Modelling Relational Databases on Multicomputer Architectures

Hyoung Jhang, Tag Gon Kim, and Raymond H. Dean
Dept. of Electrical and Computer Engineering
University of Kansas
Lawrence, KS 66045

ABSTRACT

This paper presents a methodology for the design and performance evaluation of alternative relational algorithms on multicomputer architectures. The methodology employs a unified knowledge representation scheme, called System Entity Structure (SES) and Component Model Base (MBase). Based on this representation scheme, we demonstrate how the development of simulation model of multicomputer architectures can be done systematically.

1. INTRODUCTION

During the last decade, many researchers attempted to improve the performance of database machines by developing specialized database machine architectures. Usually, bottlenecks are found and warts are added to correct them [Boral, 1984]. The database machine architectures become moving targets and makes hard to predict their performance. The attempt failed to provide one good solution to enhance the performance of DBS.

Recently, such failure has motivated a new research area of improving performance by massively connecting general purpose computers in known interconnection schemes [Hwang, 1984; Hayes, 1986] and exploiting more parallelisms. As such, a considerable amount of work has been done in developing strategies for the parallel execution of database operations [Baru, 1987; DeWitt, 1985; Richardson, 1987; Valduriez, 1984]. It is not only promising in the economical point of view, but also each component and the interconnection are well-defined in its operation. More interestingly, there exist more possible ways of executing one query than on the conventional architectures. This leaves more chances of optimization of a query [Ullman, 1982]. However, this flexibility of query execution also complicates the performance evaluation of such architectures.

Although benchmarking can make reliable predictions of the performance of database machines, its use is only limited to the existing architectures. Analytical modelling can provide useful insights into the design of multicomputer database machines. Still, bottlenecks can not be easily found this way.

The design of multicomputer architectures yields many alternatives depending on the number of processors and the type of interconnection scheme. Also, it leads to many possible of executing queries. This vast choice of executing a query complicates the construction of the simulation model. However, there exists uniformity of communication path between neighbors. This paper describes a systematic methodology for constructing simulation models of database systems on multicomputer architectures using such uniform pattern of coupling. The presented methodology is based on the SES/MBase knowledge representation scheme. The examples are implemented in the DEVS-Scheme [Kim and Zeigler, 1990], an implementation of the DEVS (Discrete Event System Specification) formalism in Scheme.

This paper is organized as follows. In section 2, we present the overview of a unified knowledge representation scheme in modelling and simulation, System Entity Structure (SES) and Component Model Base (MBase), which allows a database designer to represent architectural and behavioral knowledge of systems separately. In Section 3, query model and its execution are presented, which is used to construct a set of queries. In Section 4, simulation models of multicomputer database architectures are presented based upon the SES/MBase concept. We illustrate its use by applying the methodology to construct the simulation model of ring and cube architectures.

2. SES/MBASE FRAMEWORK

The System Entity Structure (SES) and Model Base (MBase) framework, proposed by Zeigler [Zeigler, 1987;Kim and Zeigler, 1990], is a knowledge representation scheme for structure and behavior of a system. The SES can record three kinds of relationships; decomposition, multiple decomposition, and specialization. Decomposition represents a coupled model made of smaller entities and indicated by single vertical line. The specialization relationship represents a way in which a general entity can be categorized into special entities and indicated by a double vertical line. Multiple decomposition is a special case of decomposition relationship, which represents a special entity consisting of a collection of homogeneous components. It is indicated by a triple vertical line. A coupling scheme is attached to the decomposition relationship, which defines how they are connected to form the larger model.

Model base [Ziegler, 1984], a reusable behavioral knowledge base, has a set of models that are either atomic (models with no component models) or coupled (models with component models). The model in the set represent behavioral knowledge; how models behave when they

receive stimuli. New models can be saved in, and later retrieved from, the model base. Models so retrieved may be used to create isomorphic models that can be either atomic or coupled. Model behavior so retrieved will be attached to corresponding model structure to comprise a complete model that functions as an abstract simulator. A coupling scheme is attached to a coupled model, which defines how smaller models are connected to form the larger conceptual model of simulation.

The structural knowledge saved in the SES base guides the synthesis of the final simulation model. The user is queried for the multiplicity of the multiple relationship and the selection of a particular entity of the specialization relationship. This process is called pruning.

3. QUERY MODEL

Unlike the conventional database machines, more explicit control of query execution is required. Especially in the parallel processing of queries, a complex control of query execution and addition processing stages are required. Processing of a query is done in five basic cycles of the DBS:

1) query preprocessing - query is compiled if not precompiled,
2) retrieving tuples page by page,
3) processing tuples in the retrieved page,
4) distributing non-local tuples to neighbor nodes if necessary,
5) transmitting tuples to the host.

3.1 Query Representation

The entities of interaction between the host and the DBS is a query sent from the host and result tuples made available to the host in the output buffer of each node. The proposed methodology defines a set of queries, Q = {Qi | i = positive integer}, which is passed to each node as a token. The set of predefined queries resembles the one of the Wisconsin Benchmarks [Bitton, 1983]. A query may involves more than one relation. A query is represented by a set of parameters which describes the load of the resources with respect to the query. The name of relational algorithm is included in the set of parameters describing each query and the relational algorithm is executed when the query is passed to the node. A query is represented as follows:

Qi = {ALG, {Rj} | i, j = positive integers}
Rj = {Nj, Lj, Cj, Pj | j = positive integer}

where ALG identifies a specific relational algorithm, Nj is the cardinality of the relation Rj, Lj is the average tuple length, Cj is the average number of bytes to be computed, and Pj is the average number of tuples to be examined per page.

Each relation in the query representation defines the structure of the relation. All the relations involved in the relational operation must appear in the set with the name of the relational algorithm. The resultant relation must also appear in the set as the last item. For the result relation, Cj and Pj are ignored.

Suppose a relation, STUDENT, which has 5 attributes; LAST(15:CHAR), FIRST(15:CHAR), ID(6:CHAR), DEPT(4:CHAR), and ADDR(24:CHAR). (LAST (15:CHAR) means the field is 15 byte long and its type is character.) Let's represent a query, SELECT * FROM STUDENT. This query is represented as Q10 = {SELECT, {(1000, 64, 0, 8), (1000, 64, -, -)}} in the above representation. This is interpreted as follows. It uses the relational algorithm, SELECT, and it involves only one relation, which has 1000 tuples and 64 bytes long. No comparison is to be made to be selected (0 selection time) and all the tuples in a page is to be examined based on the page size of 512 bytes. This indicates the sequential processing since all the tuples are examined in each page.

Suppose that the relation STUDENT has ID as a primary key. The query, 'SELECT * FROM STUDENT WHERE ID = 43*', is represented as Q11 = {SELECT, {(1000, 64, 4, 1), (100, 64, -, -)}}. This query is interpreted as follows. It involves the same relational algorithm and relation as in the previous example, but comparison is done on the field of 4 characters long. 100 out of 1000 tuples are selected (selectivity = 10%). Since 1 out of 8 tuples in each page is to be examined to result in 100 tuples, its access is based on the primary key.

3.2 Relational Algorithm Representation

In the proposed methodology, relational algorithm is represented in a state transition diagram. In the diagram, each node represents passage of simulation time which is depends on the relation. Each arc indicates the logical sequence of the query execution. The feedback with a number attached to an arc indicates the number of iterations on a particular path. This depends on the relation and/or number of neighbors.

Each node in the multicomputer database machine architectures allocates one output buffer for each neighbor. Output buffer is numbered 0 through M-1, where M is the number of neighbors. Output buffer i contains tuples hashed into those buckets whose addresses are identical to the address of its associated neighbor. The figure 1 illustrates a hybrid-hash join algorithm[Baru, 1987; DeWitt, 1985; Richardson, 1987; Valduriez, 1984]. The join algorithm is done basically in 8 steps. First, read local tuples of the smaller relation into the input buffer page by page. A hash table is built out of tuples hashed into the local bucket. Other tuples are placed into its output buffer. When all the hash table is built out of all the local tuples, tuples in output buffers are sent to its neighbors. At the same time, tuples sent from its neighbors are received and placed into the input buffer. Sending/receiving continues until there are no more tuples to be received or sent. Tuples in the input buffer are processed to complete the hash table in the same manner. If tuples are uniformly distributed throughout nodes, construction of the hash table in each node will be completed at almost at the same time. Tuple balancing techni-

que can be applied to achieve uniform distribution of tuples.

The same process can be applied for the larger relation except for building the hash table, tuples addressed to the local bucket are directly used to probe the hash table of the smaller relation. Passage of simulation time has to be expressed in each state using simulation macro, hold(time_of_hold), which time_of_hold has to be calculated based on the structure of the relation passed as parameters in query.

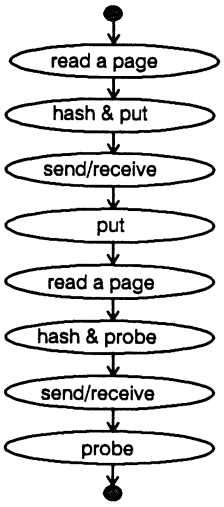


Figure 1. Hybrid-Hash Join Algorithm.

4. SIMULATION MODEL

The simulation model of this paper is to compare the performance of alternative relational algorithms on various multicomputer architectures and measure the impact of the number of processors and the type of interconnection scheme have on the performance. The simulation model is concentrating on the performance of different types of queries like the ones of the Wisconsin Benchmarks [Bitton, 1983]. So, the simulation methodology does not include workload modelling.

Each node consists of a processor, main memory, secondary storage, and two independent communication processors capable of transmitting/receiving data simultaneously via a number of links. The number of links is determined by the type of interconnection scheme and the number of processors.

The simulation model consists of hardware and software models. The software model defines a set of queries each of which can be selected to measure its performance on any multicomputer architecture. A query is represented by a set of parameters as described in the previous section. The hardware model is systematically created when the number of processors and the type of interconnection scheme is selected.

Figure 2 shows the SES representation of multicomputer database systems, MUL-DBS. It shows that MUL-DBS consists of a model, EF (EXPERIMENTAL FRAME) and MUL-HARD. Further, EF consists of TRANS and QUERIES. TRANS model measures the performance and detects the end of the simulation period, while the QUERIES model defines a set of queries.

MUL-ARCH is decomposed into SCH and NODES. The SCH contains several known interconnection scheme such as ring, cube, and star [Hwang, 1984; Hayes, 1986]. The NODES consists of multiple NODE's. Further, the NODE model is decomposed into PROC and LINKS. The PROC model will be defined into the model base. The LINKS model consists of multiple LINK's (M*). The interconnection scheme and the number of processors determine M*, where * indicates that it is determined by the system when the number of processors are given by the user.

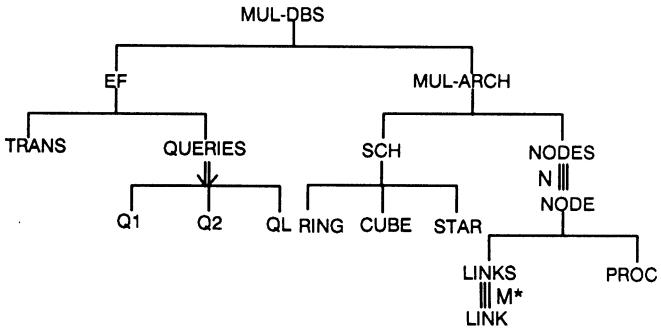


Figure 2. SES of Multicomputer Database Systems.

We do not consider the multiprogramming level of query execution since the interest of study lies on how each type of query performs on alternative architectures. The type of interconnection scheme defines the pattern of interconnection, represented by a coupling scheme. Even if different connection scheme defines different number of neighbors depending on the number of processors, there lies uniformity of such pattern of coupling between neighbors. This uniform pattern is used to develop the NODE systematically for any type of interconnection scheme.

As we mentioned earlier, the NODE model, a coupled model, is decomposed into LINKS and PROC. LINKS is decomposed into multiples of LINK, whose number is determined by the type of interconnection scheme and the number of processors connected. Figure 3 shows such formula of defining M for ring, cube, and star, where N denotes the number of processors.

ring: M = 2
cube: M = LOG2(N)
star: M = N -1

Figure 3. Formula of M.

When the type of interconnection scheme and the number of processors are given, M is determined using the formula and the connections between neighbor nodes are established. The coupling schemes are added to the NODE and NODES using the uniformity of coupling pattern. The algorithm is shown in Fig. 4.

```
FOR i=0 TO M-1
   add a coupling (LINKi.O1, NODE.Oi)
   add a coupling (NODE.Ii, LINKi.I1)
   add a coupling (LINKi.O2, PROC.Ii)
   add a coupling (PROC.Oi, LINKi.I2)
END-FOR
```

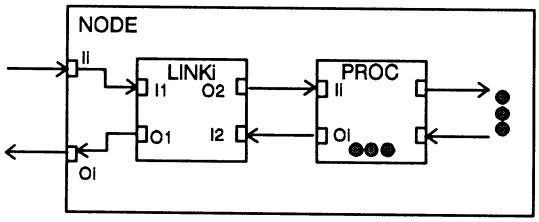


Figure 4. Node Coupling Algorithm.

Each node connected in ring architecture has 2 neighbors and so 2 bidirectional links (M=2.) Figure 5 shows a ring of 8 nodes and the model, NODE in detail. Similarly, each node in cube architecture has Log2(N) neighbors and links. Figure 6 shows a cube of 8 nodes and the model, NODE.

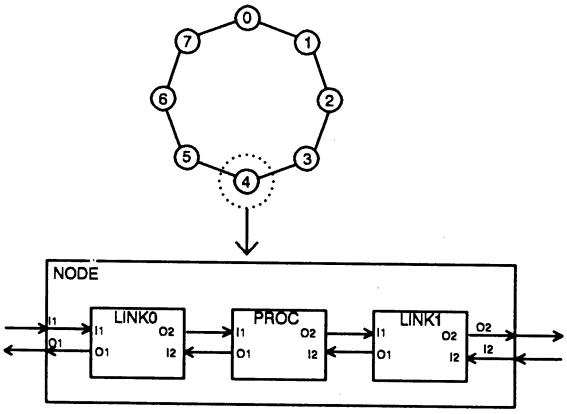
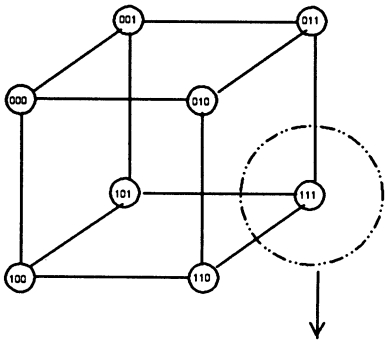


Figure 5. A Ring of 8 Nodes.



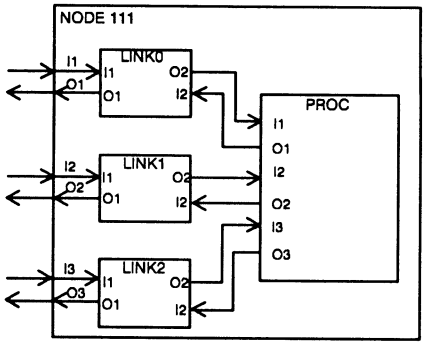


Figure 6. A 3-cube of 8 Nodes.

5. CONCLUSIONS

In this paper, a systematic way of constructing simulation models of multicomputers based on SES/MBase formalism was presented. Such methodology greatly eases the process of evaluating performance of alternative relational algorithms on various multicomputer architectures and thereby finding optimal architectures of the desired performance.

REFERENCES

Bitton, Din, D. J. Dewitt, and C. Turbyfill (1983), "Benchmarking Database Systems: A Systematic Approach," *Proc of the International Conference on Management of Data*, pp. 176-185.

Boral, Haran and D. J. DeWitt (1984), "A Methodology for Database Performance Evaluation," *Proc of the ACM-SIGMOD International Conference on Management of Data*, pp. 176-185.

C. K. Baru and O. Frieder, "Implementing relational database operations in a cube-connected multicomputer," *Proc. IEEE 3rd Int. Conf. Data Eng.*, Feb. 1987, pp. 36-43.

D. J. DeWitt and R. Garber, "Multiprocessor hash-based join algorithms," *Proc. 11th Int. Conf. Very Large Data Bases*, 1985, pp. 151-164.

Hawthorn, P.B. and D.J.DeWitt (1982), "A Performance Analysis of Alternative Database Machine Architectures," *IEEE transactions on Software Engineering*, SE-8, No. 1 pp. 61-78, Jan.

Hwang, K. and Faye A. Briggs (1984), *Computer Architecture and Parallel Processing*: McGraw-Hill.

J. Hayes, et al., (1986), "Architecture of a hypercube supercomputer," *IEEE Conf. Parallel Processing*, 1986, pp. 653-660.

J. Richardson, H. Lu, and K. Mikkilineni, "Design and evaluation of parallel pipelined join algorithm," *Proc. ACM SIGMOD Int. Conf. Management Data*, 1987, pp. 399-409.

Kim, Tag Gon and B. P. Zeigler, (1989), "Knowledge-based Environment for Investigating Multicomputer Architectures," *Information and Software Technology*, vol. 31, no. 10.

Kim, Tag Gon and B. P. Zeigler (1990), "The DEVS-Scheme Simulation and Modelling Environment," in *Knowledge Based Simulation: Methodology and Applications* (Fishwick, P.A. and R.B.Modejeski eds): Springer Verlag. New York, NY.

P. Valduries and G. Gardarin, "Join and semijoin algorithms for a multiprocessor database machines," *ACM Trans. Database Syst.*, vol. 9, pp. 131-161, Mar. 1984.

Ullman, J.D. (1982), *Principles of Database Systems* (2nd ed.): Computer Press.

Zeigler, B. P. (1984), *Multifacetted Modelling and Simulation,* London, UK and Orlando, FL: Academic Press.

Zeigler, B. P. (1987), "Hierarchical Modular Discrete Event Modelling in an Object Oriented Environment," *Simulation*, vol. 49, no. 5, pp. 219-230.