CUCS-19-81

ON THE LONG-TERM IMPLICATIONS OF DATABASE MACHINE RESEARCH

David Elliot Shaw
Department of Computer Science
Columbia University
New York, New York 10027

Reprinted from *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, Vancouver, August 1981.

ABSTRACT

It is argued that the long-term import of current research on specialized hardware for database management may extend far beyond what is now considered the immediate province of database managament systems. In particular, the potential support which "database machines" may provide for very high level mechanisms for the description of information processing tasks is considered and exemplified with an operational system for knowledge-based information retrieval.

KEYWORDS

Database machines; mathematical logic; parallel architectures; VLSI systems; program description; artificial intelligence.

INTRODUCTION

During the past few years, research involving the design of specialized hardware adapted to the requirements of large-scale database managament has emerged as one of the most rapidly growing research areas in the field of computer science. It seems clear that the very considerable attention which this research has recently begun to attract within the academic, industrial and government communities stems largely from the enormous and rapidly increasing reliance of our society on the systematic management and manipulation of large collections of data. Indeed, the fact that such machines may, in the near or intermediate future, lead to much higher cost-to-performance ratios in conventional data-base management applications is of great interest in itself. It is our suspicion, however, that the long-term implications of certain recent advances in this area may extend far beyond the scope of contemporary database management systems.

This paper will examine and exemplify the potential long-term significance of certain investigations regarding database machines in the context of some very fundamental problems involving the description and execution of information processing tasks. Our conjectures regarding the potential implications of recent findings in this area derive largely from the observation of a very interesting relationship between the general and powerful descriptive mechanisms of predicate logic and the primitive operations of a relational algebra. In particular, we will examine the "high-level"

import of nonstandard machine architectures which support the highly efficient evaluation of relational algebraic operators.

DESCRIBING INFORMATION PROCESSING TASKS

One of the most consistent trends in the history of computer science can be characterized as a gradual progression from procedural to declarative mechanisms for the description of information processing tasks. Feigenbaum (1974), for example, has noted that the human user is interested in specifying what the computer is to do, without regard for the details of how the desired task is to be performed. The chronological sequence beginning with machine language programming, extending through the use of assembly languages and the subsequent development of high-level language compilers, and leading to contemporary research on artificial intelligence and automatic programming, he argues, can be viewed as a systematic journey from the "how" to the "what" end of this spectrum.

Allowed to fantasize about the kind of computing tools we would some day like to have at our disposal, we might ask for the ability to communicate our information processing needs to the computer system in much the same way as a data processing manager now communicates with a human programmer, making references to domain-specific entities and relationships in the "real world"--such constructs as customers, inventories and accounts, for example -- in a natural, "human-like" manner. As the cost of

computer hardware becomes increasingly dominated by the time and expense involved in translating "high-level" task specifications into working computer programs, the immense potential value of even modest advances in this direction becomes increasingly obvious.

As software tools begin to claim a larger share of our attention, and as the cost-to-performance ratio for digital hardware continues to drop dramatically, it is natural to ask whether there is any way that our unprecedented capabilities in the area of computer hardware might be mobilized to help meet such unprecedented challenges in the area of computer software. In this regard, it is interesting to observe that although the cost of computer hardware has decreased by a factor of many thousands during the past few decades, the fundamental architecture of the computer itself has remained essentially the same, based closely on what is generally referred to as the von Neumann model of computation. Our own work during the past few years has been based in large part on the suspicion that characteristics not intrinsic to the process of computation, but associated instead with the particular constraints of the von Neumann machine, may have made it difficult for computer scientists to conceive of tools which might well support major advances in program description. It is our feeling that the most important long-range implications of current research on non-von Neumann "database machine" architectures may involve not improvements in the efficiency of existing information processing applications, but support for very high level program description tools which might otherwise be impractical to implement at all.

A KNOWLEDGE-BASED RETRIEVAL LANGUAGE

A surprisingly large share of the kinds of information processing activities with which both human and automated data processors are charged may be viewed as involving various kinds of knowledge-based matching or retrieval problems. Although meaning-based matching may, in a COBOL-based inventory control system for airplane parts, be deeply embedded within program loops and sort routines, and hence difficult to recognize as such, the fact remains that a large proportion of the CPU cycles which will be expended in any particular run might be thought of as identifying appropriate records for manipulation on the basis of domain-specific criteria -- in our example, criteria involving, say, airplanes, motors and aircraft-specific part-whole relationships. Our recent research has focused on a concrete task of manageable scope, but which nonetheless captures many of the most important characteristics of this important aspect of program description: the knowledge-based document retrieval application.

The user of a knowledge-based document retrieval system formulates a high-level pattern description to be matched against all descriptions contained in a large target collection (the set of books in a computer science library, for example). In contrast to a conventional retrieval system, in which mechanical, "syntactic" criteria are adequate to identify all matching documents, the success of a match in a knowledge-based document retrieval system depends in general on domain-specific entities and relationships. In the case of the computer science library, for example, the

system might be required to "know about" such entities as computers, algorithms, programmers and storage devices. Certain characteristic attributes of these entities (the "storage medium" attribute, whose values differ for different kinds of storage devices, for example) might also be included in this domain-specific knowledge. Among the typical kinds of relationships which might be embodied in the knowledge base of such system is the fact that a tape drive is a particular kind of storage device whose storage medium is always magnetic tape; one simple deductive inference involving this relationship might establish the fact that a pattern description in which the subject of the document is described as involving a storage device with magnetic tape as its medium would be satisfied by a target description in which a tape drive appeared in the corresponding position.

The description language which we used in our experimental implementation of a knowledge-based retrieval system (Shaw, 1980a) is based closely on the Language KRL (for Knowledge Representation Language) (Bobrow and Winograd, 1977). While space will not allow a detailed definition of this knowledge-based description language, the following sample document description, presented without explanation, may suggest some of its most important features:

```
a  Document with
     authors =
       set-with-all-of Thompson, Walters
       set-of an Engineer
     countries-of-publication
       set-with-any-of USA, Great-Britain
     subject involves
       an Invention with purpose =
         or Power-generation, Power-transmission
     copyright-dates =
       set-with-exactly 1935, 1962
a  Survey-article
```

MATHEMATICAL LOGIC: THE ESSENTIAL INTERMEDIARY

In the interest of general applicability to problems other than document retrieval, the rules defining the semantics of matching within our knowledge-based description language were not embedded inextricably within the code of the retrieval system, but were instead formulated as an independent, separable set of axioms expressed in a restricted first-order predicate calculus. Twnety-two such axioms were required to describe the matching rules for our knowledge-based description language. One of these axioms (which embodies the basis on which a target description of the form "a Storage-device with medium = Magnetic-tape" would successfully match a less restrictiv description of the form "a Storage-device", for example) is presented below in an attempt to convey the general flavor of the matching axioms:

$$
\begin{aligned}
&\textit{Perspective-dtor-imp-perspective-dtor (tar-per, pat-per)} \equiv \\
&\quad \exists\ \textit{proto}\ . \\
&\quad\ (\textit{Per-proto (pat-per, proto)} \wedge \\
&\quad\ \ \textit{Per-proto (tar-per, proto)}) \wedge \\
&\quad \forall\ \textit{slot}\ . \\
&\quad\ ((\exists\ \textit{pat-fill}\ . \\
&\quad\ \ \ \textit{Obj-slot-fill (pat-per, slot, pat-fill)}) \\
&\quad\ \supset \exists\ \textit{pat-fill, tar-fill}\ . \\
&\quad\ \ \ (\textit{Obj-slot-fill (pat-per, slot, pat-fill)} \wedge \\
&\quad\ \ \ \ \textit{Obj-slot-fill (tar-per, slot, tar-fill)} \wedge \\
&\quad\ \ \ \ \textit{Dtion-imp-dtion (tar-fill, pat-fill)}))
\end{aligned}
$$

The set of matching axioms serves not only as a modular, perspicuous, and easily specified description of the semantics of the knowledge-based description language, but as an executable "program" for actually carrying out the matching process which it describes.

More precisely, the matching process is carried out by a procedure called LSEC (for Logical Satisfaction by Extensional Constraint), which interprets the set of axioms to identify all target descriptions which match the user-specified pattern description according to the matching rules for the description language, making reference to the domain-specific knowledge base. (Details of the LSEC algorithm are described in Shaw (1980a).)

THE DATABASE MACHINE AS A LOGIC ENGINE

Unfortunately, the LSEC algorithm relies very heavily on the execution of several operations which, on a von Neumann machine, are quite expensive when their operands comprise a large amount of data (as is typically the case in the sort of application with which we are concerned). Specifically, the algorithm repeatedly evaluates the most expensive operations of a relational algebra -- in particularly, the equi-join operation. Thus, while predicate calculus serves a vital role as an "intermediate form" in our approach -- acting as a very general and powerful descriptive tool, on the one hand, and (through its close mathematical relationship to the relational algebra) as a computationally effective mechanism for evaluation, on the other -- we have been forced to consider various alternatives to the von Neumann architecture in order to justify the promise of our approach in realistic applications.

The particular machine architecture we have developed (Shaw, 1979, 1980) which is based on a hierarchy of associative

storage devices, evaluates the "difficult" relational algebraic operators in a highly efficient manner, particularly when the argument relations are large. Specifically, this architecture permits an asymptotic improvement of O(log n), with very favorable constant factors, over the best known evaluation methods for these operators on a conventional computer system, without the use of redundant storage, and using currently available and potentially competitive technology. Although it was the efficient implementation of large-scale knowledge-based operations which originally motivated our design, these results have recently begun to attract attention within the database management community by virtue of the important role played by the more complex relational algebraic primitives within database management systems based on the relational model of data (Codd, 1970). It is this connection with the concerns of relational database systems, together with the close relationship between our work and earlier research on specialized hardware for database management (for example, the RAP (Ozkarahan, Schuster and Smith, 1974), CASSM (Su, Copeland and Lipovski, 1975), DBC (Hsiao, 1977) and DIRECT (DeWitt, 1979) machines) which has motivated our use of the phrase "relational database machine".

It is our conjecture, however, that this area of research may in the long run have an impact extending far outside the immediate province of data base management.

ACKNOWLEDGEMENTS

The author is indebted to Bob Floyd, Don Knuth, Gio Wiederhold and Terry Winograd, of Stanford University, for substantial contributions to the work reported in this paper. This research was supported in part by the Defense Advanced Research Projects Agency under contract MDA903-77-C-0322 to the Knowledge Base Management Systems Group at Stanford.

REFERENCES

Bobrow, D. G., and Terry Winograd (1977). An overview of KRL-0, a knowledge representation language, Cognitive Science, 1, no. 1.

Codd, E. F. (1970). A relational model of data for large shared data banks, Commun. ACM, 13, no. 6.

DeWitt, David J. (1979). DIRECT--A multiprocessor organization for supporting relational database management systems, IEEE Trans. Comput., C-28, no. 6.

Feigenbaum, Edward A. (1974). Unpublished lecture.

Hsiao, David K. (1977). The architecture of a database computer--a summary, Proc. Third Workshop on Non-Numeric Processing.

Ozkarahan, Esen A., Stewart A. Schuster, and Kenneth C. Smith (1974). A data base processor, Technical Report CSRG-43, Computer Systems Research Group, University of Toronto.

Shaw, David Elliot (1979). A hierarchical associative architecture for the parallel evaluation of relational algebraic database primitives, Stanford Computer Science Department Report STAN-CS-79-778.

Shaw, David Elliot (1980). A relational database machine architecture, Proc. 1980 Workshop on Computer Architecture for Non-Numeric Processing.

Shaw, David Elliot (1980a). Knowledge-Based Retrieval on a Relational Database Machine, Ph.D. Dissertation and Stanford Computer Science Department Report (in press).

Su, Stanley, Y. U., George P. Copeland, and G. J. Lipovski (1975). Retrieval operations and data representations in a content-addressed disc system, Proc. Int. Conf. on Very Large Data Bases.