Research and development on knowledge base systems at ICOT

Hidenori ITOH

ICOT Reseach Center
Institute for New Generation Computer Technology

Mita Kokusai Building, 21F,
1-4-28 Mita, Minato-ku, Tokyo 108 Japan

Abstract

This paper describes ICOT's research and development activities on knowledge base systems. Our approach is to investigate knowledge base systems from the viewpoint of logic programming. We have defined several models on the bases of which knowledge base systems will be developed step by step. Two types of knowledge base management software system models are discussed. One is a model combining a logic programming system and a database management software system. The other is an integrated model. Pilot, distributed, and parallel knowledge base machines models are also discussed.

1. Introduction

One of our guiding principles is that logic programming can become a new, unifying principle in computer science[1][2]. This is because logic programming covers computer architecture, new programming styles, programming language semantics, and database semantics. Logic programming will also play an important role in such fields as linguistics and artificial intelligence.

We selected Prolog as the logic programming language to be used as a research tool. From Prolog a kernel language was developed corresponding to a conventional machine language. The kernel language is the center around which hardware and software systems for the Fifth Generation Computer will be developed. Inference and knowledge base machines will be developed as hardware systems; basic and application software will be developed as software systems (Figure 1).

Development is being pursued in three stages. The first, which started in 1982 and ended in 1984, is complete, and we are now in the second stage.

In the initial stage the kernel language KL-0[3] was developed with an object-oriented programming feature added to Prolog. The personal sequential inference machine (PSI)[4][5] was developed as an inference machine based on KL-0, and the relational database

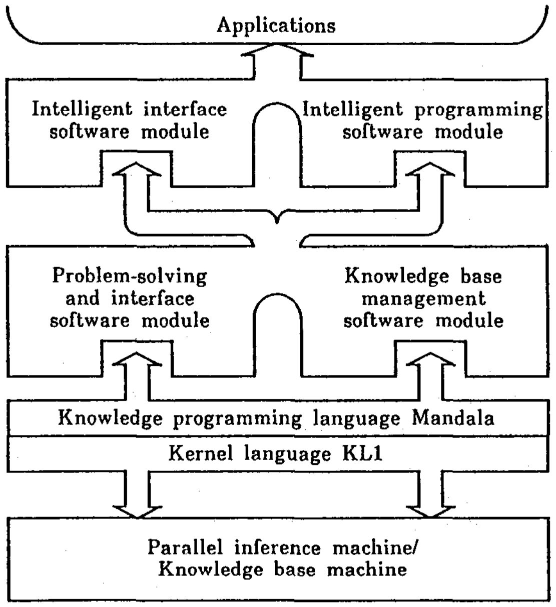

Applications
Intelligent interface software module
Intelligent programming software module
Problem-solving and interface software module
Knowledge base management software module
Knowledge programming language Mandala
Kernel language KL1
Parallel inference machine/ Knowledge base machine

Figure 1. ICOT philosophy: Kernel language / machine / basic software / application software

machine (DELTA)[6] was developed as the first step to a knowledge base machine.

At present we are developing kernel language KL-1[7], which includes a parallel processing feature added to the logical language, a more simplified machine language than KL-0. Based on the know-how obtained through the development of the PSI, the parallel inference machine (PIM)[8][9] is being developed to establish parallel inference technology. ICOT's knowledge base machine (KBM) is being developed within a logic programming environment[10] building on the skills and experience gained in the research on DELTA.

The ultimate target is to develop a prototype of a highly parallel computer for knowledge information processing in VLSI.

2. Requirements for a knowledge base system

What are the necessary conditions for knowledge base system development?

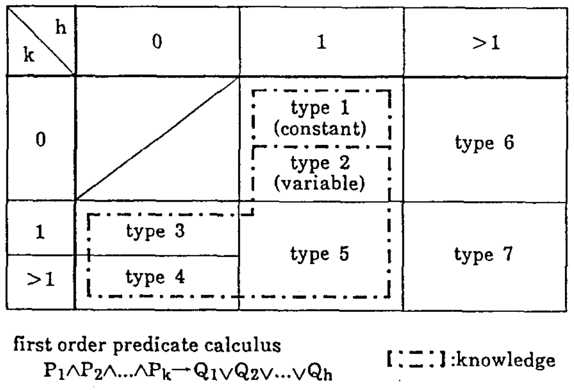
A conventional database system stores a large amount of shared user data, and retrieves data in response to different queries from different users accepted simultaneously. The requirements for a conventional database system include: clarification of the logical basis of the system, efficient use of storage space/management/processing, simple user interface, and a high degree of user program data independence.

The bottom-up approach in which requirements for a conventional database system are reviewed in the light of logic programming is very important. It has been pointed out that data and program, and metadata and object data can be uniformly processed, and that deductive retrieval is possible, if database construction is pursued from the standpoint of logic programming[11].

It is also important to investigate the requirements of a knowledge base system top-down. We must clearly answer the question "What is knowledge?" We must also answer the following questions: How should knowledge be represented? How should it be acquired? How should the knowledge, acquired and represented, be managed? Which query language should be used? How should it be retrieved and processed? How should we define basic operations on knowledge? Where and how should it be stored, physically and logically? Which definitions of data view should be allowed? Which machine architecture should be used for the implementation of these functions?

In domains where the knowledge is relatively scant and limited, these questions have been solved to a certain extent, resulting in various expert systems. However, when the amount of knowledge is large, and the knowledge base has to be shared, the difference in quantity often entails a difference in quality on these issues.

A knowledge base system consists of a knowledge base machine and a knowledge base management software system. Dedicated hardware for efficient knowledge processing is added to the knowledge base machine as a knowledge processing control unit, based on the inference machine. Also, a mass storage mechanism for large amounts of knowledge is added to the knowledge processing control unit. The mass storage is investigated from the standpoint of memory architecture. Parallel processing on the knowledge base is necessary to obtain high performance. The requirements mentioned above should be implemented in query control, optimization algorithms for parallel retrieval/parallel computation, parallel execution management, and the parallel resource management functions of the knowledge base management software system.



k \ h	0	1	>1
0		type 1 (constant) / type 2 (variable)	type 6
1	type 3	type 5	type 7
>1	type 4	type 5	type 7

first order predicate calculus
$P_1 \wedge P_2 \wedge \ldots \wedge P_k \rightarrow Q_1 \vee Q_2 \vee \ldots \vee Q_h$

[ ] :knowledge

Figure 2. Knowledge representation

3. Knowledge representation and knowledge acquisition

We begin with a discussion of the representation of the knowledge to be processed by the knowledge information processing computer.

3.1 Knowledge representation

Various problem-oriented knowledge representation languages have been developed. A knowledge base system must manage and process a large amount of shared knowledge. We selected the first-order predicate calculus as our knowledge representation language. The first-order predicate calculus representation can handle variables and recursive programming can be used to represent an infinity of concepts. Knowledge elements can be represented as logical structures. Actually, we use only Horn Logic, a subset of the first-order predicate calculus, because processing efficiency is an important issue.

The term (relational) data in the following refers to those items represented by type 1 in Figure 2[12], unless otherwise specified. Knowledge refers to those items represented by types 2-5. Type 1 data is sometimes called relational data or facts and type 5 is called a rule. Types 3 and 4 are called queries.

3.2 Knowledge acquisition

Knowledge acquisition is the obtaining of knowledge from experts, representing it in a form which can be processed by computer, and organizing the knowledge in an orderly manner so that a new piece of knowledge can be added or an old piece modified. Integrity and redundancy checks are necessary in a knowledge base. The knowledge acquisition mechanism is indispensable for the growth of a knowledge base in both quantity and quality.

Knowledge assimilation, knowledge accommodation, and knowledge equilibration mechanisms are also necessary for knowledge acquisition. Knowledge assimilation entails assuming that the knowledge already stored in the knowledge base is correct, and obtaining from experts only knowledge consistent with that stored in the knowledge base. Knowledge accommodation entails assuming that the knowledge from the experts is correct, and modifying the existing knowledge base in a consistent and systematic manner. Finally, knowledge equilibration entails repeating knowledge assimilation and knowledge accommodation so that consistency between a knowledge base and experts, or among two or more knowledge bases, is obtained. Integrating these functions into the system is an important task in building a knowledge base. The process of knowledge acquisition into a knowledge base, shown in Figure 3, was implemented on the knowledge base management system KAISER[13] developed in the initial stage. This mechanism was supported by the following basic inference functions: induction, deduction, and abduction realized with meta-programming techniques using Prolog. This work also firmly established the enormous capabilities of Prolog.

These functions were restricted only to facts; however, they should be extended to the rules as a whole.

4. Knowledge base management system models

How should a knowledge base system be developed? System models are defined here with the emphasis on knowledge base management software systems. Models emphasizing control mechanisms are described in Section 6.

4.1 Combined model

A logic programming language system has knowledge representation capabilities and inference functions superior to those of other languages. Relational data processing systems have a clear logical foundation for data handling, and can process large amounts of data. Moreover, knowledge representation in a relational data model can be represented within the framework of a first-order predicate calculus logic programming language. Therefore, a logic programming language system and a relational data processing system have a close affinity and combined model has already been proposed, incorporating the benefits of a logic programming language with those of a relational database system.

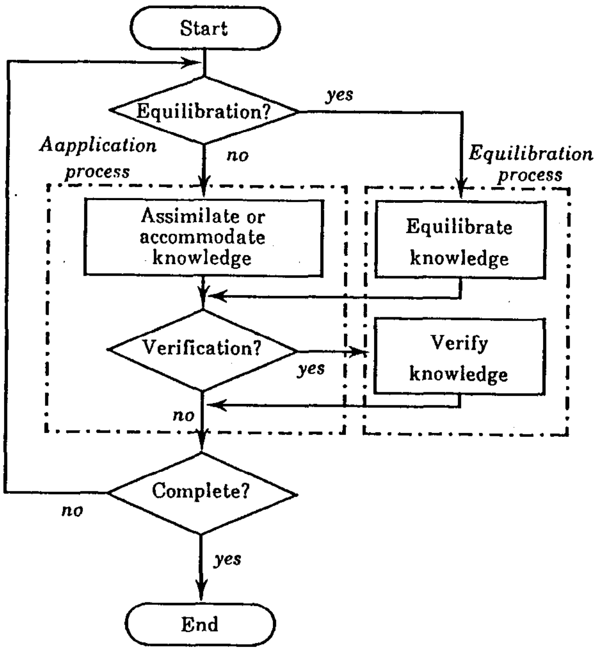


Figure 3. Knowledge acquisition process

Several different models can be developed depending upon the ways in which a knowledge base system manages intentional database (IDB) and extensional database (EDB) and also upon the relationship between the knowledge base system and user program. These models are discussed below.

4.1.1 Combined model with three layers

A combined model with three layers was developed on KAISER/DELTA with the approach taken in the initial stage.

The logic programming language system consisted of a user program and a knowledge base management system, both on the PSI. The relational data system was a relational database management system on the DELTA (Figure 4). The IDB of a user program was managed by the knowledge management program (KAISER), while the EDB was managed by the relational database management system. The knowledge management program converted the Horn clause query to a relational algebraic query using the deferred evaluation method[14]. The knowledge management program and the relational database management system were logically con-

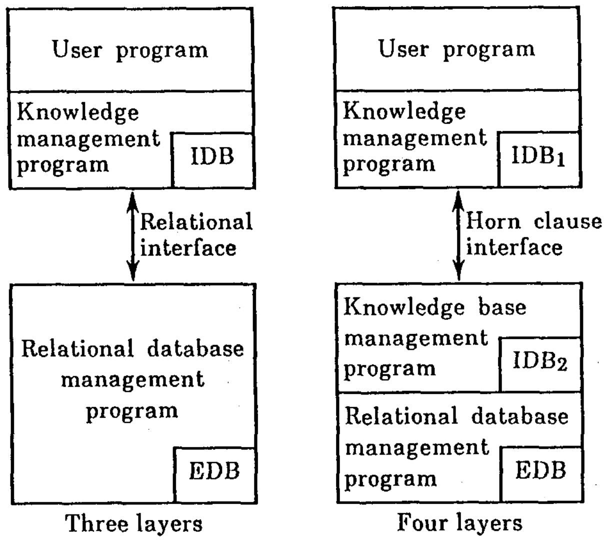

User program
Knowledge management program
IDB
Relational interface
Relational database management program
EDB
Three layers
User program
Knowledge management program
IDB1
Horn clause interface
Knowledge base management program
IDB2
Relational database management program
EDB
Four layers

Figure 4. Layer combined model/combined model with four layers

nected by a relational algebraic command interface, and the DELTA and the PSI were physically connected by a LAN. The relational database management system retrieved the EDB by relational algebraic type commands. Since this approach led to a logically low level interface between a knowledge management program and the relational database management system, problems in overall performance were detected, because a large number of commands and responses had to be transferred through a physically narrow channel. Another problem was that the relational database management system's inference function had poor extensibility.

4.1.2. Combined model with four layers

In the combined model with four layers, the knowledge base management system is situated at the side of the knowledge base system. It is expected that this new architecture will solve the problems found with the combined model with three layers. Better overall system efficiency, and load reductions for the knowledge management and user programs are also expected. A Horn clause interface is used between the inference machine and the knowledge base machine to achieve these objectives. The amount of communication was reduced by as much as possible through the physically narrow channel.

The program structure on the inference machine is the same as that in the combined model with three layers.

The knowledge base management system on the knowledge base machine consists of a knowledge management layer and a data management layer. The knowledge management layer managers a part of the IDB, and the data management layer manages relational data. The IDB in the knowledge management layer contains shared user knowledge and control knowledge (meta-knowledge), provided by the knowledge base machine. The IDB on the inference machine is managed by the knowledge management program KAPPA.

Since the Horn clause interface is used between the inference machine and the knowledge base machine, the knowledge management layer must be able to process Horn clauses. The knowledge management layer converts a Horn clause to an equivalent Horn clause by the Horn clause transfer (HCT) conversion algorithm[15], which is then converted to relational algebraic expressions. Next, the data management layer retrieves relational data using the relational algebraic expressions from the knowledge management layer.

An example of HCT conversion is shown below:

```
(1)
 Query      ?:-ancestor(taro,Y)
 IDB   ancestor(X,Y):-parent(X,Y)
       ancestor(X,Y):-parent(X,Z),ancestor(Z,Y)
       parent  (X,Y):-edb(father(X,Y))
       parent  (X,Y):-edb(mother(X,Y))
 EDB   father relation
       mother relation
```

where edb(father(X,Y)) means that the relation exists in the EDB.

(2) HCT conversion: IDB is converted to the following equivalent clauses by HCT.

```
ancestor(X,Y):-edb(father(X,Y))
ancestor(X,Y):-edb(mother(X,Y))
ancestor(X,Y):-edb(father(X,Z)),ancestor(Z,Y)
ancestor(X,Y):-edb(mother(X,Z)),ancestor(Z,Y)
```

(3) Deriving relational algebraic expression: The above equivalent clauses are converted to the following relational algebraic expressions.

$$\begin{aligned} ancestor_2 = \; & \pi_2 \; ((\sigma_{1=taro}(father)) \\ & \cup \; (\sigma_{1=taro}(mother)) \\ & \cup \; (\pi_{1,4}(\sigma_{1=taro}(father \underset{2=1}{\bowtie} ancestor))) \\ & \cup \; (\pi_{1,4}(\sigma_{1=taro}(mother \underset{2=1}{\bowtie} ancestor))) \end{aligned}$$

The Horn clauses converted by HCT are equivalent to the original Horn clauses, and contain only edb predicates and recursive predicates.

Processing by knowledge management layer can be interpreted as conversion processing from declarative query to procedural form. It also means that the inference function is allowed in a view definition. This indicates the possibility of systematic and higher definition of data view.

Two approaches are possible for implementing the combined model with four layers.

(1) Add a deduction function to the database by adding a logic programming language system to the relational database system.

(2) Integrate the relational database system into the logic programing language system, and take maximum advantage of the inference function of the logic programming language system.

We have employed the second approach to develop the knowledge management system KAPPA, and the knowledge base system PHI-1 on the inference machine PSI (the machine model of PHI-1 is discussed in Section 6). The knowledge management program KAPPA is being developed as a tool for natural language processing and as a theorem prover. It retrieves knowledge from a thesaurus in non-first normal form with approximately one million words, and from an electronic dictionary with approximately a half million words.

4.2 Integrated model

It is necessary to improve the capabilities of the knowledge base system by storing user-shared object knowledge (rules) in the EDB of the knowledge base machine. Two approaches are possible: object knowledge and data may be managed separately, or they may be managed together. The first approach is advantageous when the amount of knowledge is small. It is a natural extension of the combined model with three layers. The second approach is better when the amount of knowledge is large, or when knowledge and data are closely connected logically in small units. Thus, we defined a new integrated model to treat knowledge and data in the same manner. For this purpose we define the term relation, a natural extension of relational data, as follows:

$$T \subseteq K_1 \times K_2 \times \ldots \times K_n$$

K is a set of terms. We call T a term relation. To operate the term relation, a unification relational operation was introduced. The unification relational operation was an extension of the relational data model to the relational knowledge base model, operations of conventional relational algebra, such as join and restriction, are extended to operations based on unification. In other words, equality-check operations between constants are enhanced to unification operations between terms. Thus (equi) join and restriction are extended to unification-join and unification-restriction, respectively[10]. Repeatedly using these operations, the knowledge which want to be take is retrieved. Retrieval using these basic operations is called retrieval by unification RBU. RBU will be used in breadth-first-search, in fields where a large amount of knowledge is processed by shallow inference. Handling treatments of the temporal term relations in RBU, and modularization of sets of terms must also be considered.

5. Retrieve control and update control

A query can be interpreted as a retrieval from a model-theoretic viewpoint, or as a theorem-proving request from the proof-theoretic viewpoint. In both cases, it is important to optimize query processing control. Therefore, concepts of query compilation and knowledge compilation are introduced to optimize the control of query processing.

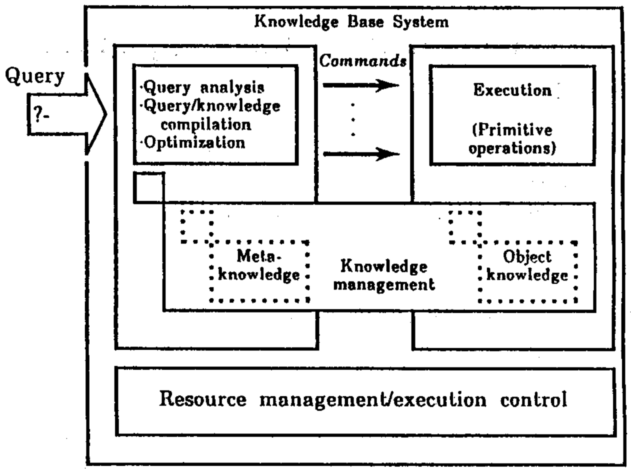


Figure 5. Optimization strategy using meta-knowledge about position of the query

The query language used by the user is a user-friendly problem-oriented language. Query compilation processing entails compiling the problem-oriented language (POL) into the basic operation codes of the knowledge base machine, after semantic and syntactic analysis of the POL expression. From a different angle this also entails compiling various query languages into a single intermediate language within the knowledge base system. That is, the intermediate language is positioned between the kernel language and the knowledge representation language. This intermediate language should be defined taking both knowledge representational power and processing efficiency of the knowledge base system into account, very important research topic.

To improve the efficiency of retrieval or update processing, the concept of partial evaluation[16] technique applied to meta-programs is to be introduced in the implementation of a knowledge compiler. A knowledge server and predicate server are provided by the knowledge compiler for retrieval. At update processing, the knowledge base is reorganized using an inductive inference function to improve the efficiency of subsequent retrieval processing. A retrieve control code will be embedded in the knowledge base. The multi-world concept will be used to modularize the knowledge base into layers in order to improve query processing efficiency (Figure 5). Partial compilation or complete compilation can be selected depending upon the application. When partial compilation is used, optimization will be obtained by monitoring the status of intermediate results.

We have begun investigation of these research topics using both the combined model with four layers and the integrated model.

6. Knowledge base machine model

We are designing a further advanced parallel knowledge base machine in which logic processors and memory devices are linked in a high speed network. A brief description of this machine will be given in this section.

6.1 Processor / dedicated hardware / network

The processor is an inference machine, and has a data-stream type dedicated hardware system to distribute simple repeated processing loads of massive amounts of data. The processor performs parallel control of n dedicated hardware systems, generates data streams for several. It also controls the generation sequence.

Functions of the dedicated hardware system can be classified into the following two types:

(1) Limit the amount of data transmission between the primary and secondary storage devices.

(2-1) Perform preprocessing for the processor, reducing processing load.

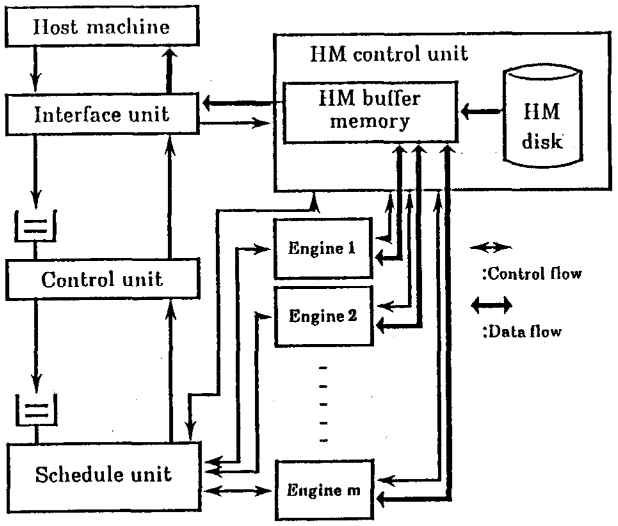


Figure 6. Block diagram of relational database machine

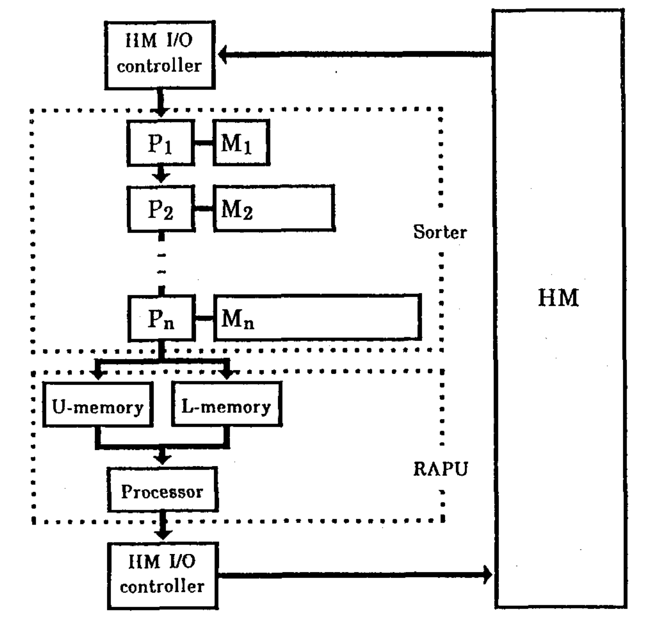
(2-2) Reduce the load of simple repeated processing on a large number of objects with the processor.

Therefore, it is important to develop an algorithm for the dedicated hardware system that processes a data stream in linearly over time as much as possible.

Four relational engines (RE) were developed for DELTA as dedicated hardware systems (Figure 6). A sorting function using pipelined two-way sort/merge algorithm [17], corresponding to (2-1) above, and relational operations such as join and restriction, corresponding to (2-2) above, were implemented in VLSI in the RE. The RE has a sorter, two 64K bytes registers and a relational algebra processing unit (RAPU) , and can process a 3M bytes/s data stream (Figure 7). Engine parallel control method and multiple-engine effect were analyzed by DELTA[18].

We are now developing a unification engine (UE)[19] as dedicated hardware which can process streams of terms (variable length/structure) and unify them. The term generality concept, viewed in the light of unification, is introduced so that this engine can also perform term sorting processing efficiently (Figure 8).

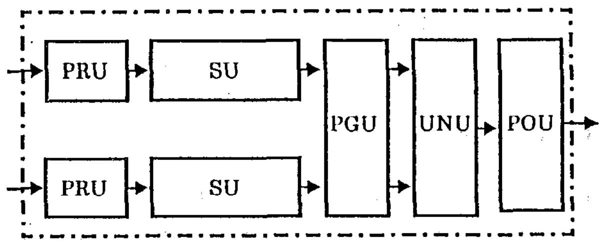
The bus and a completely connected mesh are the major structures of the network. However, multi-port page memory (MPPM)[20][21] linking network and hierarchical memory devices is also included. These components are assembled to develop knowledge base machines step by step. The following machine models are being developed for that purpose.



P_i: ith processor of sorter *M_i: Memory unit of ith processor*

RAPU: Relational algebra processing unit *HM: Hierarchical memory*

Figure 7. Hardware configuration of relational engine



PRU	*:Preprocess unit*	*SU*	*: Sort unit*
PGU	*:Pair generation unit*	*UNU*	*:Unification unit*
POU	*:Postprocess unit*		

Figure 8. Basic structure of unification engine

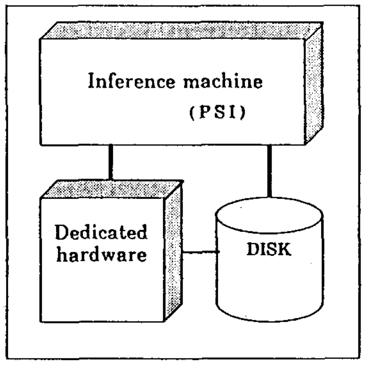


Figure 9. PHI-1 (Combined model with four layers/pilot model)

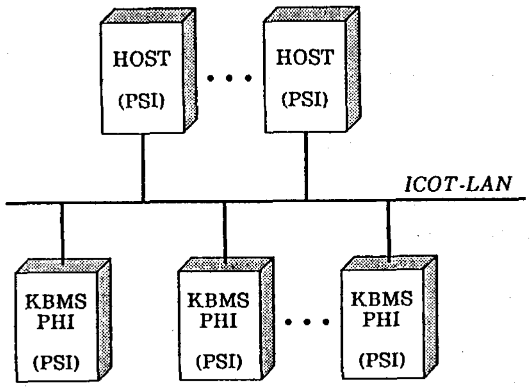


Figure 10. PHI-2 (Combined model with four layers/distributed model)

6.2 Machine models

(1) Pilot model

A pilot model (PHI-1) (Figure 9) is being developed to establish the basic technological elements for building distributed and parallel knowledge base machines. PHI-1 consists of an inference machine (PSI), dedicated hardware, and a secondary storage device. The dedicated hardware system perform associative key word searches of knowledge using superimposed-code of hashing function. The definition of the hashing function is extended to terms.

(2) Distributed model

A distributed model (PHI-2) is a model in which two or more PHI-1's are linked by LAN. Host machines will be two or more PSI's (Figure 10).

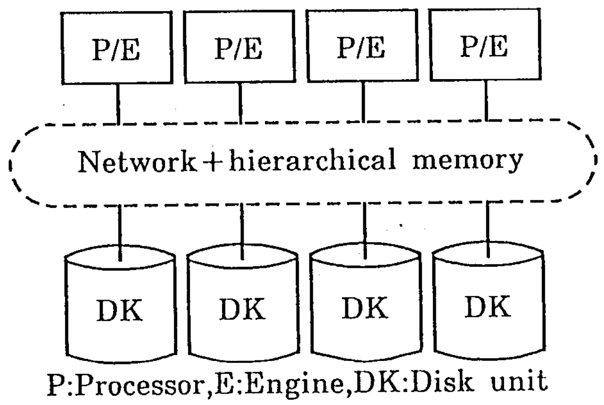

P/E
P/E
P/E
P/E
Network+hierarchical memory
DK
DK
DK
DK
P:Processor,E:Engine,DK:Disk unit

Figure 11. Experimental parallel model

This model will be used to study the relation of computation cost and communication cost, distributed knowledge management, distributed queries, distributed retrieval, and cooperative problem-solving.

(3) Parallel model

This model covers a number of research topics. An experimental hardware system will be developed to analyze basic architectural problems.

The parallel model links several processors, several engines, and secondary memory devices in a network (Figure 11). We are considering a mechanism in which network and hierarchical memory are integrated. The hierarchical memory will be placed between the primal memory within the processor and the secondary memory. The processor will be an inference machine, and the engine will be a unification engine. Hierarchical memory will be used to store temporal resolution for in knowledge base retrieval. It will also be used as shared memory by processors.

This model will be used to study ideal memory architecture for knowledge base machines, multiple memory access control methods, and parallel control of processors and engines. Characteristics of the degree of parallelism and processing volume/granularity will be evaluated against basic operations in knowledge base processing, the order in which they are applied, and the amount of knowledge to be processed. Architecture, parallel resource management, and parallel execution management will be analyzed. Compared to a relational database the degree of parallelism and the amount of processing in knowledge base processing vary more dynamically, since unification is the basic operation in the knowledge base. Also, loads on physical resources such as processor, engine, and memory vary dynamically depending upon the traffic. Therefore, parallel resource management and parallel execution management must be integrated in a coherent and comprehensive management methodology, considering processing division overhead, communication overhead, and schedule overhead.

7. Conclusion

We described the present status of research and development at ICOT on knowledge base systems. The knowledge base system and the inference machine will be linked more closely both physically and logically. ICOT is also working on developments of an intelligent interface program and shell program of expert system on the PSI inference machine, although we could not touch upon these activities in this paper. Based on the technologies obtained and accumulated through these activities, we will develop a prototype of a highly parallel computer for knowledge processing, which is our ultimate aim.

Acknowledgements

The author would like to thank Dr. Fuchi, the director of ICOT, who contributed many useful suggestions for this research and development. The author wishes to thank the members of the KBM working group and the people in industry who participated in joint research programs for their helpful discussions. The author also thanks Drs. Furukawa, Uchida and K. Yokota for their valuable comments on this paper as well as Y. Morita and H. Monoi who helped in preparing the manuscript.

References

[1] K. Fuchi, "Revisiting Original Philosophy of Fifth Generation Computer Systems Project", International Conference on Fifth Generation Computer Systems. Conference Report pp.18-25,1984.

[2] K. Fuchi, K.Furukawa, "The role of Logic Programming in the Fifth Generation Computer Project", International Logic Programming Conference, July 1986.

[3] T. Chikayama, "Unique Feature of ESP", International Conference on Fifth Generation Computer Systems. 1984, ICOT(1984), pp. 292-298.

[4] M. Yokota, A. Yamamoto, K. Taki, H. Nishikawa, S. Uchida, "The Design and Implementation of a Personal Sequential Inference Machine: PSI", New

Generation Computing, Vol.1 No.2, pp. 125-144, 1984.

[5] S. Takagi, T. Yokoi, S. Uchida, T. Kurokawa, T. Hattori, T. Chikayama, K. Sakai, J. Tsuji, "Overall Design of SIMPOS", Proc. Second International Logic Programming Conference, July 1984.

[6] K. Murakami, T. Kakuta, N. Miyazaki, S. Shibayama, H. Yokota, "A Relational Database Machine; First Step to Knowledge Base Machine", Proc. of 10th Symposium on Computer Architecture, June 1983.

[7] K. Ueda, "Guarded Horn Clauses", Logic Programming '85, E.Wada(ed). Lecture Notes in Computer Science 221, Springer-Verlag (1986).

[8] K. Murakami, T. Kakuta, R. Onai, N. Itoh, "Research on Parallel Machine Architecture for Fifth-Generation Computer Systems", June 1985.

[9] A. Goto, S. Uchida, "Current Research Status of PIM: Parallel Inference Machine", ICOT TM-140, November 1985.

[10] H. Yokota, H. Itoh, "A Model and an Architecture for a Relational Knowledge Base", The 13th Annual International Symposium on Computer Architecture, June 1986.

[11] H. Gallaire, J. Minker, and J.-M. Nicolas, Data Base Theory. Vol.1, (1981), Vol.2, (1984), Plenum Press.

[12] H. Gallaire, J. Minker, and J.-M. Nicolas, "Logic and Database: A Deductive Approach", Computing Surveys 16(1984), pp.153-185.

[13] H. Kitakami, S. Kunifuji, T. Miyazaki, and K. Furukawa, "A Methodology for Implementation of Knowledge Acquisition System," Proceedings of the 1984 International Symposium on Logic Programming, pp.131-142, Feb. 1984.

[14] H. Yokota, K. Sakai, H. Itoh, "Deductive Database System on Unit Resolution", The 2nd Data Engineering Computer Conf. pp.228-235, Feb.1986.

[15] N. Miyazaki, H. Itoh, et al. "Compiling Horn Clause Queries in Deductive Database", ICOT TR-183.

[16] A.Takeuchi, K. Furukawa, "Partial Evaluation of Prolog Programs and it's Application to Meta Programming", ICOT TR-126.

[17] S. Todd, "Algorithm and Hardware for Merge Sort Using Multiple Processors", IBM Journal Research and Development, Vol.22, No.5, Sep. 1978.

[18] H.Itoh, C.Sakama, et at. "Parallel Control Techniques for Dedicated Relational Database Engines", ICOT TR-182.

[19] Y. Morita, H. Yokota, K. Nishida, H. Itoh, "Retrieval-by-Unification on a Relational Knowledge Base Model", to appear in Proceedings the 12th International Conference on VLDB, Aug. 1986.

[20] Y. Tanaka, "MPDC: Massive Parallel Architecture for Very Large Databases", Proc. International Conference on 5th Generation Computer Systems, pp.113-137, Nov 1984.

[21] H. Boral, et al, "Implementation of the Database Machine DIRECT" IEEE Trans. Software Eng. vol SE-8. No. 6, Nov. 1982.