Report on a Research Visit to ICOT

Paul. F. Wilk

Artificial Intelligence Applications Institute,
University Of Edinburgh, Edinburgh, Scotland, U.K.

Introduction

I was invited to the Institute for New Generation Computer Technology from 13th January 1986 to 7th of February 1986. Originally, I had been invited between September and December 1985, but Dr. Fuchi had kindly allowed me to visit later in the financial year so that I could complete existing commitments to the Alvey Logic Programming Initiative, before the end 1985. January was the best compromise for both ICOT and myself because members of ICOT had to complete their end of year progress report for MITI and I expected to be heavily involved in the Alvey Logic Programming Environment Project during the rest of 1986.

I was invited to ICOT for a number of reasons. Firstly, I had received many researchers from ICOT since 1982 and they had become aware of my experience in evaluating AI workstation hardware. Also in the many tedious hours, I had spent evaluating Prolog Programming system implementations both from a qualitative and quantitative point of view. Much of this work had been used by Alvey and the Science and Engineering Research Council in determining choice of hardware and software for both AI and Engineering researchers in the U.K.

My work on the development of Prolog Programming Environments was less well known, but when Mr. Takagi visited Edinburgh as part of his world tour for establishing Electronic Mail connections between ICOT and the rest of the world, our discussions led to a broadening of my invitation to discuss and evaluate SIMPOS, the PSI machine operating system. The fact that I arrived at ICOT safely is testimony that Mr. Takagi has done an excellent job in establishing network communications, because I conveyed much of my visiting schedule to Mr. Kusama (ICOT's research manager) through the electronic mail network.

My reasons for accepting the invitation to ICOT were many. Personally, it was an honor to be invited to ICOT as in my opinion it is currently the world focal point for research in Computer Technology. In addition, despite my Edinburgh colleagues attempts to promote AI research to U.K. government funding bodies, the formation of ICOT was the catalyst that has made AI funding available to many British researchers including myself, and so enable them to remain, and in some cases return to their native Britain.

I arrived at Tokyo Narita airport at the scheduled time. Mr. Takagi met me at the airport which took most of the stress out of getting into Tokyo from Narita airport.

Introduction to ICOT

Mr. Kusama, managing researcher, met me at my hotel on my first day at ICOT. I was given an introduction to ICOT by a slick but somewhat dated slide show; ICOT expect to have it updated by the summer to place more emphasis on the intermediate stage of the FGCS plan. After the slide show Mr. Kusama took me on a conducted tour of the ICOT building. I was taken to each of the five research laboratories at ICOT. This is somewhat misleading though because the second, fourth and fifth laboratories are in an open plane area. The first and third laboratories are in a separate part of the building on the same floor. It is interesting to note that they feel that communication is not all that it should be because of this separation.

There are now sixty five full time research staff in the five laboratories. In addition they have thirty systems support staff. Many members of the staff are new, as 70% researchers returned to their parent companies in June of last year.

There are approximately 300 researchers and engineers outside ICOT connected with the project. Of course, many of these work for the parent companies, but it is interesting to note that the link person to the parent company is quite likely not to be one of their own researchers.

Presentation by 5th Research Laboratory

The 5th laboratory were only established in June of last year. Currently their number is eight, but they would soon lose one researcher who would return to his parent company. But, they would soon receive another three researchers to boost there number to ten. Currently, the group was still at the stage of evaluating expert systems shells. The ones they had acquired were KEE (Symbolics), ART, Knowledge craft, Expert-U and Zeus, the latter two shells are distributed by Japanese companies.

Although they were evaluating these systems it was not there intention to implement their expert system in any of them. The expert system to be developed is extremely ambitious; An Expert System to Evaluate the FGCS Project !

I discussed the work done by the members of the Department of AI at Edinburgh and exchanged research reports published by them. In addition, I explained the role of the Knowledge Representation Systems Trails Laboratory in transferring AI and Expert Systems techniques and technology to affiliated industrial companies.

Presentation by 2nd Research Laboratory

Dr. Yokoi's groups work seemed to be a parallel work that was being done in the Department of AI.

During the presentation by the 2nd laboratory, Mr. Mukai asked my opinion about Object Oriented Programming, querying whether it belonged in the Logic Programming paradigm. I would say that it already exists within the Logic Programming paradigm. I think it is very easy to program in an Object Oriented style in Prolog without having to provide the Object Oriented framework. In my opinion, Mr. Mukai is correct to assert that any extensions to Prolog should be within the Logic Programming paradigm. Unfortunately, Logic Programming purists will probably see Mukai's addition of assignment to Prolog as contravening the Logic Programming ideal.

Presentation by 1st Research Laboratory

I was given three presentations by the 1st laboratory on:

1. Parallel Logic Language.
2. PIMOS - Parallel Inference Machine Operating System.
3. Parallel Application Group.

The parallel logic language is called GHC which is a simplified version of flat concurrent Prolog. There is a restriction that only system functions may appear in the guard part of a clause. Guarded Horn Clause language does not have read only annotations. It differs from Concurrent Prolog in that when goals are to be executed or-parallel fashion, the unification processes suspend until the value of the term in the head of a clause is instantiated from a call in the body. Effectively, I think it is the same as a wait declaration in MU-Prolog. The language also has arrow like annotations to denote direction of dataflow between processing elements e.g. P :- A^:

The Parallel Inference Machine Operating System is in four parts over three layers: KL-1 U (User) contains a module structure and all solution's search algorithm etc. KL-1 C contains the implementation of flat Guarded Horn Clauses. KL-1 P (Pragma) is responsible for goal allocation and processor scheduling. KL-1 B (Base) is a sequential machine implementation of the Warren engine. Effectively KL-1 U language is translated through the KL-1 C and KL-1 P levels into KL-1 B. As far as the PIMOS implementation is concerned, KL-1 C is complete and the other three parts are only partially complete. The dilemma for the implementation group is that they have not yet decided on which PIM architecture to choose. Hence, PIMOS will initially be targeted at Multi-PSI.

The Parallel Application Group.

This group has three members:

1) Takeuchi is working on a Shapiro's algorithmic debugging concept and partial evaluation for Guarded Horn Clauses.
2) Matsumoto has been working on a parallel parser equivalent to Chart Parsing.
3) Ohki is working on Mandala - the Knowledge Representation system.

Presentation by 4th Research Laboratory

The SIMPOS system is currently 120000 lines of source code. A major feature of the system is that Prolog programs are not invocable through a shell of any sort, only from the debugger. Currently, a capability based protection mechanism for the operating system is in place, but the system has not been released yet. A major feature of the system is its object oriented approach to program development. It differs from Smalltalk in that it does not explicitly pass messages between objects. Presumably, this is replaced in SIMPOS by shared variables.

Currently, there are two levels of debugging; a low level for system debugging that can be used as a probe to follow particular slot values in the system that are typically accessed by more than two processes. There is also the usual source code debugger which is an extention of Byrd's box model. I was given a demonstration of a multi-process debugger by Mr. Ishibashi which would enable SIMPOS to be debugged easily within the ESP language environment.

Recent performance figures given for PSI are interesting. A re-implementation of Warren engine on PSI will increase the LIPS rating to around 100K. This performance improvement comes from the compile time analysis of source code that is made possible with the Warren engine. This compares with a 114K experimental re-implementation of their original engine. This performance improvement is mostly attributed to "not so clever" performance optimizations at the firmware level. By moving to a Warren engine and combining this with the performance optimizations made at runtime they expect to break the 200K LIPS barrier on PSI. Most of the performance improvement at runtime was attributable to the residual control possible with the firmware architecture of PSI using residual control in registers. Benchmark figures given for a number of problems are interesting. Generally, the firmware optimization has led to a two fold performance improvement apart from NREV, which has improved 3.6 times.

At the moment PSI has 1K words of general purpose registers. It is thought that about only 64 are necessary. 2 bits of the instruction set are used for GC and there are six bits for Tag. The six bits are directly connected to the control unit. Mr. Taki then reflected on the lessons PSI has taught. 64 types of conditional branch instruction are possible in PSI of which about 16 types are used regularly.

The bridge to the parallel inference engine is through Multi-PSI. An overview was given of the Parallel Inference Machine architectures that were being studied. The parallel reduction machine originally had the three network connections between the various elements in the system. However, the simulation study has led to the belief that in fact only one is necessary and there is no performance degradation. Technically the engineering of this machine is not too difficult. The approach is interpretive and can be seen as a top down approach to the solving of Prolog programs. This contrasts with the data flow machine which is technically difficult to manufacture; as the design of new types of circuits is required. Furthermore, it is a compilation based system using a compiled dataflow graph to direct the execution of the processor. So far simulation studies using a 68000 based simulator reveal that there is little difference in the performance of the two architectures. A further architecture developed by Fujitsu is being considered as the control process for goal allocation and processor scheduling.

The expected performance of this architecture is 4 micro seconds for the high speed ring and 50 kilo bytes per second for the high speed switching network. The basic algorithm for load balancing is to divide and conquer the proof tree. When a processor is free it sends a packet request around the high speed network. A heavily utilized processor will send part of its pending computation across the high speed data network to the processor that requested some work. From performance figures, it is deducible that the message routing overhead is minimal compared with the time spent in unification and procedure calling. It is likely that the Parallel Inference Machine will integrate the good points of all three systems effectively taking the middle ground between top-down and bottom-up approach to architecture design.

Presentation by 3rd Research Laboratory

The presentation of the third laboratory focused on the relational knowledge base machine.

Basically this model extends the relational model to include terms. Resolution is performed by using unification retrieval. There are three types of operation: unification restriction, unification join and projection. At present, the system is simulated using four unification engines.

Collaborative Research at ICOT

The 4th laboratory think that a re-implementation of the New Warren machine would take PSI up to about 100K LIPS. My benchmarking efforts on PSI would seem to confirm this. PSI's performance on the dynamic execution efficiency benchmarks generally make it perform at twice the speed of DEC-10 Prolog. For static execution efficiency tests such as NREV, the current implementation is half the speed. Re-engineering of the implementation should bring the static test results in line with the others. In terms of memory utilization, the PSI machine is far superior than the original DEC-10 implementation. This is partly due to the virtual memory management of PSI though. One source of disappointment is the PSI garbage collection algorithm. Dr. Chikayama and I have discussed proposals for improving this. Technical observations about the performance of PSI are contained in a separate report.

Many discussions were held between Dr. Chikayama, Mr. Takagi and Mr. Taki on both the design of SIMPOS, PSI and Multi-PSI. My ideas were fed back to them as and when they arose. In addition, during the course of my benchmarking efforts, I had great delight in giving bug reports to the SIMPOS development team.

I look forward to the 4th laboratory running my benchmarks on Multi-PSI. Essentially it will be a loosely coupled system but with more rigorous communication protocols than the SIMPOS operating system. At the moment, SIMPOS is only seen as suitable for mailing and remote file copying. But, the Multi-PSI system will have a file server.

Other topics discussed were the use of the SUN-3 to be delivered. It appears that this will only be used as a network communication facility to external networks. This is so as to overcome any reliability problems they might otherwise have by direct connections of PSI machines to the outside world. Given the problems we have had with maintaining the mail system at Edinburgh, this would appear to be a shrewd decision which will result in a valuable saving of otherwise wasted manpower. I commented that I thought that the SUN-3 would seriously compete with LISP machines in terms of performance and certainly in price. Mr. Taki agreed and also thought they threatened PSI in its current form, and this is why super PSI was required.

During my stay at ICOT I was involved with a small working group on the design of a Prolog listener for the PSI machine. The group had only met on two previous occasions and so the specification was still at the "hand waving stage". The group consisted of Dr. Chikayama, Mr. Furuichi from Mitsubishi and Mr. Ishibashi of the 4th Laboratory, who had implemented the ESP debugger.

Although Dr. Chikayama considered that the programs running under Prolog on the TOPS-20 system were relatively small it was still a tedious job to convert the programs to ESP which required the developper to create a class object, catalogue it and enter it into the librarian etc. So it was their intention to create a DEC-10 compatible top level though with extended features. So we had a broad discussion on the qualities of a number of Prolog systems e.g. to have the option of calling ESP programs from the top level. The group was also interested in a tightly coupled interface with a text editor they were developping called PMACS. A major design decision was whether control of data should be handled through the editor or Prolog. I had suggested the elegance of an interface manager. So in the case of EMACS to create an interface manager it would be necessary to decouple control. However, this would probably require a major effort to re-implement PMACS. So Dr. Chikayama decided on a data buffer through which either both Prolog and the editor could directly write to and read from or which only the editor could read and write directly to; with Prolog having to go through the editor. In my decoupled system access to the buffer would be controlled by the interface manager. Dr. Chikayama favored the DEC-10/EMACS style of working as opposed to the Micro Prolog context editor style of working where the clause database is manipulated directly.

At my second meeting with the group, Mr. Furuichi had come up with a design for the Prolog listener. My prejudices about whether the editor or Prolog or both should have control over interaction in this approach come from the fact that all our editors are written in C. PMACS on PSI is written in KL0. So it is conceptually easy for both the Prolog listener and the PMACS editor to have direct read/write access to a text buffer. In PSI's case, it is easily possible to have the editor and the Prolog listener as a single process. No language interfacing has to be done. By the end of my visit, we had a paper design for the Prolog listener which would be produced as an ICOT technical MEMO.

Lectures at ICOT

My first formal presentation lasted about an hour and a half, and I managed to talk in discussion afterwards for about half an hour. There were about 30 members of the PIM/SIM working group present. Most of these were from the 4th Laboratory with members from parent companies and Japanese Universities. I was familiar with the work of one of the visiting Professor whose area of research was adaptive computers in particular QA-1, which I had come across.

The talk went reasonably well although my presentation of the Qualitative and Quantitative results was far from systematic. However, I think I managed to cover all the points I wanted to make without going in to too much depth. I stressed that they should look for the detail in the technical reports. The most difficult part was to give them leads on improving the inference machine. I was reluctant to do this, because I don't think enough empirical analysis has been done of major Prolog AI applications.

Here are some general ideas on areas for performance improvement of the PSI Machine Prolog implementation:

1. More compile time analysis of argument types, and switches to direct compiler on what code optimizations to make e.g. fastcode/compactcode, bignum or not.

2. Improved indexing mechanisms required for large clause bases; for example a different indexing strategy for different classes of clause particularly where procedures only contain facts. Automated program/data analysis ought to generate indexing strategy suitable for the problem.

3. More optimizations akin to register based argument passing in Warren machine. The ability to get at the microcode ought to lend the PSI architecture to many more forms of residual control.

4. Rationalization of machine hardware to reduce and simplify the number of machine instructions particularly when they aren't generated by the compiler.

5. More effort in source code program transformation optimizations.

6. Reduction in number of memory accesses for fetching data i.e. make sure one to one mapping between virtual and real machine word sizes.

7. Caching strategy based on runtime procedure call analysis to try and determine a working set of prolog procedures.

8. More dynamic analysis of AI program behavior as there is a marked contrast between CHAT-80 (deep backtracking) and PRESS and NREV benchmarks (shallow backtracking).

9. Analysis of performance and utilization of builtin predicates, particularly all solutions predicates. Possibly a separate microcoded inference engine. Particularly important is the search for more of these higher order functions.

10. Garbage collector as a separate processor. This will probably be easier with PIM than PSI. Note that applications are being written which create an abundance of garbage atoms, so it is necessary to garbage collect atoms. Intuitively, I would expect a stop and copy garbage collector to be more effective for large none determinate cases because individual processors can be collected in the isolation utilizing free processors.

11. More residual control in general recursive case may be possible in firmware, it may be necessary to adapt processor in parallel machine, e.g. detection of recursing down a long structure.

12. Use associative memory to improve case where Warren engine optimizations are disabled e.g. the NREV variations, possibly in conjunction with 7. This is particularly useful for CHAT-80 type programs with procedures that contain many facts. This memory is also useful for structure unification.

My second talk was on Prolog Programming Environments, and as usual I managed to speak for about two hours. They were mostly the ICOT SIMPOS development team who seemed mostly to be a floating population. They were resident on the 22nd floor of the same building as opposed to the 21st. The head of the SIMPOS group of the 4th laboratory was from the Mitsubishi laboratories. The audience was fairly responsive to my talk. The major questions were related to the debugger. Comment was made that Prolog tracing debugging was often at too low a level particularly if you had some sort of language built in Prolog that was being used for the development. I had mentioned the principle of "What You See is What You Debug" in my talk.

There was an interesting discussion between Dr. Chikayama and one of the CIL researchers. The former had designed ESP whereas the latter was one of 2nd laboratory "DUALS" researchers. ESP doesn't provide assert and retract nor any of the Prolog higher order functions. Consequently, the DUALS people found it difficult to port their system to PSI and contended that in its present form the Object Oriented approach of ESP did not make it possible to produce efficient algorithms for AI problems such as Intelligent Planning.

Once again many of the researchers were interested in soliciting my views on Prolog versus Lisp. Although they were working on SIMPOS, many of them preferred LISP. But it was apparent that it was because LISP is what they had been used to and that many more man years had been put into Lisp development. The major advantage of Prolog and Logic programming ought to become apparent with the realization of parallel inference machines, because I can see the parallelism in the expression of Logic Programs whereas in the Lisp languages I have seen for parallel processors, the language has been distorted considerably further.

Other Visits and Presentations

I visited the NTT Musashino laboratory with Mr. Goto. It is about 90 minutes from ICOT by Japanese National Railway. Outside the main entrance was a 12 foot diameter loud speaker which was used as part of the NTT research into acoustics.

I was given a twenty minute introduction to the research work done in NTT laboratories. The Musashino laboratory was by far the largest with about three thousand researchers of a total work force of about 300000.

At NTT, I was shown the DIPS computer, a dataflow machine built by Dr. Hasegawa. He will present his work on this architecture at the IEEE Computer Architecture Conference in Tokyo. The major impression of this machine was its considering that only two processors were connected. Communication between processors was currently by a simple bus, but it was agreed that a switching network would be required beyond eight processors. According to the ICOT researchers, the granularity of control in the system was at too low a level. Mike Reeve of Imperial college had also made the same point when he had visited. Programming the machine also depended on the use of an in-house functional language called VALID. Dr. Yokota at ICOT later pointed out that the examples that were used in demonstration were rather small number crunching examples which for the claimed parallelism were somewhat less efficient in performance and monetary cost than many sequential processors. Applicability of such an architecture would have to be proven by the building of large AI systems.

The next demonstration was the VALID programming environment that was implemented on a VAX-750 under VMS (it typically had to support about twenty researchers by mid-afternoon). A description of VALID will be made available at the forthcoming IEEE Computer Architecture conference. The characteristics of the language were similar to many found in other functional languages.

Next I was shown the ELIS machine that had been built NTT as the target hardware for the TAO language developed by Okuno, Takeuchi et al. ELIS was to be used as the in-house AI machine for developing AI applications. The processors were named after metallic elements although this included hydrogen ? The current interpretive implementation of ELIS gave a performance improvement of about twice a Symbolics 3600. The development of a TAO compiler, which might be complete by next March, would further improve performance by four. The technology used for ELIS was TTL but there were plans for a VLSI design.

TAO was intended to be a mixed paradigm language which aimed to combine C, Prolog, Smalltalk and Lisp. I was amused to see that the language looks like Lisp. The demonstration I was given of TAO covered the usual small benchmarks i.e. NREV, Fibonacci series etc., but with algorithms written in the different programming styles (although they all still looked like Lisp to me).

They assured me that the Prolog algorithm I was looking at was in the declarative style of Prolog programs. However, I was disturbed by the preponderance use of the word assert that seemed to proceed each representation of what didn't appear to me to be a Prolog clause ? In essence, the designers saw the requirement for mixed language programming just as we at Edinburgh do. However, I think the general applicability of the language outside NTT is zero, because they have to rewrite any imported software they may care to use for their AI applications, or write C, Prolog, Lisp and Smalltalk interfaces.

The final presentation before my talk was by the Expert System Knowledge Engineering group. The work was called KRINE (Knowledge Representation INference Environment), which has been used to build a CAD system as a knowledge based system for the design, synthesis, verification and production of circuits for VLSI. The environment claimed to parallel that of LOOPS giving the user a multiple paradigm programming environment.

During the final week of my stay at ICOT, I visited The Electro Technical Laboratory at Tsukuba Scientific City. I was given presentations on the URANUS system, a Knowledge Engineering environment implemented by Dr. Nakashima. I also had a presentation from the developpers of an expert system which gave expert help on the Japanese Patent laws. While at ETL, I gave the same presentation of my research as I had given to NTT.

I was impressed by the enthusiasm and expertise of the researchers at NTT and ETL. We have agreed to exchange research reports in the future and communicate using electronic mail via ICOT.

Other Discussions

I had an executive lunch with Mr. Hiroichi Hiroshige (Executive Director of ICOT), Dr. Fuchi (Director of ICOT) and Mr. Takashi Kurozumi (Assistant Director). The topics we discussed were: Logic programming initiative, recruitment of researchers, Prolog standardization effort, AI at Edinburgh, British style of management, National and Internal research competition, Research funding and Golf.

On January 27th morning, I was introduced to Dr. Treleaven who was visiting Japan for a week. On the Friday, he was to give a talk to ICOT about an Esprit project 415, which has been given 16 million pounds to research Parallel architecture and Language for AIP - a VLSI directed approach. A consortium of Phillips, GEC, SCELT, Bull, Nixdorf, AEG are to study a specific class of parallel computer. Note that in addition the companies are collaborating through working groups on the study of common design problems.

Results of Collaborative Research

Visiting ICOT has enabled me to make contact with many new researchers. In particular, there is agreement to exchange research reports and software with ICOT, NTT and ETL, where possible.

My writing of a detailed report on the work of researchers outside my field of expertise will enable my colleagues in the U.K. to establish communication with the appropriate researchers in the U.K.

I have been able to feedback many bug reports on the SIMPOS operating system and made qualitative and quantitative suggestions on how I think PSI and SIMPOS can be improved. These results will be detailed in a separate technical report. Included in this report are quantitative and qualitative results of the PSI machine.

I have been able to project my own research work and that of collaborating workers in other U.K. institutions through the four presentations I have given during my stay, and more importantly through the social contact I have had with Japanese researchers outside normal working hours.

I have arranged with Dr. Furukawa for the involvement of ICOT and other Japanese researchers in the British Standards Institute effort to standardize on Prolog.

Research Environment at ICOT

The research environment I found at ICOT is similar to the one that exists at Edinburgh, the main distinction is the spoken language. My only concern is that ICOT has outgrown its premises. At Edinburgh, we have the same amount of floor area for about a third the number of researchers. While I have no problem in concentrating in these crowded conditions, I think it is a problem for many researchers. Furthermore, most of the Japanese researchers were surprised that I wore a T-shirt most days. I don't think the air conditioning can cope with heat output from computing equipment and ICOT researchers brains; its too hot!

Dr. Fuchi's brain child is a success. The fact that ideas could be freely exchanged between researchers from competing companies under one roof had set a unparalleled precedent in the advancement of one countries future technology, and indeed the world's. In my opinion, the inability of Alvey to create a similar air of collaboration between industrial companies and academic researchers, who in many cases are not allowed to publish their results, could be the stumbling block of many projects.

FGCS Project Future

The goal of the FGCS project - to produce an AI system which can evaluate itself is an ambitious one. During my stay at ICOT, I have suggested that a first step towards this goal should be to produce an Intelligent Process Planning System. I think that many of the difficult problems that ICOT will face in the latter stages of the intermediate stage of the 10 year project show themselves vividly.

In my personal opinion, if ICOT achieves only ten percent of its first stated goal, the project will be a success. I believe it is already a success, because the creation of a software industry in Japan particularly in AI will have the same impact on the world as the Japanese Electronics Industry has.

Conclusions

Let me conclude by saying that I have totally enjoyed my research visit to ICOT. In gaining my Ph. D, I was humbled into realizing how little I knew about computing and AI; four weeks at ICOT have given me that same feeling again.

Acknowledgments

I would like to thank Mr. Takagi for suggesting me as an invited researcher, Dr. Fuchi and Dr. Uchida for making it possible. Not least, I wish to express deep gratitude to Mr. Kusama who was the perfect host during my visit to ICOT and to all the other researchers who took their turn at chaperoning me.

Research and Development Career

I was a Research Assistant at Hatfield Polytechnic from October 1978 to June 1982, supervised by Dr. Gordon Bull. My research during this period resulted in the award of a Doctor of Philosophy and covered the following topics;

1. Dynamic computer architecture.

2. User microprogramming.

3. Program/microprogram development environments.

4. The software/firmware production life-cycle.

5. Bit-slice computer architecture.

6. Computer system reliability.

From June 1982 until September 1984 I was employed as a Research Fellow in the Programming Systems Group of the Department of Artificial Intelligence under the supervision of Robert Rae and Professor Jim Howe. My work during this period covered a number of topics:

1. The implementation of a Prolog interpreter in Pascal.

2. The extensive benchmarking and analysis of Prolog systems on a large number of computers.

3. The evaluation of single user machines for Alvey and the Science and Engineering Research Council to determine choice of Unix machine for AI/IKBS researchers in the UK.

4. The original porting of C-Prolog from the VAX to many different Unix Systems.

5. The original porting of CHAT-80, PRESS and Many of the Edinburgh Prolog library utilities to Unix Systems.

6. Additions to the Edinburgh library, e.g. a code unfolder and an extended and-or tree Prolog debugger.

7. The study of Lisp Programming Environments.

8. Involvement with the Alvey Logic Programming Initiative particularly in the area of Logic Programming Environments.

From October 1984 to the present time I have been employed as a Project Leader in the Artificial Intelligence Applications Institute, at the University Of Edinburgh. Initially, work has involved surveying Programming Environments in general and preparing a functional specification for Logic Programming Environments.

Current work includes extensions to our New Implementation of Prolog to support RPC and IPC; the design and implementation of a graphical interface to Prolog and other AI languages; writing a prototype interface manager in Prolog; integration of the Edinburgh Prolog toolkit using the concept of a metadatabase; extension of run-time debugging and tracing facilities using graphics, meta-interpreter extensions and intelligent error reporting. Also, I am responsible for monitoring the performance of Prolog systems and have implemented some tools to study the dynamic behaviour of Prolog programs.