ON RDBM DELTA'S RELATIONAL ALGEBRA PROCESSING ALGORITHM

Shigeki Shibayama, Takeo Kakuta, Nobuyoshi Miyazaki
Haruo Yokota, Kunio Murakami

Research Center
Institute for New Generation Computer Technology (ICOT)

1. Introduction

At ICOT, detailed design of a database machine "Delta" is in progress. This paper presents the processing procedure by which the relational algebra operations are performed. As is described in [1],[4], Delta is planned to be used as a research tool in the intermediate stage of the Fifth Generation Computer Project. This fact requires that Delta fulfill a two-fold demand. One is to develop a hardware-oriented relational database machine and the other is to make it operational in the sense that it should support intrinsic characteristics of database systems such as data recovery and so on. In this paper, how to manipulate such features of databases in a hardware oriented relational database machine is also briefly described.

2. Implementation Design

Delta's global architecture is shown in Fig.1. This figure, however, only shows the concept of the global hardware configuration and not the actual machine implementation. The difference between the global design and implementation design is mostly due to hardware development problems. As the schedule is quite tight, we had to utilize existing computer resources to make up the components in the global architecture. A great many man-hours will be needed to finish a totally new piece of hardware.

The Interface Processor will be implemented using a one-board minicomputer with on-board 512KB main memory. The Control Processor will also be implemented by the same minicomputer with 1 MB main memory. The Relational Database Engine (RDBE) is a new piece of hardware, our idea being based on an algorithm using the hardware supported merge-sorting implemented by the RDBE. Most hardware intensive efforts are being done for the implementation of RDBE, in the sense that it will become the first practical hardware relational database engine. RDBE will also use minicomputer to control the RDBE's hardware resources. In order to manipulate various relational database processing requirements, for instance, floating-point data calculations, RDBE could not help but become a little sophisticated.

The Hierarchical Memory (HM) subsystem is implemented using a general-purpose computer as a controller (HMCTL), a large amount of fast semiconductor random access memory in the form of a semiconductor disk (SDK) and large capacity moving head disks. The SDK is used for temporarily storing classes of relations generated and managed in Delta. The SDK will be non-volatile (at least from software point of view) to avoid disk accesses invoked by write-through storage management. We realize there remains a lot to be investigated and researched to make a real hardware-oriented HM. So we decided to simulate the HM using a general-purpose computer system for collecting performance data and making the points to be improved clear in this research. One of the most decisive factors in choosing a general-purpose computer as the HM is that it provides an operating system containing control software on state-of-the-art disks, the capacity of which is over 2GB per unit.

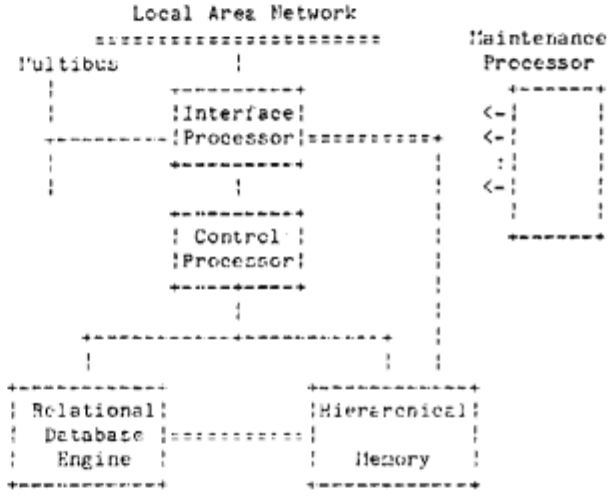

Fig.1 Delta global architecture

Another major problem facing Delta is the communication problem between Delta subsystems. As Delta is a distributed function type relational database machine, communications between subsystems are of great importance for performance. In this case also, creating an overhead-free communication protocol, using both hardware and software, is ideal. Instead, communications between IP, CP and RDBE are done by a standard bus, minimizing software communication overhead by the means of I/O software system modifications. The highest transfer rate is required between RDBE and HM subsystems because through this interface data transfer intensive operations are performed most frequently. We adopted a channel interface for this interface with slight modifications. Through this interface, data are sent back and forth in a stream at a maximum rate of around 3 MB/second. Each RDBE (a few RDBE's will be provided in the course of implementation) will be provided with two independent I/O channels for input and output. RDBE is intelligent enough to schedule its own I/O activities. A Maintenance Processor will be attached to Delta

for supervising the whole Delta system, an important subsystem for RAS and data collection.

3. Detailed Processing algorithm in Delta

The fundamental processing scheme of Delta is described in [2],[3]. We have established a subcommand set for each of the RDBE and HM subsystems. The general command processing sequence is as follows:

(1) Receive a command-tree from a host (a SIM which made the query) by IP
(2) Send the command-tree to CP
(3) Analyze the command-tree and generate subcommands in CP
(4) Issue subcommands to RDBE and HM from CP
(5) RDBE and HM set-up
(6) HM activation
(7) Data stream transfer to RDBE from HM
(8) Processing the data stream in RDBE ((parallel to (7))
(9) Result transfer to HM from RDBE (parallel to (7) and (8))
(10) Tuple-reconstruction (if necessary) in HM directed by CP
(11) Result tuple transfer to IP
(12) IP to host transfer via LAN

Note that Delta can receive concurrent queries. This sample sequence will be applied to each such query. If this sequence consists of a transaction, this should be explicitly specified for data management purposes.

4. Subcommands

Subcommands are defined as Delta's inner subsystems' commands. Subcommands are issued, for example, from CP to HM, RDBE to HM, CP to RDBE and so on. Subcommands are used for activating each subsystem in order to operate the Delta system. Fig.2 shows the major HM's subcommands and Fig.3 shows RDBE's major subcommands. Basically, RDBE's subcommand set is based on the set processing algorithm incorporated in RDBE. Delta's non-procedural interface to hosts is supported by RDBE's hardware relational algebra processing algorithm. In effect, RDBE's subcommand set is semantically close to Delta's interface. Corresponding to this RDBE processing requirement, HM ordinarily offers a stream-based interface to RDBE. Generally speaking, HM serves as the storage space not only for relations but also for other information such as directory and logging data in the Delta system and is passive under usual processing. RDBE does not have to worry about such storage dependent parameters as page boundaries, physical addresses and so on. In the procedure shown in the previous chapter, subcommands are the main information passed between subsystems in each step.

5. Data Management in a database machine

As Delta will be used in a practical way, the database should be protected from disasters invoked by human errors as well as hardware malfunctions. RDBMS, sequential inference machine's software, and Delta are both responsible for the security of databases. Mainly RDBMS is responsible for the unauthorized access to a database; Delta is responsible for transaction backout and concurrency control. The Control Processor keeps track of each transaction and if it successfully ends, CP commits the updates made by the transaction. If it fails for some reason, Delta rolls back the updates made so far by the transaction.

Start-Stream-In	Start-Stream-Out
Transpose-To-Tuple	Transpose-To-Attribute
Prepare-Buffer	Release-Buffer
Start-Paged-Stream-In	Start-Paged-Stream-Out
Read-Directory	Write-Directory

Fig.2 Major HM Subcommands

Join-Equal	Join-Greater-Than
Restrict	Restrict-Range
Restrict-Equal	Restrict-Equal-Immediate
Sort	Unique
Difference	Update
Aggregate-Function	

Fig.3 Major RDBE Subcommands

Hardware malfunctions are classified into categories in Delta. As Delta has some subsystems, error could occur in 1) intra-subsystem and 2) inter-subsystem. Non-fatal intra-subsystem errors, basically, are detected and reported autonomously to MP. Transient inter-subsystem errors such as data parity errors will be detected by the receiver and be retried. Watchdog timers will be installed at major interfaces to supervise silent death as well as heavy inter-subsystem errors. SDK in HM is non-volatile, so even if a crucial malfunction occurs in a processor, data consistency could be maintained afterwards using a recovery process.

6. Conclusion

We described implementation decisions, the overall relational algebra processing procedure and data management scheme in the Delta detailed design. A number of problems must be solved during this stage. However, these efforts are of great significance, we believe, in orderd to give birth to a working, new concept machine.

7. references

[1] Kakuta, T. et al.,"A Relational Database Machine "Delta" (I)", Proc. of 26th IPSJ conference, 4F-6, in Japanese.
[2] Shibayama, S. et al.,"A Relational Database Machine "Delta" (II)", ibid.,4F-7, in Japanese.
[3] Yokota, H. et al.,"A Relational Database Machine "Delta" (III)",ibid.,4F-8, in Japanese.
[4] Murakami, K. et al.,"A Relational Database Machine: First Step to Knowledge Base Machine", Proc. of 10th International Symposium on Computer Architecture, June 1983, pp. 423 - 425.