1

Introduction Parallel Database and Knowledge-Base Systems

Mahdi Abdelguerfi[†] and Simon Lavington[††]

[†]Computer Science Department, University of New Orleans, USA
[††]Computer Science Department, University of Essex, UK

1.1 INTRODUCTION AND SCOPE

Database technology has been extended into many new applications areas such as computer-aided design and manufacture, geographical information systems, and real-time intelligent decision making. Such applications bring with them demands for richer information models and the capability to handle larger quantities of heterogeneous data. Furthermore, increasing numbers of online users are expecting some form of smartness from their information systems. We use the term *smart* information systems, or knowledge-base systems, to describe computer applications that incorporate nontrivial features such as the ability to adapt and evolve, or to carry out inferencing. All these factors combine to produce a need for improved performance and additional functionality.

This book consists of 13 invited papers, chosen to illustrate ways in which novel parallel hardware is being used to provide improved performance and additional functionality for a variety of information systems. Two of the machines described have each sold to well over one hundred customers (see Chapters 8 and 9). Two more are in the final stages of product launch (see Chapters 8 and 11). The rest of the machines described in this book are research prototypes. Overall, they provide a survey of the latest trends in performance-enhancing architectures for smart information systems. The book will thus appeal to all engaged in the design or use of high-performance (parallel) architectures for nonnumeric applications.

The machines featured in this book have been designed to support various information systems, ranging from relational databases to semantic networks and other artificial-intelligence paradigms. Knowledge-representation formalisms, as such, are not described in any detail. Although expertise in knowledge representations is not required for an understanding of the hardware architectures in this book, readers wishing for background information on this subject may refer to Frost.[1]

The topic of database and knowledge-base machines is a large one, and no single volume can hope to cover the field in a truly comprehensive manner. Particularly, this book does not touch on distributed or federated databases,[2] heterogeneous databases,[3] or related issues. However, many of the projects we have chosen to emphasize contain generic architectural ideas that may be used to support several sets of higher-level requirements. In other words, some of the machines are based on hardware designs that strive to be semantics-free. For completeness, we list the major review books, conference proceedings, and survey papers that have appeared within the last ten years on the hardware and software of database and knowledge-base machines.[4-14] Papers describing ongoing research appear from time to time in several journals, particularly in the *IEEE Transactions on Knowledge and Data Engineering*, in the *Proceedings of the ACM SIGMOD*, and in the *Proceedings of the International Conferences on Very Large Databases.*

1.2 THE TECHNICAL CONTEXT

Many mainframe database management systems already have difficulty meeting the I/O and CPU performance requirements of information systems that support large numbers of concurrent users, and/or handle massive amounts of data.[15] To achieve the required performance levels, several mainframe designers have incorporated add-on special-purpose accelerators. ICL's Content-Addressable File Store is an early example of an add-on hardware accelerator, the latest version of which is now incorporated into every Series 3 ICL mainframe.[16] CAFS is designed to alleviate the disk I/O bottleneck and reduce CPU loading. Rinda was developed at NTT to accelerate the execution of relational database queries, and is now connected to NTT's general-purpose DIPS series mainframes.[17] Hitachi's largest mainframes can now be purchased with an optional hardware accelerator (the integrated database processor) that performs sorting and merging.[18] Greo is a very-large-scale-integration sorter and merger used as a performance accelerator for Mitsubishi machines.[19] On a more specialized topic, there are add-on performance accelerators that undertake particular tasks such as free-text searching. An example is the High Speed Text System designed to be connected to a PDP/11.[20]

The hardware parallelism, where it exists in the above add-on accelerators, is quite localized. For example, it does not usually extend to whole-structure operations such as those implied by dyadic relational algebraic operators—a theme we return to later. An emerging trend in database technology is to employ parallelism more explicitly.[15] For conventional database management systems that use industry-standard database models, such as the relational, the approach to parallelism can take one of two forms.

Firstly, massively parallel general-purpose computers such as Intel's iPSC, nCube, and the Distributed Array Processor marketed by Cambridge Parallel Processing, have all become platforms for DBMS and related commercial software.[21-23] DBMS vendors such as Oracle have been active in promoting these developments. Some of the newer players in the high-performance parallel computer market, like Kendall Square Research Corp., are taking business and commercial applications seriously. Thus, high-performance platforms, which have hitherto been targeted at numeric applications, are also being used in the non-numeric, that is, symbolic, area. On a somewhat related theme, parallel processing has been shown to give spectacular commercial advantages in the field of financial modeling and securities trading.[24] These developments are bound to make end-users of conventional data processing equipment restless, and in the long run encourage fresh architectural initiatives on the part of computer manufacturers.

In the second approach to parallelism in DBMS, some of these initiatives are already apparent. This approach is based on the use of arrays of off-the-shelf components, such as microprocessors and (cheap) disks, to form parallel add-on database machines and performance accelerators. For the most part, these hardware systems are based on multiple instruction, multiple data, shared-nothing, parallel architectures.[25] Besides providing high performance, such systems are often modular and therefore easily scalable. The current market leader is Teradata's Database Computer DBC/1012,[26] which makes use of Intel microprocessors and a special interconnecting network known as the Ynet. Teradata has recently become part of NCR, which has announced the NCR 3700 as a successor to the DBC/1012.[27] This system uses a high-performance multistage interconnection network, known as BYNet, and redundant arrays of inexpensive disks (RAIDs).[28] A new United Kingdom company, White Cross Systems, has just announced a database machine using arrays of transputers.

Unlike conventional mainframes, the above parallel systems can usually be scaled up to meet the future performance needs of conventional database software. Furthermore, parallel computing platforms such as nCube are reported to be more cost-effective than mainframe solutions, by factors of 20 or so.[21] Of course, a company that adopts a massively parallel solution may have considerable difficulty running its existing programs and ready-made software packages. Nevertheless, the performance and cost benefits of parallel solutions are compelling. In addition, the availability of high-performance platforms will encourage designers of information systems to experiment with advanced features, for example, deductive, object-oriented, temporal reasoning, and fuzzy logic, which may have previously introduced unacceptable overheads on conventional uniprocessor platforms. However, adding such advanced features does itself bring additional challenges for the systems architect—as is discussed shortly.

Apart from the machines mentioned previously, many more parallel database hardware systems exist in the development stages or as research prototypes. Examples are Multibackend Database Supercomputer MDBS,[29] GAMMA,[30] IDIOMS,[31] the Datacycle project,[32] and the parallel multi-wavefront work of Su.[33] MDBS is a multibackend database system having the interesting property that, despite an increase in the size of the database, a transaction's response time can be kept constant by increasing the degree of backend parallelism. GAMMA is a multiprocessor system in which a set of identical computers, each with its local main memory and secondary storage, is interconnected by a ring bus. The latest version of GAMMA is based on a 32-node Intel iPSC/2 hypercube with a disk attached to each node. The IDIOMS machine, Professor Su's prototype, and Datacycle all employ transputers. IDIOMS is of interest because it is designed to support both online transaction processing and the decision-support queries that result from management information systems that use the same OLTP data. Su's multi-wavefront parallel database system uses the object-oriented paradigm as its underlying data model. The Datacycle parallel system supports SQL plus some extra functionality such as fuzzy queries. These last three systems are examples of projects that go beyond classical (relational) DBMS, in response to user's needs for smarter information systems.

The functional requirements of smart information systems involve the efficient implementation of tasks such as rapid pattern-directed searching and whole-structure operations such as join and transitive closure—tasks that are believed to be generic to knowledge-base systems. There is much natural parallelism to be exploited in these tasks. It is felt by several researchers that such tasks highlight the mismatch between the functional requirements of knowledge-base systems and the operational characteristics of both conventional von Neumann uniprocessors and many existing parallel architectures. Particular examples are the

mismatch (1) between logical data-objects of varying size and granularity and blocks of linearly addressed memory and (2) between the rich variety of data types used by knowledge-base software and the limited scalar primitives supported by conventional arithmetic logic units. These mismatches result in overcomplex software and unacceptably long runtimes. How to overcome these mismatches is a topic of current research; the IFS/2 project (see below) represents one possible approach.

Adding more functionality to produce "smart" systems covers a range of software paradigms. Many of these draw on the insights of artificial intelligence. It is convenient to divide the resulting hardware architectures into two categories: those that take a broad approach to knowledge representation, and those that focus on one representation or problem-solving paradigm. The first category, which may be called knowledge-base machines, often remains compatible with the *n*-ary relational model but extends this model toward deductive databases and the support of object-based and rule-based systems. Support may also be given to some AI-like knowledge representations such as semantic networks, while still retaining an ability to deal with commercially realistic quantities of data. An example of this first category of machines is the European Declarative System, a collaboration among ICL, Siemens, and Bull.[34] Some of the EDS insights originated in a research project to design a parallel graph reduction machine for functional languages. However, the production EDS takes relational databases as its starting point. EDS supports Extended SQL,[35] as well as LISP and PROLOG. Research into knowledge-base machine architectures includes the recent work of Stolfo[36] and the IFS/2.[37] Professor Stolfo has examined rule-based systems such as Datalog and OPS5, with a view to extracting the common inherent parallelism. The IFS/2 uses a modularly extensible array of single instruction, multiple data search engines under transputer control, to achieve associative processing of structures such as relations, sets, and graphs. At the AI end of the smart information system spectrum come machines whose architecture is focused on one particular knowledge representation or one (declarative) language. There exists a long tradition of machines dedicated to production-rule systems such as DADO.[38]

More recent examples of machines designed to support semantic networks are SNAP[39] and IXM/2.[40] Computers designed to speed up execution of declarative languages (that is, logic and functional) are well reviewed in Kogge.[9] An interesting example of a PROLOG engine is the Knowledge Crunching Machine, which comes from the European Computer Manufacturers' Research Centre in Munich.[41] Unfortunately, not many of these architectures perform well when handling commercially realistic volumes of data. (By "commercially realistic" we mean at least several tens of megabytes for the knowledge-base systems and perhaps several tens of gigabytes for the more conventional smart DBMS applications). On the very-large-scale-integration front, there have been several associative (that is, content-addressable) memory chips that contain added facilities for declarative languages.[42] There are also several examples of VLSI accelerators for the more general forms of pattern-directed search, as found in Stormon.[43]

1.3 GUIDE TO THE REST OF THE BOOK

The main text is divided into four sections, which broadly follow the developments described above.

(1) *Database machines:* In this category come add-on machines and performance-enhancing units that employ parallel hardware mainly in support of conventional database models such as the relational. Object-oriented systems are also beginning

to emerge. However, SQL typically forms the programmer's interface to the majority of machines featured in this section.

(2) *Using massively parallel general computing platforms for DBMS:* Two examples are given to illustrate how high-performance computers can be used to support database and related software, even though some of these platforms (in this case a single-instruction, multiple-data array processor) may originally have been designed with scientific/numeric applications in mind. Once again, the emphasis is on information systems that follow conventional models or service well-known applications.

(3) *Knowledge-base machines:* This section features projects that go beyond the relational model toward what has been called "smart" information systems. The ability to handle realistic quantities of data is retained, but this data may have a complex structure that includes rules. These machines thus attempt to combine some of the management features of DBMS with techniques drawn from artificial intelligence. The programmer's interface may be extended SQL (that is, with deductive features) or a declarative language.

(4) *Artificial-intelligence machines:* Two examples are given of machines that are deliberately designed to speed up a particular knowledge-representation formalism or a particular problem-solving paradigm. The projects in this section can be regarded more as research prototypes than products; nevertheless, experience gained with such machines will no doubt influence the development of future architectural features relevant to smart information systems.

Mahdi Abdelguerfi, University of New Orleans, USA
Simon Lavington, University of Essex, UK
November 1994

REFERENCES

1. R.A. Frost, *Introduction to Knowledge-Based Systems*, Collins, London, England, 1986.
2. S. Ceri and G. Pelagatti, *Distributed Databases: Principles and Systems*, McGraw-Hill, New York, N.Y., 1985.
3. D.K. Hsiao and M.N. Kamel, "Heterogeneous Databases: Proliferation, Issues, and Solutions," *IEEE Trans. Data and Knowledge Eng.*, Vol. 4, No. 3, June 1992, pp. 45–62.
4. D.K. Hsiao, *Advanced Database Machine Architectures*, Prentice-Hall, Englewood Cliffs, N.J., 1983.
5. S.Y.W. Su, *Database Computers: Principles, Architectures, and Techniques*, McGraw-Hill, New York, N.Y., 1988.
6. J.S. Kowalik, *Parallel Computation and Computers for Artificial Intelligence*, Kluwer Academic, Norwell, Mass., 1988.
7. M. Kitsuregawa and H. Tanaka, eds., "Database Machines and Knowledge-Base Machines," *Proc. 5th Int'l Workshop Database Machines*, Kluwer Academic, Norwell, Mass., 1988.
8. H. Boral and P. Faudemay, eds., *Proc. 6th Int'l Workshop Database Machines*, Springer-Verlag, New York, N.Y., 1989.
9. P.M.Kogge, *The Architecture of Symbolic Computers*, McGraw-Hill, New York, N.Y. 1991.
10. P. America, *Parallel Database Systems*, Springer-Verlag, New York, N.Y., 1991.
11. P. Valduriez, *Parallel Processing and Data Management*, Chapman and Hall, London, England, 1992.
12. A.R. Hurson, et al., *IEEE Tutorial: Parallel Architectures for Database Systems*, IEEE CS Press, Los Alamitos, Calif., 1989.
13. Hurson, A.R., et al., "Parallel Architectures for Database Systems," *Advances in Computers*, Vol. 30, 1989, pp. 107–151.

14. "Database Prototype Systems," *IEEE Trans. Knowledge and Data Eng.*, Special Issue, Vol. 2, No. 1, Mar. 1990.
15. D. DeWitt and J. Gray, "Parallel Database Systems: The Future of High-Performance Database Systems" *Comm. ACM*, Vol. 35, No. 6, 1992, pp. 85-98.
16. V.A.J. Maller, "Information Retrieval Using the Content Addressable File Store," *Proc. IFIP-80 Congress*, North Holland, Amsterdam, The Netherlands, 1980, pp. 187-190.
17. T. Satoh and U. Inoue, "Rinda: A Relational Database Processor for Large Databases," Chapter 4 in this book.
18. S. Kojima, et al., "IDP—A Main Storage Based Vector Database Processor," *Proc. 5th Int'l Workshop Database Machines*, Kluwer Academic, Norwell, Mass., 1988, pp. 47-60.
19. M. Kitsuregawa and W. Yang, "Evaluation of 18-Stage Pipeline Hardware Sorter," *Proc. 6th Int'l Workshop Database Machines*, Springer-Verlag, New York, N.Y., 1989, pp. 142-145.
20. *HSTS: High Speed Text Search System Product Literature*, Operating Systems Inc., 1982.
21. M. Stroud, "Massively Parallel Computer Makers Begin to Target Database Applications," *Investors Daily*, June 1992.
22. J. Spiers, "Mapping a Fourth Generation DBMS to a Range of Parallel Machines," in *Parallel Processing and Data Management*, P. Valduriez, ed., Chapman and Hall, London, England, 1992, pp. 53-67.
23. N. Bond and S. Reddaway, "A Massively Parallel Indexing Engine Using DAP," Chapter 9 in this book.
24. S. Zenios, "Parallel and Supercomputing in the Practice of Management Science," *Proc. UNICOM Seminar on Commercial Parallel Processing*, Uxbridge, Middlesex, England, 1992, pp. 67-95.
25. M. Stonebraker, "The Case for Shared-Nothing," *Database Engineering*, Vol. 9, No.1, 1986, pp. 4-9.
26. J. Page, "High-Performance Database for Client/Server Systems," in *Parallel Processing and Data Management*, P. Valduriez, ed., Chapman and Hall, London, England, 1992, pp. 33-51.
27. F. Cariño, et al., "Industrial Database Supercomputer Exegis—the DBC/1012, the NCR 3700, the YNet and BYNet," Chapter 8 in this book.
28. D.A. Patterson, et al., "The Case for Redundant Arrays of Inexpensive Disks (RAID)," Proc. *ACM SIGMOD*, ACM Press New York, N. Y., 1988, pp. 109-116.
29. D.K. Hsiao, "From DBC to MDBS—A Progression in Database Machine Research," Chapter 3 in this book.
30. D. DeWitt, et al., "The GAMMA Database Machine Project," *IEEE Trans. Knowledge and Data Engineering*, Vol. 2, No.1, Mar. 1990, pp. 44-62.
31. J. Kerridge, "IDIOMS: A Multitransputer Database Machine," Chapter 2 in this book.
32. T.F. Bowen, et al., "The Datacycle Architecture: Applying VLSI Filters to Large Databases," Chapter 7 in this book.
33. S.Y.W. Su, et al., "Parallel Multi-Wavefront Algorithms for Pattern-Based Processing of Object-Oriented Databases," Chapter 6 in this book.
34. L. Bormann, et al., "EDS: An Advanced Parallel Database Server," Chapter 11 in this book.
35. G. Gardarin, et al., "ESQL: An Extended SQL with Object and Deductive Capabilities," *Proc. Int'l Conf. Database and Expert System Applications*, Springer-Verlag, New York, N.Y., 1990, pp. 299-307.
36. J.S. Stolfo, et al., "A Parallel and Distributed Environment for Database Rule Processing: Open Problems and Future Directions," Chapter 12 in this book.
37. S.H. Lavington, "The IFS/2: Add-On Support for Knowledge-Base Systems," Chapter 10 in this book.
38. J.S. Stolfo and D.E. Shaw, "DADO: A Tree-Structured Machine Architecture for Production Systems," *Proc. Nat'l Conf. Artificial Intelligence*, Morgan Kaufman, Pittsburgh, Penn., 1982, pp. 242-246.
39. D. Moldovan, et al., "SNAP: A Marker-Propagation Architecture for Knowledge Processing," *IEEE Trans. Parallel and Distributed Systems*, Vol. 3, No. 4, July 1992, pp. 397-410.

40. T.S. Higuchi, "IXM/2: A Parallel Associative Processor for Knowledge Processing," Chapter 13 in this book.
41. J. Noye, "An Overview of the Knowledge Crunching Machine," Chapter 14 in this book.
42. I. Robinson, "A Prolog Processor Based on a Pattern Matching Memory Device," *Proc. 3rd Int'l Conf. Logic Programming*, Springer-Verlag, New York, N.Y., 1986, pp. 172–179.
43. C.D. Stormon, et al., "An Architecture Based on Content-Addressable Memory for the Rapid Execution of Prolog," *Proc. 5th Conf. Logic Programming*, MIT Press, Cambridge, Mass., 1988, pp. 1448–1474.