PRESENT STATUS AND PLANS FOR RESEARCH AND DEVELOPMENT

Takashi Kurozumi

Deputy Director, Research Center
Institute for New Generation Computer Technology
4-28, Mita 1-chome, Minato-ku, Tokyo 108, Japan

ABSTRACT

The Fifth Generation Computer Systems Project was launched in 1982 as part of the information-related policy of the Ministry of International Trade and Industry (MITI). Its purpose is to research and develop a new computer technology that will provide the basis for the creation of knowledge information processing systems (KIPS) needed in the 1990s.

ICOT has been entrusted by MITI to promote this national project in cooperation with manufacturers, national and public research organizations, and universities.

The project has been proceeding according to a ten-year plan, which is divided into an initial three-year stage, an intermediate four-year stage and a final three-year stage. This year, 1988, is the last year of the intermediate stage.

This report describes the R&D status of the intermediate stage and the R&D plan for the final stage.

1 OUTLINE OF THE FIFTH GENERATION COMPUTER SYSTEMS PROJECT

The R&D programs of this project aim at creating prototypes of fifth generation computer systems. The basic framework of fifth generation computers, an outline of the general R&D plan, and the organization to promote this project are described below.

1.1 Basic Framework of Fifth Generation Computers

1.1.1 Concept of Fifth Generation Computers

Conventional computers have been classified into generations according to their constituent hardware elements: vacuum tubes, transistors, IC, LSI and VLSI. But they are all based on the same Von Neumann architecture, which is characterized by sequential processing and stored-program schemes.

In present-day computers, the characteristics of the architecture determines the type of machine language and software based on machine language is procedural. Present-day computers are limited because there is an enormous gap between the way that they work and the way that human beings think, knowledge-based inference. Computers must follow pre-defined procedures; they cannot do processing that depends on the circumstances.

There are basic needs that future computers must satisfy. They should be intelligent, easy to use and readily available; their software must be productive. Although conventional computers have the architectural limitation explained previously, there are technical seeds such as research into artificial intelligent technology, architecture technology and software engineering technology. These technologies have been developed independently.

The objective of the Fifth Generation Computer Systems Project is to overcome the technical restrictions of conventional computers and develop innovative computers capable of intelligent information processing. Such machines will be essential in the information-oriented society of the 1990s.

The concept of the fifth generation computer system stemmed from the idea that meeting future needs would be possible by selecting the existing R&D results that can be used to further development and by combining these results in a completely new framework.

The new framework can be built by specifying a predicate logic language as a new machine language, by creating a hardware system that performs highly parallel inference processing based on the new language, and by creating a software system that performs a new type of processing, a combination of the basic inference processings provided by the hardware system.

1.1.2 Basic Structure of Fifth Generation Computer Systems

A fundamental characteristic of intelligent activity is inference that uses every piece of stored knowledge, whether it is conscious or unconscious. Inference based on predicate logic is a procedure to extract unknown information using existing knowledge.

In fifth generation computers, hardware and software are based on a programming method called logic programming in which programs are described in the form of a logic and executed as inference. The predicate logic languages assigned to do this are called the kernel language.

Based on the findings of previous artificial intelligence research, we estimate that fifth generation computers will require an inference speed 1000 times greater than conventional computers. The high-level integration provided by advanced VLSIs enable us to make a reasonably compact and inexpensive computer with more than a thousand processors working in parallel. In this project, we aim at an inference execution speed of 100M LIPS to 1G LIPS, using prototype hardware consisting of one thousand processing elements.

In the logic programming framework, a knowledge base for inference will also be represented in a form based on predicate logic. A relational expression in a current relational database can correspond to a predicate logic form as its extended form. For the knowledge base function, we will start working from current relational database techniques, and proceed to processing knowledge data that is represented in a variety of ways in the logic programming framework.

We think that fifth generation computers must have a basic software system with the following basic functions for the knowledge information processing system. The functions needed in the future include an intelligent interaction function and an inference function that uses knowledge bases.

(1) Problem-solving and inference function: A function to perform meta-level inference such as inductive inference, used to control hardware effectively and solve given problems.

(2) Knowledge base management function: A function to acquire, store and use various types of knowledge needed in the course of inference. It has advanced database management capability, a knowledge acquisition capability that collects knowledge by judging whether it is meaningful, and the ability to retrieve and use knowledge effectively.

(3) Intelligent interface function: A function to make computers easy to use, enabling humans and computers to communicate with each other in a flexible and natural way through natural language.

(4) Intelligent programming function: A function to lighten the user's workload in the processes from writing programs to maintenance. It supports program development, converts given problems to more efficient programs, and verifies the accuracy of the programs.

Our aim is to realize the above four basic functions with fifth generation computers. Although the interface between the basic software system and hardware will be implemented in the kernel language, user languages and other languages will be defined as high-layer languages that have a modularization function and various types of knowledge representation functions.

1.2 Research and Development Plan of the Fifth Generation Computer Systems Project

1.2.1 Initiation of the Project

The history of fifth generation computers began when a survey committee was founded by MITI in 1979. In that year, the survey was undertaken on the prospects of fifth generation computers. In 1980, the survey focused on the goals and subjects of R&D. During these two years, the vision of a fifth generation computer for large-scale knowledge information processing took clear shape.

In 1981, MITI established an R&D plan for fifth generation computer systems, which was presented at an international conference held by MITI in October 1981 (FGCS '81). In 1982, ICOT (Institute for New Generation Computer Technology) was founded as the central organization for promoting R&D, and it began R&D work on fifth generation computers, under the auspices of MITI.

1.2.2 R&D Stages

This is a national project that spans 10 years. It is divided into three stages: an initial three-year stage, an intermediate four-year stage, and a final three-year stage.

(1) Initial stage (1982 to 1984)

R&D in the initial stage was aimed at developing the basic technology required for fifth generation computers. Within the framework of this project, the R&D results concerning knowledge information processing were analyzed and selected results were restructured to achieve the goals of the initial stage.

The specific subjects of R&D included an inference subsystem, knowledge base subsystem, basic software system and pilot models for software development. Goals were specified independently for each subject.

To sum up the results of R&D in the initial stage, we evaluated the basic technology needed to develop fifth generation computers by reviewing and testing various experimental systems. We became convinced that the basic framework described in section 1.1 was not mere hypothesis, but was viable and effective. The major results of each R&D subject are as follows.

(a) Inference subsystem: We experimentally reviewed and evaluated various inference methods, including data flow and reduction, by implementing software simulators and hardware simulators.

(b) Knowledge base subsystem: We reviewed and evaluated the relational database scheme as the basis of the knowledge base function by making an experimental parallel relational database machine (Delta).

(c) Basic software system: We proposed GHC as the parallel logic programming language and verified the effectiveness of the logic programming method by implementing experimental software systems such as an experimental relational database management system (KAISER) and a discourse understanding experimental system (DUALS V.0).

(d) Pilot models for software development: We showed that systems based on logic languages were viable and could be effective by developing sequential inference machines (PSI, CHI), sequential logic programming languages (KLO, ESP), and a sequential inference machine operating and programming system (SIMPOS), although all of them were sequential.

(2) Intermediate stage (1985 to 1988)

The objective of the intermediate stage is to create small- to medium-scale subsystems that will provide a basis for the fifth generation computers, while considering development of the individual research subjects and the integration of the subsystems in the final stage.

The focus of the first half of this stage was on the specifications of models, algorithms and the basic architecture on which software and hardware were to be built, based on the results of the initial stage. In the second half of this stage, experimental subsystems were implemented and evaluated using the specifications.

R&D activities of each subsystem are performed in the following way.

(a) Basic software: Research on the use of parallel logic languages as the kernel language is conducted, and experimental software is implemented and basic technology is developed for each of the following functions ; problem-solving and inference, knowledge base management, intelligent interface, and intelligent programming.

(b) Hardware system: Part of the system (Inference/knowledge base subsystem), whose scale is 1/10 that of the prototype, has been developed, and its functions verified.

(c) Development support system: An infrastructure has been completed by developing a parallel software environment and equipping it with a network system.

(3) Final stage (1989 to 1991)

In the final stage, the results of R&D activities in hardware and basic software in the initial and intermediate stages will be integrated, and a prototype system of a fifth generation computer will be implemented. The prototype system will demonstrate that the framework and functions described in section 1.1 can be achieved in a system in which software and hardware are integrated, and that the system is fit for knowledge information processing.

1.3 Organization of the Research and Development of the Fifth Generation Computer Systems Project

The Fifth Generation Computer Systems Project is being carried out by ICOT, under the auspices of MITI. The ICOT organization consists of a general affairs office and a research center. In the initial stage, the research center consisted of a research planning department and three laboratories. In the intermediate stage, it has a research planning department and five laboratories. The responsibilities of each laboratory sometimes change in the middle of a stage, depending on the progress that is made. The laboratories' subjects in 1988 are as follows.

(1) First Research Laboratory: The kernel language (extended version), problem-solving and inference software (including meta-level inference), and intelligent programming software for the basic software.

(2) Second Research Laboratory: Intelligent interface software for the basic software.

(3) Third Research Laboratory: Knowledge base subsystem

(4) Fourth Research Laboratory: Inference subsystem, which includes the first version of the kernel language and parallel inference machine OS (PIMOS); development support system; and knowledge base management software (knowledge base management basic software) for the basic software.

(5) Fifth Research Laboratory: Knowledge base management software (acquisition and use of knowledge base) and experimental demonstration software for the basic software.

The research staff at ICOT is made up of researchers on loan from the Electro-technical Laboratory, Mechanical Engineering Laboratory, NTT, KDD, computer manufacturers and others.

There were 50 staff at the beginning of the intermediate stage, and their number has increased every year. At the beginning of 1988, there were about 90, and soon there will be about 100.

This project has been executed in an R&D organization which formed with the idea that the minds of the first-class researchers in related fields in Japan should be brought together.

(1) MITI has set up an advisory committee to provide overall guidance concerning plan and R&D status of this project. The chairman is professor H. Aiso (Keio University), and members are authorities on this area from universities, research institutes and companies.

(2) The researchers at ICOT conduct the core R&D activities, and ICOT entrusts other R&D work that needs experimental manufacturing and development to manufacturers. We are promoting the project as a single structure.

(3) Experts from universities and institutes have been participating in the Project Promotion Committee (PPC) and Working Groups (WGs). PPC supplies us with general advice about the project. WGs facilitate the exchange of information about each research subject. In 1988, we have 12 WGs that are organized according to the status of the R&D.

We consider it important that ICOT researchers, other researchers in Japan, and scientists abroad stimulate each other, present their results, and exchange information. That is why ICOT is actively promoting the following research meetings and information exchange activities with foreign research institutes.

(1) ICOT researchers make presentations at international conferences. We publish the ICOT technical reports and technical memorandums. We also exchange papers with foreign research institutes.

(2) We welcome researchers from other countries as visitors and our researchers visit research institutes and universities abroad to exchange information.

(3) Every year we invite several renowned researchers from abroad for a short period.

(4) Based on the memorandums with NSF in the United States and INRIA in France, we receive researchers from both countries for a long period.

(5) To disseminate the result of the R&D activities, we sponsor symposiums and logic programming conferences every year, as well as this international conference ('81, '84, '88). In addition, we have held a Japan-Sweden workshop four times, a Japan-France AI symposium twice, and a Japan-U.S.A. AI symposium once.

All the R&D expenses for this project are covered by the national budget, and the amount is determined each year according to the government's budgeting system. The budget was 8.3 billion yen for the three-year initial stage and about 21.5 billion yen for the four-year intermediate stage (4.7 billion for '85, 5.5 billion for '86, 5.6 billion for '87, 5.7 billion for '88).

The results of the R&D of this project are reported to the government each year. Intellectual property such as patents are administered by the Agency of Industrial Science and Technology, and any corporate entity that desires to use one of them can obtain permission by paying a fee. PSI, SIMPOS, and other technologies that have already been developed as development support systems are permitted by the government to be used commercially. The spread of these technologies is the start of the distribution of fifth generation computer technology.

2 RESEARCH AND DEVELOPMENT STATUS OF THE INTERMEDIATE STAGE

The objective of the intermediate stage is the development of subsystems. The major R&D subjects are the basic software system, hardware systems, and development support systems. The basic software system further breaks down into the 5G kernel languages, problem-solving and inference software, knowledge base management software, intelligent interface software, and intelligent programming software. The hardware systems are divided into the inference subsystem and the knowledge base subsystem. Three and a half years of the intermediate stage have passed, and as a whole it is proceeding smoothly, as planned, toward the goal.

2.1 Basic Software System

2.1.1 5G Kernel Languages

The 5G Kernel languages play an important role in bridging hardware research and software research. There are two types of 5G kernel languages, sequential processing logic language (kernel language version 0 (KL0)) and parallel processing logic language (kernel language version 1 (KL1)). The goals of the intermediate stage are to perform design, experimental implementation, and feasibility testing of KL1, and to define the conceptual specifications of the parallel logic language (refined kernel language) for the prototype machine.

In the initial stage, we designed Extended Self-contained Prolog (ESP), a system description language, by adding the modularization function and the macro expansion function to KL0 created by improving a logic language Prolog. ESP has been used to describe a number of systems including the sequential inference machine operating programming system in the initial and intermediate stages.

Research into parallel logic languages has resulted in the presentation of Guarded Horn Clauses (GHC) in the initial stage. Since Concurrent Prolog and other existing languages, as well as OR parallel Prolog needed multiple environments, the way that they computed was too complex. GHC is a viable and efficient parallel logic language that is suitable for implementing hardware efficiently.

Also, in the intermediate stage, we developed Flat GHC (FGHC) as a subset of GHC that prohibits the use of user-defined predicates in guarded parts.

We decided to base development of KL1 on FGHC. KL1 consists of the following four languages according to the design of the parallel inference machine and its OS.

(1) KL1-b (base): An abstract machine language which will be the basis of the design of the parallel inference machine.

(2) KL1-c (core): The core language of KL1, designed from FGHC as the base with an additional meta-call function.

(3) KL1-p (pragma): The language for priority control and process allocation for an efficient execution on the parallel machine.

(4) KL1-u (user): KL1-c plus the modularization function.

Regarding KL1-b/c/p, we have experimentally implemented and evaluated the processing system on the sequential inference machine (PSI) and have developed a development support system. We are now working on the design of KL1-b/c/p and their experimental implementation on an experimental parallel inference hardware machine (Multi-PSI). For KL1-u, we designed A'UN, a parallel object-oriented language and implemented an experimental processing system to examine the modularization and stream processing functions.

To develop the refined kernel language for the prototype machine in the final stage, we are studying the possibility of supporting the meta-function, functions for program transformation and partial evaluation, and database and knowledge base function, in addition to the functions of KL1. We have done the following.

* We have made meta-programming more efficient and implemented self-appliable partial evaluation programs that can derive a compiler from an interpreter or a compiler from a compiler.
* We have implemented a constraint solver used for constraint logic programming.
* We have experimented to discover whether GHC, a committed-choice language, can be used to search problems by means of a layered stream structure.

2.1.2 Problem-Solving and Inference Software

The goal of the intermediate stage is being approached in two ways with regard to problem-solving and inference software. One way is to implement the core of the OS that is to control and manage the parallel inference machine (PIM). The other way is to study, from a long range view, meta-level inference methods such as inductive inference and analogy, and cooperative problem-solving methods by complement of knowledge.

The OS for PIM (PIMOS) is an integrated logic OS described in KL1. Its basic scheme is to have sufficient functions for parallel software development, and to achieve high-level user facilities.

PIMOS is to be used on KL1 machines such as Multi-PSI and PIM, and I/Os are processed virtually by the front-end processor. We are now evaluating its design by implementing part of it experimentally, while development is proceeding toward the completion of the core at the end of the intermediate stage. At the same time, we are studying parallel programming techniques and program paradigms.

We have proposed ascription for meta-level inference methods. Ascription is a type of non-monotonic reasonings that offers a technique for a common sense reasoning mechanism to integrate default reasoning, analogy, and induction. We are studying it by implementing experimental systems.

2.1.3 Knowledge Base Management Software

The goal of the intermediate stage is being approached in two ways with regard to knowledge base management software. One way is to study knowledge representation methods, inference methods using knowledge, and knowledge acquisition methods. The other way is to develop basic knowledge base management software that represents various types of knowledge in an integrated manner.

The study of knowledge representation, and acquisition and use of knowledge is related to other themes of the basic software system. To study knowledge representation languages, we designed and applied a specific language for each of the fields, including natural language processing and proof support. Regarding the meta-level inference methods, we are studying common sense reasoning mechanisms in the framework of non-monotonic reasoning. In addition, R&D is being performed on the next generation tools for expert systems, in order to verify the effectiveness of the basic software. The next generation tools for expert systems are planned to be developed throughout the intermediate and final stages. In the intermediate stage, we implemented and evaluated an experimental second generation tool for the current generation level, PROTON. As a result, we are studying hypothetical reasoning, qualitative reasoning using deep knowledge, and knowledge acquisition support as element technologies for the next generation tools. We are implementing an experimental hypothetical reasoning module, APRICOT, qualitative reasoning module, Qupras, and knowledge acquisition support module, EPSILON/EM. We are developing an experimental system that incorporates several application areas, while considering their relations to the element technologies. We are also studying knowledge representation and search method by implementing computer GO playing as a research subject of inference using knowledge.

R&D of knowledge base management basic software (Kappa) is being done to provide a DBMS/KBMS for knowledge information processing systems. Kappa provides a constructing function to build a large DB/KB that can be used for natural language processing, theorem proving support, and various types of expert systems. It also can be used as a tool for a number of knowledge application systems running on PSI/SIMPOS. We aim at a knowledge base function composed of a deductive DB plus an object-oriented DB. We have already developed Kappa-I, experimental DB software based on a nested relational DB model, and have confirmed that terms and networktype structure data can be efficiently stored on it. We also developed an enhanced nested relational DB software and designed CRL as the internal knowledge representation language. To store and manage real knowledge, we are now implementing Kappa-II, which is to have the knowledge conversion function, knowledge management function and user interface function. We have begun to create a Japanese-language dictionary consisting of dozens of thousands of words in the nested relational DB software of Kappa-II for a research of natural language processing.

2.1.4 Intelligent Interface Software

In the intelligent interface software, we aim at the establishment of discourse understanding techniques required for natural language processing for the Japanese language. R&D activities have two goals. One goal is to study grammar, dictionaries and execution analysis required for the processings at the syntactic level, and to experimentally implement them as general-purpose tools. The processings just mentioned include morphological analysis and syntax analysis that are commonly needed for high-level natural language processings. The other goal is to develop high-level techniques such as semantic analysis and context analysis that are essential for discourse understanding, and construct discourse models. More specifically, we are conducting this research by implementing an experimental discourse understanding system, specifying various targets and applications.

For the study of syntactic level processing, based on the unification-based grammar, we performed R&D on grammar and parser for morphologic analysis and syntactic analysis, R&D on grammar and processing system for Japanese sentence generation, development of a dictionary, formulation of semantic descriptions and study on the syntactic theory about the Japanese phrase structure grammar (JPSG). We developed a semantic analysis description language (CIL) for describing meanings and discourse structure. CIL features data notation by partially specialized terms and delayed execution control function by constraint solver and stream processing. These software modules are the group of software that will provide the basis for the study of context processing. We have made a library of these software modules as a general-purpose Japanese processing system (LTB) and are using it as our common tool for experimentally implementing discourse understanding systems. LTB includes the CIL processing system and its programming environment, morphological analysis and syntactic analysis system, sentence generation system and grammar, Japanese dictionary and KWIC, and LTB-Shell which is used to integrate and use the other modules.

Various types of experimental discourse understanding systems have been implemented to establish the syntactic/semantic analysis and sentence generation processing method and to promote the study of element technologies such as discourse models, context processing, and deep semantic understanding, while verifying their operation in the system. One of the experimental systems is the discourse understanding system being developed at ICOT (DUALS). We began to test the first version of DUALS in the last year of the initial stage. The first version used Prolog descriptions, on short texts (18 sentences). In 1986, the second version, to which CIL and parallel syntactic analysis method were adopted, was completed. We are now developing the third version, which will use LTB and will be able to understand the meaning of text consisting of 100 sentences, or 2000 words. In addition to these three versions of DUALS, we are developing several experimental systems to study the element technologies for understanding meaning and discourse.

2.1.5 Intelligent Programming Software

The purpose of the intelligent programming software is to study, based on logic programming, various problems in software engineering. We have two themes in the intermediate stage. One is the study of theorem proving support and transformation/synthesis/ verification of programs, based on the similarity between programming and mathematical proof, and the implementation of an experimental system. The other theme is the implementation of pilot models that manage and support the software life cycle from development to maintenance, and the development, as a software knowledge management system, of development support tools that will be put into practice at ICOT as a programming environment.

We are studying intelligent programming software from the point of view of mathematics and logic programming. Based on the similarity between mathematical theorem and program specification (theorem proving and programming), we are investigating a computer aided proof system (CAP) to support mathematical proof as an element technology, and we are building an experimental system. The CAP system consists of a structure editor for proof description, a proof checker, an equation checker, and a proof compiler that creates programs from proofs.

From a logic programming point of view, we are developing an experimental system for program transformation, analysis and verification (Argus). Argus is an experimental system for performing the following operations uniformly on the program with high-level description by using the characteristics of logic programming languages. The operations are transformation and synthesis of the program, extraction and analysis of the basic features of the program, and modification of the program.

As pilot models for the processes from development to maintenance of software, we implemented a prototyping support system that combines programs as modules, and an experimental system that supports consistency verification of design and program by graphically displaying the operation of programs.

As the software knowledge management system, which provides a logic type program development environment for the sequential inference machine OS (SIMPOS), we enhanced library management, program/document generation and management support, and other functions.

2.2 Hardware Systems and Development Support Systems

2.2.1 Inference Subsystem

The goals for inference subsystem R&D in the intermediate stage are to establish a parallel inference machine architecture based on the results of the initial stage R&D on parallel inference machine simulators and sequential inference machines, and to build pilot systems. The parallel inference machine consists of about 100 processing elements. This machine is for parallel inference languages and can efficiently execute the kernel language version 1 (KL1).

The basic concept of R&D for inference subsystem in the intermediate stage is described below.

* To integrate research on hardware and software, the R&D are performed under an integrated organization.

* Regarding the development of the machine, hardware technologies must be accumulated so that they can be used in the final stage. The hardware needs to be used as a software development tool for a wide range of purposes, including the R&D of the final stage.

Regarding the parallel inference machine (PIM), as a pilot system, the performance goal was set at 2 to 5 MLIPS with about 100 processing elements in the planning stage. In the course of R&D in the intermediate stage, we decided to develop a Multi-PSI (V1) as an R&D tool for parallel software such as PIMOS in the development support system, and, by using the results, to develop an experimental parallel inference hardware (Multi-PSI (V2)). Because we determined that we could reach the initial performance goal with the Multi-PSI (V2), we changed the operating performance goal of PIM with 100 processing elements to 10 to 20 MLIPS. This is how the R&D work on inference machines has proceeded during the intermediate stage. The kernel language version 1 (KL1) processing system, parallel software including PIMOS, software development support system and inference system can enhance each other and be combined.

The configuration of the PIM system consists of sixteen clusters, each with eight processing elements (PE), and an inter-cluster network. The PEs adopt tag architecture, high-speed process switching and pipeline processing to realize high-speed processing. PEs support KL1-b as a machine language that can be optimized by compiler, and also support real time GC function using the MRB method. We expect the performance to be 200 to 500 KLIPS per PE. In a cluster, eight PEs are tightly coupled by a shared bus, and the coherency of the cache memory of each PE and shared memory are ensured by parallel caching. Clusters are linked by a message passing network.

The specification design of the intermediate stage version of PIM has almost been completed. Now we are working on the manufacturing designs including VLSI and others. PIM will have one PE for each board, 64 PEs for each cabinet, and clusters connected by networks such as hypercubes.

The Multi-PSI (V2) is also KL1 machine for parallel software development. It uses the CPUs of the small version PSI as processing elements and connects them by a two-dimensional mesh network. Its hardware was completed in F.Y. 1987. The Multi-PSI (V2) provides a basis for R&D on the KL1 processing system and PIMOS. It is also used for the research on the load distribution method and parallel algorithms for the parallel software described in KL1, the debugging method for parallel software, and the program development environment.

2.2.2 Knowledge Base Subsystem

The intermediate stage research and development goal for the knowledge base subsystem is to research the basic technologies needed to implement the knowledge operation mechanism essential for the establishment of a knowledge base machine, and to establish the requisite architecture. Concretely, we established two basic models of the knowledge base for parallel processing and distributed processing, studied mechanisms and architectures to support these models and implemented experimental machines.

We did experiments on a knowledge base operation mechanism by connecting a relational database machine (Delta) developed in the initial stage and a sequential inference machine as the host to each other. They proved the affinity of logic programming language and relational databases, and the high-speed processing of sorting and relational algebra operations by relational database engine. As a result, it was recognized that the knowledge base machine required the following funcitons.

(a) Integrated processing of diverse knowledge representations, broad applicability to multiple fields.

(b) Efficient handling of search, retrieval and knowledge processing using inference for mass volumes of knowledge.

An experiment in the knowledge base function by actively introducing inference function was carried out on sequential inference machine (CHI).

The theme for the distributed processing was to research technologies to integrate and manage multiple knowledge base systems connected through a network in such a way that they became logically one knowledge base system. The experimental system used here consisted of multiple knowledge base systems (PHI) and a host (PSI). The interface between the host and the PHI units was a Horn clause, and the PHI was used as a simulator for deductive database. The PHI consisted of software to manage knowledge and control distribution in the PSI, and an operation processing module with an accelerator to speed up comparisons and searches.

The design and implementation of a knowledge base hardware simulator to research knowledge base parallel control technologies were carried out. This experimental machine consisted of eight element processors and a Multi-port page memory (MPPM) that held the knowledge base. The element processors were connected to the knowledge base operation engine through co-processor interfaces. The knowledge base operation engine was a unification engine for performing operations such as unification join and unification restriction on relational knowledge set. This experimental machine was connected to the PSI and connection tests with the inference machine were performed. As a basic experiment to interface these machines with GHC, the knowledge compiler was implemented and a GHC description of the knowledge retrieval management function was created.

2.2.3 Development Support System

This project is being promoted based on a new type of logic programming language unlike conventional languages. In the intermediate stage, a development environment including a parallel logic language was essential, and therefore one goal was to develop a tool for the development of parallel software. Another goal was to establish a network for the interconnection of development support machines, and an international network for the exchange of information with overseas universities and research institutions by means of electronic mail.

The parallel software development tool began with the first-step development of the Multi-PSI(V1) in 1986 through the networking of six PSI units developed in the initial stage. The KL1 distributed processing system was implemented in ESP, and the characteristics of parallel programs were evaluated. Next, the Multi-PSI (V2) was developed as an experimental machine for the PIM in an inference subsystem. As part of the development of the Multi-PSI(V2), we developed a compact CPU of the PSI for use as a processing element (PE), and developed the front-end processor (FEP) for the Multi- PSI(V2) using this PE. This PE was also used in the small-sized PSI (PSI-II). The KL1 processing system was developed as firmware for the PE to enhance the execution of parallel logic language on the Multi -PSI (V2) and the FEP, and at the same time the pseudo-parallel processing development environment was refined on the PSI-II.

In the development of the sequential inference machine, we also improved the size and performance of the high-speed, large-memory type machine called CHI, in the first half of the intermediate stage.

We organized the software development environment by the development of the in-house network system in which sequential inference machines were connected to LAN and DDX to transfer mails and files to each other. We connected our in-house network to CSnet, UUCP and JUNET in order to build an international and domestic network system. This network system played an important role in improving the efficiency of our information exchange with universities and research institutions.

3 OUTLINE OF THE RESEARCH AND DEVELOPMENT PLAN FOR THE FINAL STAGE

The objectives of the R&D in the final stage are to implement prototype hardware that has a parallel architecture and that can perform high-speed inference and knowledge retrieval, and to develop prototype software that can program efficiently in a parallel logic language for knowledge information processing.

To achieve these objectives, we will determine the organization of the project in the final stage according to the state of the R&D at that time, on the basis of that at the end of the intermediate stage. In other words, the R&D in the final stage will be geared to making a prototype system, using the results that have been obtained up until the end of the intermediate stage. Research will also be conducted into the basic technologies that are needed to realize the final objectives and that may be needed in the future.

The concrete plan of the R&D in the final stage is now being planned; it still open to alteration.

The R&D themes in the final stage can be divided into the three prototype systems: hardware, basic software, and basic application. The prototype basic software system can be further divided into basic software and knowledge programming software.

In the prototype basic application system, prototypes of some application software systems will be developed to verify the effectiveness of various kinds of basic functions of the fifth generation computer and to clarify the actual application, but the actual field of application is still under consideration. For this reason, the R&D plan for the final stage concerning the prototype hardware system and the basic software system are discussed below.

3.1 Prototype Hardware System

The objective of the R&D for the fifth generation computer prototype hardware system is to implement a hardware system into which the following two hardware mechanisms will be integrated through a hierarchical structure network. They are dedicated hardwares for realizing high-speed inference function and knowledge base function on a vast amount of knowledge base in parallel hardware architecture.

This hardware system will be able to execute basic software (parallel OS) and high-speed execution of application software for large-scale knowledge information processing written in parallel logic programming language.

Concretely, the hardware will be implemented by the connection of about 1,000 processing elements to provide the various functions. For the hardware performance of inference operation we aim at is 100M to 1G LIPS.

To realize the functions, performance and scale of the prototype hardware system, the inference function theme will consist of the evaluation of the intermediate stage R&D results by large-scale parallel software. The R&D results will be: the parallel logic language, the execution mechanism, the parallel inference machine architecture and processing mechanism. We will also carry out research into high-speed data transfer between processing elements, mechanisms of parallel inference execution and control, and mechanisms of parallel software execution management and communication control. The knowledge base function theme will consist of the evaluation of the intermediate stage R&D results such as mechanisms and storage schemes suited to knowledge base processing, and research into mechanisms required for the management and operation of large-volume knowledge bases. Both functions will be implemented and integrated through parallel architecture based on parallel logic languages.

Hardware design will take all functions required by basic software into account, and will be based on the language specifications for the parallel kernel language and the execution models. The results of the intermediate stage will be used to develop new techniques and implementation technologies as required.

The prototype hardware system will be constructed to have both the inference function and the knowledge base function. To implement inference function in hardware, we will use a processing element (PE) that has all the functions needed for high-speed parallel inference processing and a cluster of about 10 PEs tightly coupled, as basic elements. In the system constructed by connecting these clusters by means of a cluster network, we will develop the following modules: inference processing module to perform low granularity parallel processing, connection control module to control the connection and manage communications, parallel execution control module to handle load balancing control and interface functions for hierarchical networking, and communication management module to handle I/O processing on the man-machine and machine-machine levels.

We will develop a knowledge operation management module for the knowledge base function. The module contains knowledge operation functions such as knowledge base retrieval processing and knowledge set operation, and knowledge base management function such as storing and updating of knowledge.

3.2 Prototype Basic Software System

The objectives of the R&D of the fifth generation computer prototype basic software system are to provide the following functions: OS functions for an efficient parallel software execution environment by controlling and managing the hardware system, functions for the description of knowledge forming the core in the development of application softwares in knowledge information processing as programs, functions such as cooperative problem-solving and meta-level inference to support the above activities intelligently, functions to construct, manage and use knowledge bases through the structural formation of described knowledge, and functions for natural language interfaces required essentially for interactive human interfaces.

This configuration consists of the prototype hardware system OS functions, basic software to provide the system programming environment, and knowledge programming software to provide the environment for natural language processing, and the description, management and use of knowledge.

3.2.1 Basic Software

The basic software will handle prototype hardware control and management. It takes as its goal the provision of OS functions, and consists of an inference control module for high-speed parallel inference execution control, and a knowledge base management module for knowledge base operation and management.

The essential functions contain resource management and execution management of the prototype hardware system with an inference function and a knowledge base processing function, efficient execution management of parallel software described in parallel kernel language, and efficient operation management for storing and retrieving large-scale knowledge bases.

(1) Inference control module

The inference control module is software primarily designed to handle execution management and resource management for parallel type inference execution. It is described in parallel logic language. The essential function is the load distribution control function, which divides the parallel program run on the hardware module and assigns the load efficiently to multiple processing elements. It also manages memory and I/O.

To bring about these functions, we will conduct research into new methods in the following ways using the parallel inference software developed in the intermediate stage: evaluation and investigation of algorithms for resource management and execution management, and implementation of technologies by evaluation software. Based on this work, resource management and execution management software design and prototyping will be advanced.

For parallel kernel language processing system, object-oriented models and constraint models will be investigated, and the research results of the intermediate stage will be used for the design and implementation of a language and a processing system suited to system programming. The design and implementation of a system programming environment including debugging function and execution monitoring function for the parallel execution environment will also be carried out during the basic investigation.

In this design and implementation, basic research will be conducted into a number of remaining problems, such as load distribution specification function for the system programming language, load distribution control for multiple processing elements and the method of implementation, and a variety of parallel algorithms and programming methods.

(2) Knowledge base management module

The knowledge base management module is software designed to manage the storing, search, and update of large-scale knowledge bases on the prototype hardware. It will include the basic database management functions and the knowledge base management functions built upon them.

On data base management function, we will conduct basic research into efficient resource management schemes for memory and processors, internal data representation and management methods suited to store knowledge, and parallel execution models for set operation and nested relational models. For efficient use of the parallel architecture prototype hardware, research will be conducted into relational operators, extension methods, and implementation methods, with attention paid to parallel processing. Research will also be conducted into database management techniques and implementation methods to process distributed knowledge. Software will be designed and implemented according to the results obtained.

On knowledge base management function, we will investigate deductive database models and conduct research into the internal representation form of knowledge suitable for various knowledge representations, a knowledge base description language for program interface function, and query languages for users. This research will be carried out through the implementation of software.

3.2.2 Knowledge Programming Software

Knowledge programming software is a group of utility softwares developed using basic software. To develop application softwares for knowledge information processing, we aim to provide a range of knowledge programming functions, a development support system, and a user interface.

Research will be performed into the development of a cooperative problem-solving technique to process input problems while avoiding the conflicts and contradictions between knowledge processing softwares developed for different application fields. Research will also be conducted into meta-level inference functions such as common sense decision that approach human intelligence, and into meta-level inference techniques for the learning mechanism.

This software provides the following functions as a step to facilitate the construction of knowledge information processing systems: knowledge programming languages, various programming functions, an intelligent programming support function, a knowledge base construction function by the extraction and arrangement of expert knowledge, a function for using knowledge base efficiently according to the application, a function for reconstructing knowledge base, and all the functions required for the construction of an interactive interface that uses natural language to provide a flexible man-machine interface.

This software will consist of the following three modules.

(1) Problem-solving and programming software module

R&D on the parallel logic languages will include work to find methods that can uniformly embody meta-functions and constraint functions, work on semantics of parallel logic language, and work on a constraint logic programming system that integrates multiple constraint solvers as one of the frameworks for cooperative problem-solving.

Concerning parallel knowledge programming languages, a language function needed in order to represent knowledge will be researched from a number of fields including the search problem field, along with research into the basic concepts, programming techniques, and support functions for knowledge programming based on a parallel logic language.

We will also cover basic research into partial evaluation, equivalent transformation (unfold/fold transformation) and verification for parallel logic programs, as well as research into software for interactive conversion, synthesis and verification functions offering parallel partial evaluation, flexibility and expandability.

Software will also be developed to provide expert support for fields that require precise mathematical assistance, such as theorem proving and algorithm design.

(2) Knowledge construction and utilization software module

To simplify the construction and use of application systems for knowledge information processing, a function to extract expert knowledge, organize it, put it in a data base and refine it, as well as an intelligent inference control function that uses deep knowledge, constraints and hypotheses, will be researched. The software to provide these functions will be researched and developed.

Research will be conducted into knowledge models that represent knowledge about structures and relations in the events that will be handled by experts. Ways to use this knowledge to represent problem-solving techniques will also be researched. Technology will be established for a knowledge acquisition support technique to assist in the extraction, and organization of knowledge from the association recalled to experts by the knowledge model, and for a technique for refining knowledge models to alter knowledge bases to suit specific applications. Software will be designed and implemented.

To implement an intelligent inference control function, research will be performed into use, focusing on element technologies such as the use of deep knowledge, cooperative problem-solving, inference control based on hypotheses, and constraint-oriented problem-solving. Along with research into these element technologies, work on the system will also be advanced, and software designed and implemented.

(3) Natural language interface software module

In a variety of application systems of knowledge information processing systems, this module will provide the basic functions essential for the establishment of an interactive interface that uses natural language.

To construct an interactive interface that can understand discourse, we will conduct research into implementation of basic functions for semantic processing, clarification of introductive methods for contextual processing functions, and the theory and modeling of functions at the basic level.

This module will be composed of functions for the analysis and generation of Japanese at the syntactic level, functions for semantic description, dictionaries needed to implement the application system, and support functions for developing a lexical database, in a software set forming a general-purpose Japanese-language processing system.

The function for analysis and generation at the syntactic level is designed for application fields with a range of several thousand to several tens of thousands of words. Basic investigation will be performed into software configuration techniques and processing techniques for efficient partial semantic processing, and software will be designed and implemented.

On semantic description function, we will investigate situation theory, type theory and constraint satisfaction method for the semantic description language and processing system. We will take parallel execution models into account in order to research into an expanded version or an refined version of the semantic description language CIL that was developed in the intermediate stage. The language specifications and processing system implementation techniques will be designed and software created.

Appropriate fields for the dictionaries and lexical database forming the basis for these functions will be identified, and thesauri, concept structure and system modeling will be researched, as will techniques for their structure and management. Software will be designed and implemented.

High-level processing functions such as semantic and context processing for discourse understanding will require research into semantic description language, techniques of semantic description and systematic analysis of sentences based on engineering principles. All the above are based on situation theory. Appropriate fields will be identified, an experimental system for evaluation implemented, and the general-purpose Japanese language processing system refined and expanded.

4 CONCLUSION

This project is researching new computer hardware and software technologies using parallel architecture based on logic programming language, with the goal of producing a working prototype. The use of over 300 sequential inference machines as a software research and development tool in the intermediate stage proved that the sequential logic language can be effectively applied to a wide range of subjects. Concerning the parallel logic language, we are advancing with hardware research and development, small-scale test software and PIMOS description, but actual utilization of parallel inference machines as parallel software research and development tools is planned in the final stage. The final stage plan will be carried out along the broad lines outlined in this paper, and the details of the concrete plan will be presented in the ICOT Journal and other sources as completed.

Technical information generated by this project is being publicly released through technical reports, technical memorandums and other publications, and has been the subject of international exchange. We will continue to actively participate in such international information exchange programs in the hope that we can pool our knowledge to everyone's benefit.

ACKNOWLEDGEMENTS

This project has been carried out through the efforts of the researchers at ICOT, and with the support of MITI and many others outside ICOT proper. We wish to extend our appreciation to them all for the direct and indirect assistance and cooperation they have provided.

REFERENCE

Kurozumi: Fifth Generation Computer Systems Project, 1987, ICOT, TM-0303