagen, Denmark, 1992.

[12] R. Sacks-Davis, A. Kent, K. Ramamohanarao, J. Thom, and J. Zobel. Atlas: A nested relational database system for text applications. Technical Report TR-92-52, Collaborative Information Technology Research Institute, Melbourne, Australia, 1992.

[13] R. Sacks-Davis, K. Ramamohanarao, and A.J. Kent. Multi-key access methods based on superimposed coding techniques. *ACM Trans. on Database Sys.*, 12(4):655–696, 1987.

[14] R. Sacks-Davis, P. Wallis, and R. Wilkinson. Using syntactic analysis in a document retrieval system that uses signature files. In *Proc. 13th Int. Conf. on Information Retrieval*, pages 179–192, September 1990.

[15] R. Sacks-Davis, W. Wen, A. Kent, and K. Ramamohanarao. Complex object support for a document database system. In *Proc. 13th Aust. Comp. Sci. Conf.*, pages 322–333, 1990.

[16] J.A. Thom, A.J. Kent, and R. Sacks-Davis. TQL: A nested relational query language. *Aust. Computer Journal*, 23(2):53–65, May 1991.

[17] J. Zobel, A. Moffat, and R. Sacks-Davis. An efficient indexing technique for full-text database systems. In *Proc. 18th VLDB*, Vancouver, Canada, 1992.

[18] J. Zobel, J.A. Thom, and R. Sacks-Davis. Efficiency of nested relational document database systems. In *Proc. 17th VLDB*, pages 91–102, Barcelona, Spain, 1991.

Deductive Databases Research

The past three decades have seen an extraordinary explosion in the amount of data stored in computer systems. There is now so much data stored in databases that we are confronted with a new problem: how to make effective use of the vast but often bewildering storehouses of information that previous research has placed at our fingertips.

Several emerging technologies are attacking parts of this problem. Hypertext systems and object-oriented databases tackle the problem of navigation: how users and programs can find the information they are interested in by following links through other pieces of information. Deductive databases, logic programming and constraint logic programming have a different emphasis: they attack our present inability to reason efficiently with large amounts of data and to reason efficiently over complex domains. Their aim is to provide, to human beings working with very large databases (whether as users or as programmers), tools that assist them in extracting all relevant information from the database, analysing this information in various complex ways, and drawing conclusions accordingly.

There are many applications of such technologies. Examples include decision support systems, expert database systems, medical database systems, legal database systems, and planning systems.

The group has also produced several software systems. The NU-Prolog system is installed in many sites around the world. The Aditi deductive database system is a database system that supports recursive rules and negation.

Research Projects

Aditi Deductive Database System

Principal Investigators: Prof. Ramamohanarao Kotagiri, Dr Guozhu Dong, Dr John Shepherd, Dr James Harland, Dr Zoltan Somogyi, Dr Peter Stuckey, Dr David Kemp, Mr Tim Leask, Mr Jayen Vaghani.

Existing database technologies, such as relational systems, are computationally weak, as complex applications cannot be completely implemented in such systems. In such cases, the database language is embedded in a host language (for example C or Cobol) which gives the required computational power. However, this approach is very inefficient, as the interface between the host language and the database language forms a bottleneck. Automatic optimization of such systems is not feasible. On the other hand, logic programming provides a uniform formalism for the expression of all data, programs, queries and integrity constraints in the system, and the semantics of all these constructs. Deductive database systems are based on logic, and are computationally as powerful as the host languages, and therefore these problems do not arise. In addition, these systems are amenable to automatic optimization.

The main emphasis in deductive databases is handling large amounts of data efficiently. Deductive databases represent an integration of logic programming and relational database technologies with some new techniques unique to the field. Relational database technology provides the implementation techniques: relational algebra operations, transaction mechanisms and client-server architecture. The main technology unique to deductive databases is query evaluation algorithms for recursive rules.

Our research is concerned with efficient implementation of deductive database systems.

Our group has developed a prototype deductive database system called Aditi (Aditi is the name of an ancient Indian goddess) [12]. The front end of Aditi interacts with the user in a pure variant of Prolog, a language that has more expressive power than relational query languages. The back end has a multiprocess client-server architecture, evaluates queries using relational algebra operations, and uses other techniques developed for relational database systems for the efficient management of disk based data. Programs and queries are translated from logic to the level of relational algebra operations by a compiler written in NU-Prolog. The bottom-up query evaluation algorithms and some of the optimization algorithms built into this compiler were especially developed for deductive database systems.

The main areas of research focus in deductive databases are as follows:

- Implementation issues, including system architecture (for example, client-server models); indexing schemes; algorithms for relational algebra operations (for example, join algorithms); and evaluation methods, such as hybrid top-down and bottom-up computations.
- Updates, including language primitives for transactions; integration of updates and transactions into the framework of deductive database systems; semantics of updates; and crash recovery issues.
- Optimization, including algorithms for efficient recursive query evaluation [6, 4, 2]; constraint propagation [7, 5]; abstract interpretation techniques to aid constraint propagation; and integration of known optimization techniques for relational databases.
- Parallelism [11], including evaluation of various models (e.g. client-server, dataflow) for the exploitation of large grain parallelism; exploitation of small grain parallelism through relational operations; program decomposition analysis to enhance parallelism; and efficient data cache design to aid parallel computation.
- Semantic issues, including efficient computational schemes for logic programs containing negation and aggregate operators [9, 8, 10, 3]; and concurrency protocols for deductive database systems.

Deductive databases have so far supported only stratified forms of negation and aggregation. We believe we have identified a promising approach to non-stratified negation and aggregation. We are exploring both the theoretical implications of our algorithm and the effectiveness of its implementation in practice.

Aditi employs a bottom-up, set-at-a-time evaluation strategy. Bottom-up strategies, which are based on relational technology, have very good average performance in a database environment, but there are some things they are not good at, such as list processing. To alleviate these problems, we are looking at ways to either incorporate other strategies (e.g. a top-down, set-at-a-time strategy) into Aditi or to interface Aditi to other systems with evaluation mechanisms [1]. In particular, we are investigating connections between Aditi and conventional and constraint logic programming languages.

The work on parallelism assumes the availability of shared memory. In the longer term, we wish to look at what it would take to implement a deductive database system such as Aditi on a shared-nothing multiprocessor. We would also like to explore the implications of large, non-volatile main memories.

References

[1] A. Davison. A survey of LP-based object oriented languages. In P. Wegner, A. Yonezawa, and G. Agha, editors, *Object-Based Concurrency*. Addison Wesley, 1992.

[2] G. Dong. On Datalog linearization of chain queries. In J.D. Ullman, editor, *Theoretical Studies in Comp. Sci.*, pages 181–206. Academic Press, 1991.

[3] G. Dong. On the monotonicity of (LDL) logic programs with set. In *Proc. 3rd Int. Symp. on Math. Fundamentals of Database and Knowledge Base Systems*, pages 201–215, 1991. LNCS 495.

[4] G. Dong. Datalog expressiveness of chain queries. In *Proc. ACM Symp. on Principles of Database Systems*, 1992.

[5] G. Dong and S. Ginsburg. Localizable constraints for object histories. *Theoretical Comp. Sci.*, 90:391–432, 1991.

[6] G. Dong and R. Topor. Incremental evaluation of Datalog queries. In *Proc. Int. Conf. on Database Theory*, Berlin, 1992.

[7] J. Harland and K. Ramamohanarao. Constraints for query optimisations in deductive databases. In *Proc. 2nd Far East*

Workshop on Future Database Systems, Kyoto, Japan, 1992.

[8] D. Kemp, D. Srivastava, and P. Stuckey. Magic sets and bottom-up evaluation of well-founded models. In *Proc. Int. Logic Programming Symp.*, pages 337–351, San Diego, 1991.

[9] D. Kemp, D. Srivastava, and P. Stuckey. Query restricted bottom-up evaluation of normal programs. In *Proc. Joint Int. Conf./Symp. on Logic Programming*, Washington DC, 1992.

[10] D. Kemp and P. Stuckey. Semantics of logic programs with aggregates. In *Proc. Int. Logic Programming Symp.*, pages 387–404, San Diego, 1991.

[11] T. Leask, K. Ramamohanarao, and P. Stuckey. Exploiting parallelism in bottom-up computation in Aditi. In *Proc. ILPS Workshop on Deductive Databases*, pages 72–81, San Diego, 1991.

[12] J. Vaghani,K. Ramamohanarao, D. Kemp, Z. Somogyi, and P. Stuckey. An introduction to the Aditi deductive database system. *Australian Computer Journal*, 23(2):37–52, 1991. Also see 'Design overview of the Aditi deductive database system', in Proc. 7th Int. Conf. on Data Eng., 240-247, Kobe, 1991.

Acknowledgements

Research in the database groups is supported from a number of sources. These include the Key Centre for Knowledge Based Systems, the Australian Research Council, the Cooperative Research Centre for Intelligent Decision Systems and Kodak Australia. The total funding for the database research effort at CITRI is approximately 2 million dollars per year.