TM-0507

Some Topics Related to Expert Systems in the Fifth Generation Computer Systems Project

by
J. Sawamoto and Y. Fujii

May, 1988

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan
(03) 456-3191~5
Telex ICOT J32964

Institute for New Generation Computer Technology

Some Topics Related to Expert Systems in the Fifth Generation Computer Systems Project

Jun Sawamoto and Yuichi Fujii

Fifth Research Laboratory,
Research Center,
Institute for New Generation Computer Technology (ICOT),
4-28, Mita 1-Chome, Minato-ku, Tokyo 108, Japan

Abstract

This paper describes some activities related to expert systems in the Fifth Generation Computer Systems (FGCS) Project. The fifth research laboratory of the ICOT research center is working on the development of experimental expert systems for the demonstration of the outcome of the FGCS Project and the research and development of basic technologies for constructing knowledge-based (expert) systems [ICOT 87]. Together with the collaborating campanies and research institutes, we are building demonstration systems in various application domains in ESP[1] on the PSI[2] machine. The activities for the development of basic technologies include building a prototype expert system tool, PROTON, using currently available technologies on the PSI, and research on next generation expert system tools and knowledge acquisition support systems. This paper focuses on the description of features of PROTON with some critiques and the activities on advanced AI technologies which seem to be the key technologies for the next generation tools.

1. ESP (Extended Self-contained Prolog) is a Prolog extended with an object-oriented mechanism. It is the end user language for the PSI as well as the system description language used for the PSI's operating system, SIMPOS, and its utilities [Chikayama 84].

2. The PSI (Personal Sequential Inference machine) is a Prolog-based machine developed at ICOT in the initial stage of the FGCS Project. The PSI is widely used as one of the software development environments in the project [Taki 84].

1. Introduction

The FGCS Project is now in the last year of its four-year intermediate stage after completing the three-year initial stage. In the fifth research laboratory, we are researching and developing basic software, emphasising user needs and application systems. We are working on the following two general areas.

(1) Research and development of basic technologies for constructing knowledge-based systems on the logic programming framework

(2) Research aimed at demonstrating the outcome of the FGCS Project (Table 1 in the appendix shows some of the experimental expert systems at ICOT. This part is carried out in collabration with companies affiliated with the project. Each theme is assigned some technical issues which should be investigated in the course of system development.)

The PSI is one of the major results of the three-year initial stage of the FGCS Project. PSIs are currently used as one of the major software development environments in the FGCS Project. The development of a prototype expert system development environment (tool) on the PSI was called for, because researchers of application knowledge systems needed a handy environment on the PSI, and because the development of an expert system tool is by itself a good research area for AI topics such as problem solving methodologies, inference mechanisms, knowledge acquisition methods and intelligent user interfaces. PROTON [Sawamoto 87] is intended to serve as the basic environment for this research [Iwashita 86]. PROTON is a second generation expert system tool, like ART [Clayton 85], KEE [KEE 86] [Fikes 85] and KC [KC 85], which provides hybrid knowledge representation schemata. Next generation tools are tools which come after second generation tools.

The following technologies are discussed as the advanced AI technologies which are researched to develop next generation tools at ICOT.

(1) Constraint logic programming,
(2) Hypothetical reasoning,
(3) Distributed cooperative problem solving,
(4) Utilization of deep knowledge and qualitative reasoning
(5) Knowledge acquisition support system

2. PROTON on the PSI Machine

2.1 Features

The knowledge representation constructs most appropriate for the knowledge in the following three categories are provided in PROTON.

(1) Static knowledge of the problem domain
(2) Knowledge for problem solving heuristics
(3) Knowledge for meta-level strategies

Static knowledge is the knowledge required for modelling problem domains. A domain can be described in terms of its structural objects, attributes and values of objects, and relations among objects. The hierarchical and relational representation of frames is most suited to this type of knowledge. Knowledge for problem solving heuristics is the expert's domain knowledge used for problem solving. Production rules are most appropriate for this type of knowledge. Production rules search objects or relations of the problem domain for their premises and modify objects or relations in the action part. Knowledge for meta-

level strategies is the knowledge for controlling the usage of the domain knowledge. Meta-rules are considered for providing this type of knowledge formalism.

ESP provides both logic and object-oriented programming. PROTON facilitates a frame-based and rule-based programming environment on the ESP programming environment. Figure 1 is an overview of the PROTON configuration. As an expert system tool on the PSI, PROTON has the following characteristics.

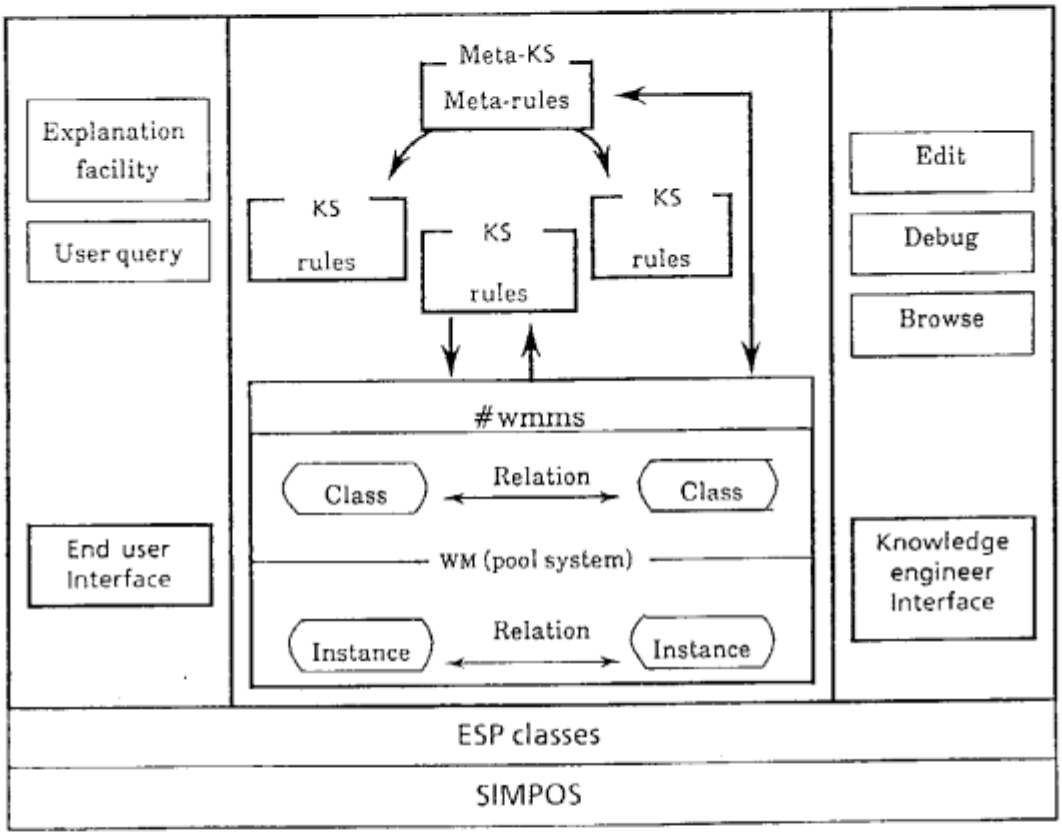

Figure 1 Overview of the PROTON configuration

(1) Three knowledge representation constructs, production rules, frames, and meta-rules, are integrated under ESP's object-oriented programming. Therefore, each construct can be used separately or in combination according to the application system's configuration. Thus, PROTON can provide a flexible and powerful knowledge representation environment on the PSI.

(2) Rules (including meta-rules) are modularized into multiple knowledge sources (KSs). KSs are implemented as class objects of ESP. KSs can be viewed as a special case of ESP coding which allows description of the rule format.

(3) Frames are also implemented as class objects of ESP. Class objects and their instances (instance objects) are stored in the *pool system* of the PSI (working

memory: WM) and accessed by sending messages to the ESP class, working memory management system (#wmms), in PROTON.

(4) PROTON provides two kinds of user interfaces, one for the knowledge engineer (KE) and the other for the end user (EU). For KEs, PROTON provides an editor, a rule and frame debugger, and a browser. They are implemented using the PSI's multi-window and menu system. For EUs, PROTON provides primitives for generating end user interfaces, such as a graphic explanation facility and a user query facility for the developed expert systems.

2.2 Critiques of Current Tools

PROTON is currently being applied to sample problems, and evaluations are in progress. Some of the demonstration expert systems at ICOT (Table 1) are being developed on PROTON. The following are some critical comments on PROTON and second generation tools in general.

- The multiple knowledge source and meta-rule mechanism of PROTON is generally useful, although some higher-level control mechanisms such as intelligent backtracking should be incorporated.
- The hybrid knowledge representation schema looks powerful. However , it takes very careful control to coordinate various kinds of knowledge in different representation schemata.
- The use of the underlying language, i.e., ESP in PROTON, is heavy. It is often an advantage for flexible programming. However, it sometimes causes difficulty in knowledge base maintenance. This phenomenon implies that there is knowledge which cannot be properly covered by the current framework of tools.
- Higher-level problem solving techniques are still out of the scope of current tools. Users somehow have to implement them within the frameworks of the given tools.
- There is no methodology or guideline for building expert systems using current tools. It is still very difficult to build expert systems for domain experts by themselves.

3. Research on Next Generation Tools

For the expert system tool or expert system development environment, three levels of support hierarchy can be identified, the programming language level, general problem solving level and generic tasks level [Chandrasekaran 86]. PROTON, like other general expert system tools, provides the lowest level of support, that is, the programming language level.

With our experience of developing and using PROTON and developing experimental expert systems in various domains, we have come to the same conclusions as have been recently recognized by researchers and builders of expert systems. They are:

(1) Need for the development of more advanced, higher-level inference technologies, and

(2) Need for the establishment of a methodology for building expert systems and the development of powerful knowledge acquisition support systems.

Taking a broad view of trends in R&D on knowledge-based systems, we recognize that the main thrust of research is shifting, with less emphasis on analytic problems (e.g., diagnostics and interpretation) and more on synthetic problems (e.g., design and planning). Taking these trends into account, the following fields have been taken up as the main technologies for the research of the next generation tools.

(1) Constraint logic programming
(2) Hypothetical reasoning
(3) Distributed cooperative problem solving
(4) Utilization of deep knowledge and qualitative reasoning
(5) Knowledge acquisition support system

(4) is researched mainly in the context of knowledge acquisition support systems.

3.1 Constraint Logic Programming

In considering design tasks, many problems can be regarded as constraint satisfaction problems. However, very little expert system tools provide an environment supporting efficient description and handling of constraints.

Constraint logic programming (CLP) [Jaffar 87] is an attempt to increase the descriptive power of logic programming employing constraint solving as a generalization of its unification mechanism. One of the advantages of constraint logic programming is that it allows the declarative description of problems in terms of constraints. At ICOT, three different CLP languages, CAL, CRL, and CIL, have been proposed and developed.

(1) CAL [Sakai 88]

CAL is a CLP language which can handle constraints in the form of non-linear polynomial equations, while almost all the other CLP languages proposed so far have concentrated only on linear equations and inequations.

The following is an example of a CAL program which computes the area of a triangle from its height and baseline length.

```
area(Height,BaseLine,Area) :- Height*BaseLine=2*Area.
rightangle(A,B,C) :- A^2+B^2=C^2.
triangle(A,B,C,Area) :- C=CA+CB,rightangle(CA,H,A),rightangle(CB,H,B),area(H,C,Area).
```

The third clause means that every triangle can be divided into two right-angled triangles. When the goal, *triangle(3,4,5,S)*, is given, the program answers *S^2* is *36*. The power of CAL compared to ordinary Prolog is demonstrated in this simple example. For example, the assertion, *5=CA+CB*, in the process of the execution of the above goal, *triangle(3,4,5,S)*, simply fails in Prolog. CAL takes this assertion as an atomic constraint and invokes the constraint solver.

CAL gives powerful means for describing and handling constraints in polynomial equations. It is required to extend to the constraints of inequalities, Boolean equations, etc. CAL is now running on the PSI.

(2) CRL [Yokota 88]

CRL is the knowledge representation language for the database system, KAPPA, which is based on a nested (non-first-normal-form) relational model [Yokota 87]. When we consider applications with structured data such as CAD/CAM, office automation and document processing, the first normal form of a relational model is not appropriate enough. One of the solutions is a nested relational model. CRL is intended to be an extended logic programming language for a deductive database based on nested relations.

CRL introduced a nested attribute-value paired term (NAV) as its data structure corresponding to the nested record in the nested relational model.

*Example of NAV : person/{name/john*hobby/{skiing,tennis,baseball}}.*

CRL's unification is extended over the set of attribute values of NAVs. Unification of NAVs corresponds to the intersection of sets. The following is an example of a CRL program.

```
parent/{jack,betty}*child/{john,cathy}.
parent/{jack,nancy}*child/{mary}.
parent/{john,kate}*child*/{bob}.
ancestor/X*descendant/Y <- parent/X*child/Y.
ancestor/X*descendant/Y <- parent/X*child/Z,ancestor/Z*descendant/Y.
```

Consider the following goals (queries) for the above database,

```
<- parent/X*child/{john}.
<- parent/X*child/{john},parent/X*child/{mary}.
<- ancestor/X*descendant/bob.
```

The first query obtains *X = {jack}, {betty} and {jack,betty}*. The second one obtains *X = {jack}*. The third one obtains *X = a non-empty element of a power set of {john,kate,jack,betty}*.

We intend to use CRL and KAPPA for the database of design expert systems such as the mechanical design system.

(3) CIL (Complex Indeterminate Language) [Mukai 85]

CIL was first developed as a system description language for natural language processing such as a discourse understanding system. CIL extends the usual Prolog terms with a structure called a partially specified term (PST). The unification mechanism is extended over PSTs of CIL . A PST is a set of attribute-value pairs, $\{a_1/b_1, ...,a_n/b_n\}$, where each a_i is a first order term which should be ground and b_i is any term including the PST itself. The extended unification over PSTs works as follows.

$$unify(\{a/1,b/2\},\{b/X,c/3\}) = \{a/1,b/2,c/3\}$$

X is unified to 2 as a result.

By using PSTs, a structure of a semantic network with cycles can be represented. CIL is used in the intelligent secretary system (Table 1) to process natural language interface.

We briefly described CLP languages at ICOT. They are powerful languages for description and problem solving within the framework of logic programming. However, from an application point of view, they are still at the programming language level. We are trying to build a mechanism which can provide higher-level constraint solving functions such as failure recovery, constraint relaxation, and preservation of dependency relations on those CLP languages.

3.2 Hypothetical Reasoning

Hypothetical reasoning is an important framework in problem solving when there is alternative knowledge, tentative knowledge, knowledge with exception incomplete knowledge or abductive knowledge. Figure 2 shows a general logical framework for hypothetical reasoning. If a contradiction occurs in the reasoning process, the system removes the original hypotheses and selects other hypotheses (e.g., TMS [Doyle 79], ATMS [de Kleer 86] or Default logic [Reiter 80]). On the other hand, abductive reasoning obtains hypotheses which explain a set of observations (e.g., Theorist [Poole 87] or diagnostic reasoning).
There are a few expert systems and tools which incorporate a mechanism based on a TMS or ATMS [Clayton 85] [KEE 86]. However, simple incorporation of a TMS or ATMS leads to the problem of combinatorial explosion. Therefore, it is very important to consider how to provide a control mechanism for efficient selection of hypotheses (contexts) which are kept under TMSs. APRICOT in ICOT [Inoue 88a] is now being designed considering various search mechanisms as a key technology for controlling multiple contexts of TMSs. APRICOT provides not only theTMS mechanism but also hypothesis generation (including abduction) or enumeration based on domain-dependent knowledge and hypothesis selection.

3.2.1 APRICOT (assumption-based problem solver in ICOT)

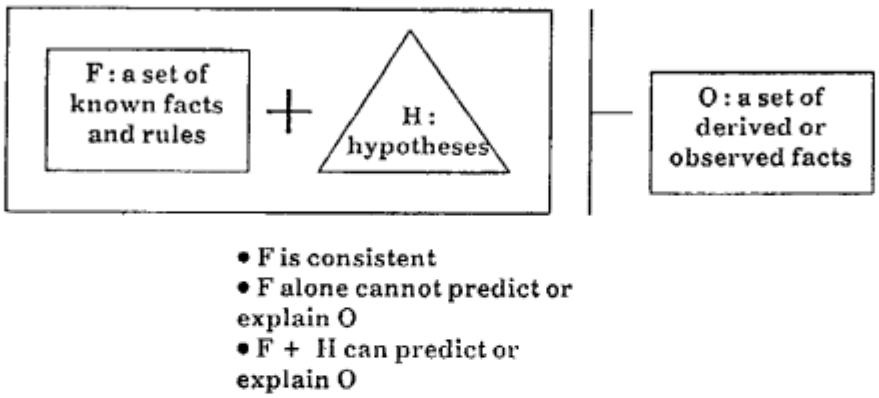

Figure 2 A logical framework for hypothetical reasoning

APRICOT provides a basic framework for using hypothetical reasoning in problem solving, and manages multiple contexts on the basis of a logic-based TMS. APRICOT consists of a problem solver, knowledge base and modules for the generation, selection and verification of hypotheses. (see Figure 3.)

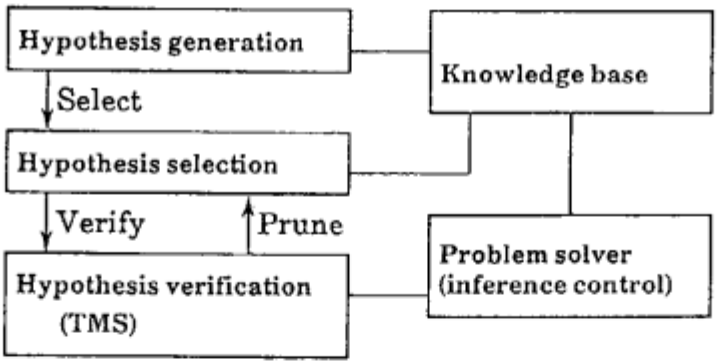

Figure 3 Block diagram of APRICOT

The problem solver performs various kinds of inference depending on the given problem, and the TMS acts as a domain-independent management mechanism of multiple contexts.

Hypothesis generation determines the hypothesis space from the knowledge base which provides domain-dependent knowledge such as the domain's deep knowledge (e.g., description of structure, functionality or laws), constraints and heuristic rules. Hypothesis selection selects candidates for acceptable hypotheses using information provided by the knowledge base and information on contradictions from hypothesis verification. Hypothesis verification keeps

multiple consistent contexts for current problem solving. In this framework, every hypothesis will not be expanded concurrently, as is the case with ATMS. And the incremental addition of hypotheses along the contexts is possible.

There are some general context search strategies which act as interfaces between the TMS and the problem solver, such as dependency-directed search (DDS), because, when there are competing multiple contexts, the selection of a context should depend on the knowledge of the problem domain. APRICOT can incorporate general AND/OR tree search methods as its inference control, which selects preferable contexts in the process of hypothetical reasoning [Inoue 88b].

3.3 Distributed Cooperative Problem Solving

Distributed AI systems have been intensively researched recently. They have advantages over conventional systems.

(1) Tolerance
Each problem solver only need to have knowledge relevant to its own problem area. The total solution is obtained by coordinating solutions of subproblems which are distributed to solvers. In this way, distributed AI systems can serve as powerful tools for solving problems with uncertain data and/or knowledge.
(2) Efficiency
A high processing speed is expected to be achieved because there are many chances to introduce parallelism between problem solvers.

At ICOT, we started research on distributed cooperative problem solving one year ago. Our approach is first to investigate various instances of distributed cooperative problem solving and then formalise problem solving technologies based on investigation. Currently, we are developing experimental expert systems such as VLSI logic design, delivery planning and portfolio analysis to investigate those instances in collaboration with companies affiliated with the project. (see Table 1.)

3.4 Knowledge Acquisition Support System

Knowledge acquisition is the most important process in building a knowledge-based system. The knowledge acquisition bottleneck (Feigenbaum's bottleneck) is a major problem we have to cope with in the research of next generation tools. Two observations concerning this problem have motivated our research in ICOT. One is that rules and frames only are not an appropriate level of representation for problem solving knowledge of experts (control saturation problems). The other is the lack of systematic support for knowledge engineering. We identify the following four major phases in the work of the knowledge engineer.

(1) Problem analysis phase: This phase includes selection of a problem, evaluation of current AI technology for the problem and identification of the knowledge sources.

(2) Expert model building phase: This phase includes identification of expert models or tasks (how the expert uses his knowledge or problem solving techniques and inference strategies) and identification of user models (how the user uses the system).
(3) Expert model instantiation phase: This phase includes extracting knowledge from the knowledge sources according to expert models and transforming knowledge so that it can be executed efficiently.
(4) Knowledge base management phase: This phase includes refinement and consistency checking.

Currently, we are mainly researching knowledge acquisition methods of the expert model building phase and the expert model instantiation phase.

3.4.1 Expert Model

A unique expert system building approach is the idea of generic tasks. Generic tasks support some expert task primitives. Its primitive tasks are useful to define expert knowledge in the knowledge acquisition stage. Six generic tasks have been found, including hierarchical classification, hypothetical matching or assesment, knowledge-directed information passing, abductive assembly, and the ongoing research is searching for other generic tasks.

Effective knowledge acquisition needs an arranged knowledge structure, which is separated into basic knowledge and meta-knowledge. The EXPERT MODEL [Taki 87] is being designed in ICOT to represent basic knowledge and meta-knowledge for the knowledge acquisition stage. The core concept of the EXPERT MODEL is the idea of generic operations. The types of generic operations are SELECTION, CLASSIFICATION, SORT, COMBINATION, TRANSFER, INPUT and OUTPUT. In the EXPERT MODEL instantiation stage, these generic operations obtain semantic information.

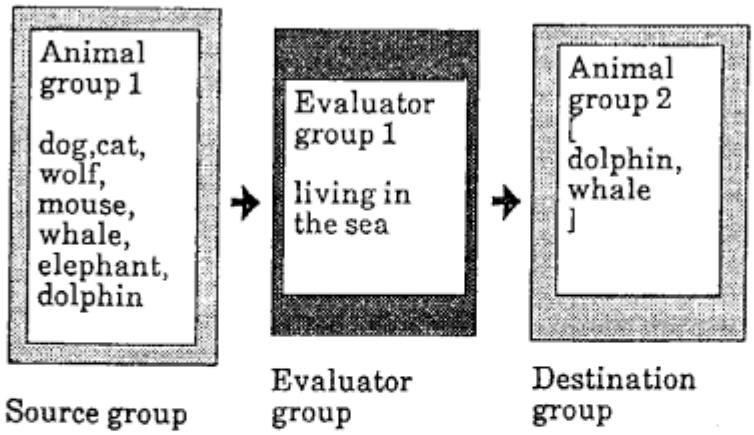

Figure 4 Example of SELECTION operation

For example, one SELECTION with test-device selection criteria information is a test-device selection operation. One generic operation consists of operation attributes, evaluators and element groups. An evaluator is the action description of a generic operation. The evaluator of SELECTION evaluates criteria of the selection (see Figure 4). The evaluator of SORT compares attributes of elements in the element group and arranges them in order of the comparison result.

The generic operations are constructed to be part of a knowledge base through modelling and instantiation. There is meta-knowledge to manage generic operations. Object-oriented architecture is adopted as this knowledge representation, because the message passing model as object-oriented architecture is suitable for top level human knowledge representation. In the EXPERT MODEL, the generic operation is an object. We are now building an experimental knowledge acquisition system for diagnostic tasks, EPSILON/EM, based on the idea of EXPERT MODEL.

3.4.2 Integrated View of Knowledge Acquisition

Several technologies have been proposed for various aspects of knowledge acquisition. They are:

(1) Interview from domain experts
(2) Knowledge compilation (from deep to shallow knowledge)
(3) Learning (e.g., rule induction and EBL)
(4) Knowledge base management and refinement

These technologies are being researched fairly independently. It is necessary to integrate them into a broader methodology for building expert systems [Mizoguchi 88]. We are working to build up the image of such a knowledge acquisition system. The main idea is the concept of building blocks which are as high level as generic tasks, generic operations or problem solving primitives such as generate & test. These technologies can be integrated around the concept of building blocks. For example, the interview system is used for analising the expert's tasks and identifying appropriate building blocks and for acquiring domain knowledge according to the identified building blocks. Knowledge compilation technology can be used for automatically generating a shallow knowledge base referring to the obtained framework of building blocks when a deep knowledge base is available [Yamaguchi 87]. A shallow knowledge base also can be generated by rule induction from examples and by EBL from the domain theory and examples.

4. Conclusion

This paper described some activities related to expert systems in the FGCS Project. The features of PROTON and some critiques of second generation expert system tools were discussed. Based on our experience of PROTON and development of experimental expert systems in various domains, we are

researching advanced AI technologies for next generation expert system tools. Facilitating higher levels of support on the logic programming framework is the focus of our current research. At ICOT, we are working on higher-level problem solving mechanisms, such as constraint logic programming, hypothetical reasoning, distributed cooperative reasoning and an integrated knowledge acquisition support system. We haven't discussed the interrelation of these technologies in this paper. However, they are interrelated each other very closely. For example, hypothetical reasoning can be executed efficiently by properly incorporating the handling of constraints. And, problem solvers of the distributed cooperative problem solving are, in a sense, solving a large constraint problem efficiently by dividing the problem into subproblems where each solver reasons truth maintenance functions. We are now investigating the interrelationship of them, and working toward more integrated problem solving mechanisms.

5. Acknowledgements

We would like to thank the researchers of the fifth research laboratory of ICOT for their help and valuable discussions in the preparation of this paper. Our thanks must also go to Dr. K. Fuchi, Director of the ICOT Research Center, who gave us the opportunity to conduct our research in the Fifth Generation Computer Systems Project.

References

[Chandrasekaran 86] B. Chandrasekaran, Generic Tasks in Knowledge-based Reasoning: High-level Building Blocks for Expert System Design, IEEE Expert, 23/30, 1986

[Chikayama 84] T. Chikayama, Unique Features of ESP, in Proceedings of FGCS'84, ICOT, 1984

[Clayton 85] B. Clayton, ART Programming Tutorial, Inference Corporation, 1985

[Doyle 79] J. Doyle, A Truth Maintenance System, Artificial Intelligence, 12 (1979) 231-272

[Fikes 85] R. Fikes and T. Kehler, The Role of Frame-based Representation in Reasoning, Communication of the ACM, 28, 9, September 1985

[Hayes-Roth 83] F. Hayes-Roth, D.A. Waterman and D.B. Lenat, Building Expert Systems, Addison-Wesley Publishing, 1983

[ICOT 87] Research Activities at ICOT: Fifth Research Laboratory, ICOT Journal, No.18, December 1987

[Inoue 88a] K. Inoue, APRICOT - Problem Solving with Hypothetical Reasoning, ICOT, TR-363, 1988 (in Japanese)

[Inoue 88b] K. Inoue, Pruning Search Trees in Assumption-based Reasoning, Proc. Avignon'88: The 8th International Workshop on Expert Systems & their Applications, 1988

[Iwashita 86] Y. Iwashita and J. Sawamoto, Development of Expert Systems in the Fifth Generation Computer Systems Project, 2nd Int'l Expert Systems Conference, London, September 1986

[Jaffar 87] J. Jaffar and J-L, Lassez, Constraint Logic Programming, Proc. 4th IEEE Symposium on Logic Programming
[KC 85] KC 3.0 Reference Manuual, CGI, 1985
[KEE 86] KEE Reference Manual, IntelliCorp, 1986
[de Kleer 86] J. de Kleer, An Assumption-based TMS, Artificial Intelligence, 28, 2, 1986
[Mizoguchi 88] R. Mizoguchi et al, A Methodology for Building Expert Systems, SIG-KBS-8801-2(4/26), JSAI, 1988 (in Japanese)
[Mukai 85] K, Mukai, Unification over Complex Indeterminates in Prolog, ICOT TR-113, 1985
[Poole 87] D. Poole, R. Goebel and R. Aleliunas, Theorist: A logical Reasoning System for Defaults and Diagnosis, The Knowledge Frontier, Cercone & McCalla (eds.), Springer-Verlag, 1987
[Reiter 80] R. Reiter, A Logic for Default Reasoning, Artificial Intelligence, 13 (1980) 81-132
[Sakai 88] K. Sakai and A. Aiba, CAL: A Theoretical Background of Constraint Logic Programming and Its Applications, Workshop on Constraints and Languages, Providence, Rhode Island, USA, April 1988
[Sawamoto 87] J. Sawamoto et al, PROTON: An Expert System Tool on the PSI, in Proceedings of the Australian Joint AI Conference, Sydney, November 1987
[Taki 84] K. Taki, Hardware Design and Implementation of the Personal Sequential Inference Machine (PSI), in Proceedings of FGCS'84, ICOT, 1984
[Taki 87] H. Taki, K, Tsubaki and Y. Iwashita, EXPERT MODEL for Knowledge Acquisition, in Proc. of 3rd Annual Expert Systems in Government Conference, October 1987
[Yamaguchi 87] T. Yamaguchi et al, Design of Knowledge Compiler Based on Deep Knowledge, Journal of Japanese Society for Artificial Intelligence (JSAI), 2, 3, 1987 (in Japanese)
[Yokota 87] K. Yokota, Knowledge Base Management System KAPPA, 5th Symposium on FGCS, June 1987 (in Japanese)
[Yokota 88] K. Yokota, Deductive Approach for Nested Relations, ICOT TR, 1988 (to appear)

Appendix

Table 1 Some Experimental Expert Systems at ICOT

Theme	Goal	Technical issues
VLSI logic design	· Designs CMOS circuits from algorithms · Acquires knowledge from designers·	·Cooperative reasoning by multiple design agents · Knowledge base management (ATMS) · Stimulates designers
Mechanical design	· Study of intelligent CAD · Expert system for mechanical design	· Constraint solving · Attribute modelling · Knowledge compilation
CAD for equipment layout in computer rooms	· Layout of computer room equipment	· Verification of object-oriented features of ESP · Division into subproblems and parallel problem solving
Portfolio analysis	· Analysis and investment planning	· Distributed cooperative problem solving · Parallel proccesing in GHC
Intelligent secretary	· Schedules business trips · Schedules meetings	· Natural language interface · Knowledge acquisition support
Delivery planning	· Plans delivery routing and resource allocation	· Distributed cooperative problem solving · Constraint solving
Diagnosis system for electronic switching system	· Diagnoses undetectable faults using an internal test program	· Hypothetical reasoning (dependency-directed backtracking) · Deep knowledge