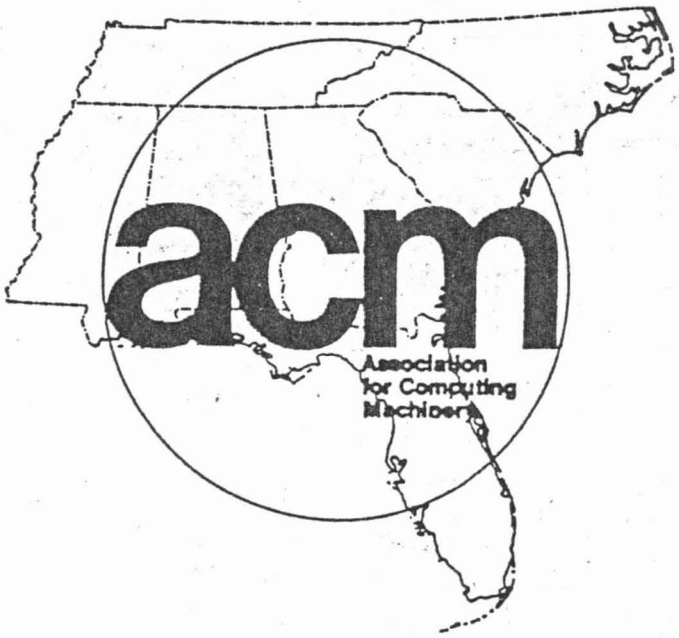
88-HT

PROCEEDINGS
26th ANNUAL
SOUTHEAST REGIONAL
CONFERENCE



James H. Cross II and Edwin Ellis, Editors

Mobile, Alabama
April 20-22, 1988

HIGH-THROUGHPUT HIGHLY-PARALLEL DATABASE SYSTEM

D.Tal, N. Rishe, D. Barton and N. Prabhakaran

School of Computer Science
Florida International University —
The State University of Florida at Miami
University Park, Miami, FL 33199

It is generally agreed that parallel database machines offer a way, perhaps the only way, to meet the ever growing demands of information processing. At the same time, it is believed by many researchers and practitioners that the currently popular Relational Database Model will soon be recognized as absolete and will be replaced by semantic data models which closely captures the information of the user's world and allows intelligent user interfaces, full data-independence and flexibility. Seeking the best of both worlds, we suggest a new architecture called the Linear-throughput Semantic Database Machine (LSDM). This architecture attempts to produce the best marriage between a highly parallel database machine and an implementation of the semantic binary database model.

1. Introduction

The traditional design of a database management system is based on a single processor, several large disks and some cache memory capable of holding a small fraction of the database. This approach entails a high overhead for maintaining data structures to minimize disk I/O and is limited by the slow sequential disk access bottleneck. As a consequence, there have been in recent years growing and widespread research into new strategies which can exploit parallelism and provide faster response to users.

Another issue which plays a decisive role in database management is the semantics of the data. The choice of the data model and the query language determine user convenience, flexibility of use of the database, and has a great impact on the efficiency of access to the database. Many researchers and practitioners believe that the currently popular Relational Database Model will soon be recognized as absolete and will be replaced by semantic data models which closely captures the information of the user's world and allows intelligent user interfaces, full data-independence and flexibility. In our database machine we use one of the semantic data models — the Semantic Binary Model [Rishe-88-DDF] which offers a very simple yet fully powerful logical data structure and allows an efficient implementation.

We propose a new database machine architecture LSDM (Linear-throughput Semantic Database Machine) which offers massive parallelism and supports the semantic binary model. Apart from the semantic advantages, the use of the semantic model in LSDM will make the machine more efficient. The architecture is designed to satisfy the following requirements:

- the computer performance should respond to the continuing increase in volume of queries and database size;
- it should be relatively easy to expand the system by adding additional processors and disks and the throughput is expected to increase linearly;
- it is necessary to have a degree of tolerance to faults in order to improve system reliability and safety;
- dynamic load balancing is desired in order to share the work among the processors and to reduce idle times.

The remaining sections of this paper are as follows. The second section describes several contemporary strategies. To see how the semantic binary model can be combined with the architecture we examine in the third section the nature of the semantic model. The fourth section introduces the new parallel architecture. An implementation is discussed in the fifth section. Finally, conclusions and further research are suggested.

2. Known Architectures

Many hardware alternatives have been suggested in order to overcome the limitations of a single processor system [Hsiao-83, Su-88]. One method is to use many general purpose computers with identical software (software multiple backend). This approach increases throughput and reduces the response time. However, this approach is extremely expensive.

Another method is to use an intelligent controller which can perform very rapid associative searches of the database. It has special hardware usually involving parallel processors to search the database, but cannot run a high level data manipulation language.

An efficient method is to use a database computer which can perform the typical relational functions — selection, projection, join, union, minus, and update functions.

Since there is a smooth continuum between database computers and intelligent controllers, they are grouped together as database machines (DBM). With declining hardware costs, many DBMs have been developed.

STARAN is an associative array processor built for radar/image processing [Ozkarahan-86]. It is an efficient system with a high degree of parallelism. However, its serial and slow paging causes a heavy overhead in I/O operations. CASSM (Context Addressed Segment Sequential Memory) overcomes the above I/O bottleneck by employing one cellular logic system per disk surface and a controller [Su-79]. It cannot be guaranteed however that each cell gets control of the bus to communicate with the controller. Also, every cell requires a storage array for its results.

DBMAC is a relational DBM that utilizes a multiprocessor architecture with parallel disk access [Missikoff-83]. It uses attribute partitioning, giving it a domain based architecture. The associative search of data is accomplished with on-the-fly (synchronous processing of data) filtering. DBMAC is efficient in handling complex retrieval queries, but not intended for update operations. IDM (Intelligent Data Machine) is a micro-computer based commercial DBM [Le-Viet-83]. Its architecture is based on DEC mini-computers. It has a software backend (synthesized hardware for the implementation) with RAM cache. The software backend does not have much impact on the response time.

MICRONET is a bus-structured multiprocessor DBM. It uses a custom-built MICRONET bus [Su-88] to connect a number of processors with the host system. The bus becomes a bottleneck when the number of processors increase since all of them have to communicate through the same bus.

A tree-structured multiprocessor DBM is used in HYPERTREE [Goodman-81]. The processors are organized as a balanced binary tree with some additional connections between nodes at the same level. Each leaf node processor is connected to a read head of a disk. The internal nodes do not have secondary storage.

A cube-connected multiprocessor DBM [Baru-87] is a relational database machine. Each processor in the cube has a separate disk. This architecture provides high parallelism and a good distribution of secondary storage to all processors.

3. The Semantic Binary Model

Since [Abrial-74], many semantic data models have been studied in the Computer Science literature. Although somewhat different in their terminology and their selection of tools used to describe the semantics of the real world, they have several common principles:

- The entities of the real world are represented in the database in a manner transparent to the user. (Unlike the relational model where entities are represented by the values of keys of some tables; and the network model where entities are represented by record occurrences.) Hereinafter, the user-transparent representations of real-world entities are referred to as "abstract objects". The "concrete objects", or "printable values", are numbers, character strings, *etc.* The concrete objects have conventional representations on paper and in the computer.
- The entities are classified into types, or categories, which need not be disjoint. Meta-relations of inclusion are defined between the categories.
- Logically-explicit relationships are specified among abstract objects (*e.g.*, "person p1 is the mother of person p2") and between abstract and concrete objects (*e.g.*, "person p1 has first name 'Jack'"). There are no direct relationships among the concrete objects. In most semantic models, only binary relations are allowed, since higher order relations do not add any power of semantic expressiveness ([Bracchi-76], [Rishe-87-RM], [Rishe-88-DDF]), but do decrease the flexibility of the database and representability of partially-unknown information, and add complexity and potential for logical redundancy ([Rishe-88-DDF]).

The advantages of the semantic models versus the relational and older models with respect to database design, database maintenance, data integrity, conciseness of languages, and ease of DML programming are known [Rishe-88-DDF]. We also believe that the semantic models can have an efficient implementation.

Typically, semantic data models are implemented as interfaces to database managements systems in other data models, e.g., the relational or the network model [UNISYS-87]. (Although, there are less typical, direct implementations, *e.g.* [Lien-81], [Chan-82], [Benneworth-81].) The efficiency of an interface implementation is limited to that of the conventional DBMS, and is normally much worse due to the interface overhead. The direct implementations are also believed to be less efficient than the conventional systems. However,

the semantic models have potential for much more efficient implementation than the conventional data models. This is due to two facts:

- All the physical aspects of representation of information by data are user-transparent in the semantic models. This creates greater potential for optimization: more things may be changed for efficiency considerations, without affecting the user programs. The relational Model has more data independence than the older models, for example the order of rows in the tables (relations) is transparent to the user. The semantic models have yet more user-transparency. For example, the representation of real-world entities by printable values is transparent to the user.
- In the semantic models, the system knows more about the meaning of the user's data and about the meaningful connections between such data. This knowledge can be utilized to organize the data so that meaningful operations can be performed faster at the expense of meaningless operations.

In LSDM we use the Semantic Binary Model (SBM) [Rishe-88-DDF], a descendant of the model proposed in [Abrial-74]. This model does not have as rich an arsenal of tools for semantic description as can be found in some other semantic models, e.g. the IFO model [Abiteboul-84], SDM [Hammer-81] (implementation [UNISYS-87]), the Functional Model [Shipman-81] (implementation [Chan-82]), SEMBASE [King-85], NIAM ([Nijssen-81], [Nijssen-82], [Leung-87]). Nevertheless, the SBM has a small set of sufficient simple tools by which all the semantic descriptors of the other models can be constructed. This makes SBM easier to use for the novice, easier to implement, and usable for delineation of the common properties of the semantic models. The results of this paper apply to most other semantic models.

The semantic binary model represents the information of an application's world as a collection of elementary facts of two types: unary facts categorizing objects of the real world and binary facts establishing relationships of various kinds between pairs of objects. The graphical database schema and the integrity constraints determine what sets of facts are meaningful, i.e. can comprise an instantaneous database (the database as may be seen at some instance of time.)

The formal semantics of the semantic binary model is defined in [Rishe-87-DS] using the methodology proposed in [Rishe-86-DN]. The syntax and informal semantics of the model and its languages (data definition, 4-th generation data manipulation, non-procedural languages for queries, updates, specification of constraints, userviews, etc.) are given in [Rishe-88-DDF]. A non-procedural semantic database language of maximal theoretically-possible expressive power is given [Rishe-86-PS]. In this language, one can specify every computable query, transaction, constraint, etc.

4. LSDM Architecture

Conceptually, the LSDM is a distributed system of many processors that control a large number of small disks drives. The processors can work concurrently on various parts of a large database. All the processors are identical and may communicate via high speed communication channels that form together a hypercube network. The hypercube network has been selected since it provides an elegant mechanism for connecting processors: inter-processor distances are small, and routing algorithms are both simple and flexible. We discuss here only those concepts of the hypercube that are relevant to the development of LSDM. A detailed description of the hypercube topology can be found in [Heller-85]. The hypercube network can be defined as a network of $M=2^n$ processors. Each of the processors is labeled from 0 to M-1 by a unique binary string of length n. There is a bidirectional communication link between two processors if and only if the binary strings associated with them differ in exactly one bit position. The hypercube architecture offers many advantages. First, it has a high degree of reliability. If a processor on the network is dead, only the information that is exclusively associated with that processor is inaccessible. All other processors can communicate among themselves. Second, the communication among the live processors is guaranteed as long as the number of dead nodes is less than $\log_2 n$. Third, the throughput of the network increases linearly with the quantity of processors. Finally, it provides a high degree of parallelism due to its symmetric nature and the tight coupling of the processors.

The database and all its indexing information is represented by one logical file. This file is partitioned into many small fragments, each residing on a separate small disk. Each disk is associated with a fairly powerful processor. The number of disk-processor pairs is sufficient to accommodate the totality of the database. In order to minimize slow disk accesses, each disk-processor pair is associated with a large cache memory, allowing a semi-assossiative reference. Each processor can retrieve the information from the disk, perform the necessary processing on the data and deliver the result to the user. Similarly for updates: the processor verifies all the relevant integrity constraints and then stores the updated information on the disk.

A processor might not have all the required information on its local disk. In such cases, it will find out what processors have access to the information and will transmit a request to the processors for that information. Many database fragments are queried or updated concurrently.

5. Implementation

We use INMOS transputers as the processors for the implementation of LSDM. Each INMOS transputer consists of a fast processor (ten MIPS), memory up to four megabytes and four buffered 20-megabaud com-

munication links. The links may be interconnected directly or via software switches to other transputers or peripheral devices such as disk drives. We connect the transputers in a hypercube network.

The native language of the transputers is OCCAM. This language provides a powerful set of primitives for specifying process concurrency and synchronization, and has the additional advantage that a program written for a single processor system need only be re-compiled to run on a multi-processor system.

We plan to build a massive hypercube of 1024 processors. Each processor will have ten neighbors. Hence the number of communication links of each transputer should be increased to at least ten. Additional links are required for connecting a transputer to a disk. The INMOS COO4 board can extend a communication link to 32 lines, but it requires external control signals to select an appropriate line and does not recognize incoming signals. We are considering other hardware components to extend the number of links for a transputer.

Currently we are emulating LSDM architecture and perform statistical runs on the system. This will give the average size of messages and will help to decide the optimal packet size for the hypercube network.

6. Conclusions and Further Research

The LSDM attempts to alleviate certain problems inherent in contemporary database machines. The prime features which make this architecture promising are the high degree of parallelism, reliability, and the scalability which enables us to increase computing power simply by the addition of hardware resources.

Currently, further work continues in detailed design and simulation of the LSDM and in the development of the algorithms required for the validation of update transactions against constraints, transformation of queries into the internal level, and optimization and distribution of queries.

REFERENCES

[Abiteboul-84] S. Abiteboul and R. Hull. "IFO: A Formal Semantic Database Model", TR-84-304, Computer Science Department, University of Southern California, 1984.

[Abrial-74] J.R. Abrial, "Data Semantics", in J.W. Klimbie and K.L. Koffeman (eds.), *Data Base Management, North Holland, 1974.*

[Baru-87] C.K. Baru and O. Frieder. "Implementing Relational Database Operations in a Cube-connected Multicomputer", *Proceedings of the International Conference on Data Engineering,* Los Angeles, Calif., Feb 1987.

[Benneworth-81] R.L. Benneworth, C.D. Bishop, C.J.M. Turnbull, W.D. Holman, F.M. Monette. "The Implementation of GERM, an Entity-relationship Data Base Management System". Proceedings of the Seventh International Conference on Very Large Data Bases. (Eds. C. Zaniolo & C. Delobel.) IEEE Computer Society Press, 1981. (pp 465-477)

[Bracchi-76] Bracchi,G., Paolini, P., Pelagatti, G. "Binary Logical Associations in Data Modelings". In G.M. Nijssen (ed.), *Modeling in Data Base Management Systems.* IFIP Working Conference on Modeling in DBMS's, 1976.

[Chan-82] Chan,A., Danberg,Sy, Fox,S., Lin,W-T.K., Nori,A., and Ries,D.R. "Storage and Access Structures to Support a Semantic Data Model" Proceedings of the Eighth International Conference on Very Large Data Bases. IEEE Computer Society Press, 1982.

[Goodman-81] J.R. Goodman and C.H. Sequin. "Hypertree: A Multiprocessor Interconnection Topology", *IEEE Transactions on Computers,* Vol. C-30, No. 12, pp. 923-933, 1981.

[Hammer-81] M. Hammer and D. McLeod. "Database Description with SDM: A Semantic Database Model", *ACM Transactions on Database Systems,* Vol. 6, No. 3, pp. 351-386, 1981.

[Heller-85] S. Heller. "Directed Cube Networks: A Practical Investigation", CSG Memo 253, M.I.T., July 1985.

[Hsiao-83] D.K. Hsiao. *Advanced Database Machine Architecture,* Prentice-Hall, Inc. Englewood Cliffs, N.J., 1983.

[King-84] R.King. "SEMBASE: A Semantic DBMS" Proceedings of the First Workshop on Expert Database Systems. Univ. of South Carolina, 1984. (pp. 151-171)

[Le-Viet-83] C. Le-Viet. "The NOAH Database Machine", *Proc. of Compcon Conf.,* pp. 364-368. 1983.

[Lien-81] Y.E. Lien, J.E. Shopiro, S. Tsur "DSIS -- A Database System with Interrelational Semantics". Proceedings of the Seventh International Conference on Very Large Data Bases. (Eds. C. Zaniolo & C. Delobel.) IEEE Computer Society Press, 1981. (pp 465-477)

[Missikoff-83] M. Missikoff and M. Terranova. "The architecture of a Relational Database Computer Known as DBMAC", In D.K. Hsiao (ed.), *Advanced Database Machine Architectures,* Prentice-Hall, Inc. Englewood Cliffs, N.J., pp. 87-108, 1983.

[Nijssen-81] G.M. Nijssen "An architecture for knowledge base systems", Proc. SPOT-2 conf., Stockholm, 1981.

[Nijssen-82] G.M.A. Nijssen and J. Van Bekkum. "NIAM - An Information Analysis Method", in Information Systems Design Methodologies: A Comparative Review, T.W. Olle, et al. (eds.), IFIP 1982, North-Holland.

[Ozkarahan-86] E. Ozkarahan. *Database Machines and Database Management*, Prentice-Hall, Inc. Englewood Cliffs, N.J., pp. 209-215, 1986.

[Rishe-86-PS] N. Rishe. "Postconditional Semantics of Data Base Queries." *Mathematical Foundations of Programming Semantics.* Proceedings of the International Conference on Mathematical Foundations of Programming Semantics, April 1985, Manhattan, Kansas (ed. A. Melton), Lecture Notes in Computer Science, vol. 239. Springer-Verlag, 1986. (pp 275-295.)

[Rishe-86-DN] N. Rishe. "On Denotational Semantics of Data Bases." *Mathematical Foundations of Programming Semantics.* Proceedings of the International Conference on Mathematical Foundations of Programming Semantics, April 1985, Manhattan, Kansas (ed. A. Melton), Lecture Notes in Computer Science, vol. 239. Springer-Verlag, 1986. (pp 249-274.)

[Rishe-87-RM] N. Rishe. "On Representation of Medical Knowledge by a Binary Data Model." *Mathematical Modelling*, vol. 8, 1987. (pp. 623-626)

[Rishe-87-DS] N. Rishe, *Database Semantics. Technical report TRCS87-002, Computer Science* Department, University of California, Santa Barbara, 1987.

[Rishe-88-DDF] N. Rishe. *Database Design Fundamentals: A Structured Introduction to Databases and a Structured Database Design Methodology.* Prentice Hall, Englewood Cliffs, NJ, 1988.

[Shipman-81] D.W. Shipman. The Functional Data Model and the Data Language DAPLEX, *ACM Transactions on Database Systems*, v. *6*, no. 1, 140-173, 1981.

[Su-79] S.Y.W. Su, et al. "The architectural features and implementation techniques of the multicell CASSM", *IEEE Transactions on Computers*, Vol. C-28, No. 6, pp. 430-445, 1979.

[Su-88] S.Y.W. Su. *Database Computers: Principles, Architectures & Techniques* McGraw-Hill, New York, 1988.

[UNISYS-87] UNISYS Corp. The Database Management System SIM. 1987.