REPORT ON VISIT TO ICOT

September 29–October 17, 1986

Steve Gregory
PARLOG Group
Imperial College

INTRODUCTION

I was honoured to receive an invitation to visit ICOT for a period of three weeks—my second such visit—to exchange ideas on the design and implementation of parallel logic programming systems.

My first long visit to ICOT was almost exactly three years ago (October 10–28, 1983), so I was looking forward to seeing how things have changed. Of course, I was not completely uninformed: I had also visited ICOT briefly during the FGCS84 Conference, and my colleagues Keith Clark and Mike Reeve have visited ICOT a few times since then. Moreover, we are very fortunate at Imperial College in being visited quite frequently by researchers from ICOT, and by other people involved in the FGCS Project. I also get the chance to talk to ICOT people at conferences and workshops, etc. and we keep in touch by mail. Therefore, we maintain good research links, which have proved mutually beneficial to our work. Nevertheless, the visit provided a good opportunity to fill gaps in my knowledge about ICOT research, and get an overall view of the FGCS Project.

PRESENTATIONS

I was asked to give two lectures/seminars at ICOT: one was to be an overview of the progress of the PARLOG project; the other was on implementation issues concerning parallel logic programming languages. Due to scheduling constraints, the implementation talk took place in the second week, before the overview talk, which was arranged for the third week.

In my lecture on implementation issues, I began by outlining the current state of the PARLOG language, and the motivation for its design; I then presented two abstract computational models which are being used as the basis of PARLOG implementations. Finally, I explained the details of the two main implementations of PARLOG that are being developed: one on conventional, sequential machines; and another on the ALICE machine, a parallel reduction architecture.

I particularly enjoyed giving this talk. Even though it was very technical, the audience (many of whom were experts in GHC implementation) appeared to understand my presentation and asked some very intelligent and searching questions.

In my second talk (logically the first), I outlined the history of the PARLOG project, together with its current status. This talk was not so interesting because there was insufficient time to go into the technical details of any subject. The talk was followed by a demonstration of the Sequential PARLOG Machine system running the PPS (PARLOG Programming System), a declarative programming environment designed and implemented by Ian Foster.

DISCUSSIONS AT ICOT

I had several interesting and fruitful discussions on technical issues with many ICOT researchers. In addition, various people took the time to explain their research to me.

Ueda-san and I discussed various aspects of language design and implementation. Although I was familiar with most of his work, I was interested to learn about his ideas on distributed unification. The designs of our respective languages (PARLOG and GHC) are now essentially the same, aside from some technical considerations concerning whether guards should be "flat" or "safe". Our discussions therefore concentrated on two main issues:

1. The specification of the "control metacall" was discussed. This is an extension to the basic language that appears to be necessary for realistic applications. The control metacall adds to a flat language the power of "deep" guards, as well as additional functionality such as: fine control over an object evaluation, exception handling, priorities, etc. Of course, the provision of the metacall makes an implementation more complex; something which should be avoided. It is essential to identify exactly what features must be provided by the metacall, without presenting an unacceptable overhead. Another problem concerns the declarative semantics of the metacall primitive: we discussed how stream communication between an object program and a meta program can be implemented correctly.

2. Some language features may need to be reconsidered in the light of experience with parallel implementation. I was interested to hear Ueda-san suggest the possibility of abandoning full unification for output arguments, especially since we have only recently adopted full unification in PARLOG, an idea inspired by GHC. Full unification may be a problem in a parallel implementation. Another problem discussed was the difficulty of testing whether a term is an unbound variable, and testing the equivalence of two variables, in a parallel implementation. It was also suggested that variable/variable bindings should be disallowed in clause bodies, for similar reasons.

Some of the above issues were discussed in larger meetings, including Tanaka-san, Chikayama-san, Miyazaki-san and Taki-san. Tanaka-san was particularly concerned about the metacall features necessary to implement the user interface of an operating/programming system. Chikayama-san raised many interesting issues, including the problems of parallel implementation and deadlock detection. I was particularly interested to learn of his proposal for an efficient, concurrent garbage collection technique. This is based on the idea of a 1-bit reference count: although it may not collect all garbage, it incurs much less overhead than a full reference count scheme. I look forward to reading about his algorithm in detail when the design is complete.

Miyazaki-san and Taki-san presented the implementation of FGHC on the multi-PSI. Although the paper describing this implementation is only available in Japanese, they explained the details so that I could fully understand it. I also saw an impressive demonstration of the system running the n-queens problem; I was particularly fortunate in seeing this system in action, because it had only just become operational.

Miyazaki-san also demonstrated the sequential FGHC implementation on the PSI machine. This appeared to be a very powerful and user-friendly system. Unfortunately, I had insufficient time to master the system myself, especially because the manual was only available in Japanese.

Ueda-san explained the various implementations of GHC that are under way. I was impressed by the amount of effort being expended on this implementation work, not only at ICOT but elsewhere. As well as the FGHC systems running on top of Prolog (on DEC-10 Prolog and on the PSI), and the parallel implementation on multi-PSI, there are at least two serious sequential implementations. One is an emulated abstract machine approach (KL1-BS), being developed in conjunction with Fujitsu Laboratories. Another interesting system is a native code compiler for the VAX, being implemented by Mitsubishi. The latter is specially oriented toward high performance: it implements only the deterministic form of FGHC, to avoid much of the run-time overhead.

The details of the KL1-BS instruction set were explained to me by Kimura-san and Chikayama-san, who wrote the FGHC to KL1-BS compiler. I was surprised that the compiler is written in Prolog, not in FGHC. Kumon-san of Fujitsu explained the KL1-BS emulator.

Goto-san of the Fourth Lab. gave me an overview of the work on the PIM project. I learnt that the PIM is to be a distributed system comprising clusters of tightly coupled processors. While the multi-PSI system will be used as a testbed for the loosely coupled system, a Sequent 21000 shared memory multiprocessor is to be purchased to experiment with the design of each cluster.

In the applications area, Fujita-san explained to me his work on partial evaluation. His work is a development of that of his colleague Takeuchi-san, who has returned to Mitsubishi.

It was interesting to meet Hirata-san of the University of Tsukuba when he visited ICOT. I already knew of his language Oc, which is very similar to FGHC. He now has some interesting ideas on distributed implementation and has proposed a new dialect, named Doc. Doc imposes restrictions on unification to overcome some of the difficulties with full unification in a parallel system.

VISITS

I found time to visit two places outside ICOT in my final week.

On October 14, I visited Fujitsu Laboratories in Kawasaki, primarily Hattori-san's group. This was of particular interest to me because of the collaboration recently established between Fujitsu and the PARLOG Group, as well as the FGHC implementation work that is being done in conjunction with ICOT. On my visit, I gave a short presentation on the work of the PARLOG Group, and learnt about the Kabu-wake method (of which I already knew) and the FGHC implementation work. I also saw a demonstration of the Kabu-wake method, and of the Cellular Array Processor (which has nothing to do with logic programming but generates impressive colour graphics!).

On October 17, I visited Professor Aiso and Professor Tokoro at Keio University, and saw various demonstrations of the work in their laboratories, including the SM2 multiprocessor and the Orient-84K object-oriented language. I was most interested in the demo of the P-Prolog system.

IMPRESSIONS OF ICOT

The physical environment at ICOT is much the same as three years ago. While ICOT occupies a greater proportion of the 21st floor, there are now almost twice the number of researchers, more than 70, so the density is similar (i.e. very crowded). One effect of the increased size is that the labs are now split across opposite sides of the building. In particular, the First Lab. and the Fourth Lab. (who work together) are physically separate, which may make communication more difficult than formerly.

In common with most industrial research laboratories, including those in other countries, ICOT researchers are almost always to be found at their desks, except when involved in formal meetings. I personally find the academic environment more conducive to research, where people work individually and exchange ideas informally. There seems to be less chance for such informal discussions at places like ICOT. On the positive side, the formal meetings such as the Working Groups organized at ICOT seem like a very good idea, providing a way for people, including those outside ICOT, to keep in touch with each other.

An unusual feature of ICOT is its policy of short-term secondment: researchers return to their companies after about three years. Aside from the chiefs of the laboratories, very few people who were at ICOT on my previous visit are still present. This is undoubtedly a loss to ICOT, though it probably benefits the companies.

I was surprised to find that the DEC-20 mainframe is still the main computing facility; the PSI machine appears to be relatively little- used at present. I assume that the PSI will gradually replace the DEC-20. Even so, ICOT is unusual among research institutions in not adopting the Unix operating system as standard (there is only one reliable Unix machine, which is too small for general use). I don't know whether this is a disadvantage to ICOT, but our computing environment at Imperial College has greatly improved since Unix became established.

In terms of research, I can only comment on the work on parallel logic programming, which is the area of interest of the PARLOG Group. This work is mainly performed in the First Lab., but includes some of the work in the Fourth Lab. as well as some work outside of ICOT.

Three years ago, ICOT was one of only three institutions (including Imperial College and the Weizmann Institute) committed to the idea of "committed choice" parallel logic programming languages. This is still the case, but all three projects have made considerable progress since then, and grown in size. As well as developing many application areas, the problems in implementation have become clearer, and this has had an impact on the language designs.

In the year between my first visit and the FGCS84 Conference, I felt that the work in parallel logic programming had not progressed very far: there had been a lot of work attempting to implement Concurrent Prolog. This work seemed very difficult and was not conclusive, though it provided useful experience. Perhaps the most significant breakthrough came at the end of 1984 with the design of Guarded Horn Clauses by Ueda. This language adopted some of the ideas of PARLOG. Since that time, the designs of PARLOG and of GHC have converged, so that they are now very similar. In the last two years, Ueda-san and his colleagues have done some very impressive work in the design and implementation and application of GHC. There is no doubt that ICOT has made a tremendous contribution to this area of research, and will continue to do so.

It is interesting to observe how the designs of parallel logic programming languages have become progressively simpler, as experience has been gained with implementation. It seems likely that further simplification will be necessary in the future, as parallel implementations start to be developed. ICOT's multi-PSI implementation already exists, so ICOT is likely to lead the way in this field.

CONCLUSIONS

On my previous visit to ICOT, we had many discussions on the design and implementation of parallel logic programming languages, a subject which was only just beginning to be studied at that time. Then, many interesting problems were identified and discussed, even though they were not all solved immediately.

I am confident that the discussions that we have had on this visit will have similar fruitful results, posing problems that both ICOT and the PARLOG Group will need to work on over the coming years.

ACKNOWLEDGEMENTS

I am very grateful to Kazuhiro Fuchi for the invitation to visit ICOT, and to Koichi Furukawa, the chief of the First Lab. I would specially like to thank Kazunori Ueda, who efficiently organized all of the technical aspects of my visit.

Hiroyuki Kusama, the ICOT research manager, took care of my hotel reservation and all the administrative arrangements and made sure that everything went smoothly.

I would also like to thank Yasunori Kimura for organizing my trip to Fujitsu Laboratories.

CURRICULUM VITAE: S. GREGORY
31 August 1986

PERSONAL DETAILS

Name	Steven Gregory
Address	5 Ash Court 56 Worple Road Wimbledon London SW19 4EY England
Telephone	01-947 6322
Date of birth	15 December 1958
Nationality	British

EDUCATION AND RESEARCH

Education

October 1977 - June 1980:

Department of Computing
Imperial College of Science and Technology
University of London
180 Queen's Gate
London SW7 2BZ
England

Research

October 1980 - December 1981,
January 1982 - September 1985 (part time):

Department of Computing
Imperial College of Science and Technology

Supervisor: Dr Keith Clark.
Thesis title: Design, application and implementation of a parallel logic programming language.

Abstract:

Declarative (functional and relational) programming languages have become very widely used in recent years, largely because of their high-level nature: declarative programs can be understood without reference to the behaviour of any particular machine. An increasingly important influence in favour of declarative languages is that, due to their lack of side-effects, they are well suited to parallel evaluation. Among declarative languages, logic programming languages are claimed to be particularly human-oriented. However, the most common logic programming language, Prolog, embodies a control strategy which is designed for sequential evaluation.

This work explores the parallel evaluation of logic programs. In principle, two types of parallelism are possible in logic programs: and-parallelism and or-parallelism. Many combinations are possible, but not all of them can be efficiently implemented.

The main objective of the research described herein is the design of a new parallel logic programming language: PARLOG. PARLOG incorporates into logic programming the idea of committed choice non-determinism from Dijkstra's guarded commands and Hoare's CSP language. This feature, combined with "mode" declarations, makes possible the and-parallel evaluation of relation calls as processes, with stream communication through shared logical variables. This "stream and-parallelism" is the logic programming analogue of Kahn and MacQueen's model of parallel evaluation for functional programs. It constitutes a powerful programming paradigm, and one which can be efficiently implemented on parallel computers.

The design of PARLOG is presented, and its use illustrated, by means of examples. Particular attention is paid to the new

applications of logic programming made possible by the realization of stream and-parallelism.

Ease of implementation is a very important consideration in the language design. In this regard, it is shown how the features of PARLOG allow programs to be largely compiled into efficient low-level instructions. The final chapters demonstrate the principles of implementation of the language on a range of parallel architectures.

QUALIFICATIONS

1980:

B.Sc. (Engineering): First Class Honours
in Computing Science,
University of London

ACGI (Associate of the City and Guilds of London Institute)

1985:

Ph.D. in Computing Science,
University of London

DIC (Diploma of Imperial College)

EMPLOYMENT

9 August 1976 - 16 September 1977,
28 March 1978 - 21 April 1978,
26 June 1978 - 22 September 1978:

Wootton, Jeffreys and Partners (Computer Consultants)
Cemetery Pales
Brookwood
Surrey GU24 0BL
England

Programmer.

Involved in the development of commercial software written in Fortran for PRIME minicomputers, e.g. nominal ledger system, cheque writing system.

9 July 1980 - 8 September 1980:

Department of Computing
Imperial College of Science and Technology

Research assistant.

Enhancements to IC-PROLOG, an experimental logic programming system written in Pascal. Implemented some new language features such as pseudo-parallel evaluation.

10 September 1980 - 22 November 1980:

School of Computer and Information Science
Syracuse University
Syracuse
New York 13210
USA

Graduate assistant.

Continued enhancements to IC-PROLOG logic programming system.
Teaching assistant in logic programming course given by Keith Clark.

1 January 1982 - 30 September 1985:

Department of Computing
Imperial College of Science and Technology

Research assistant.

The design, application and implementation of PARLOG, a parallel logic programming language.

1 October 1985 - present:

Department of Computing
Imperial College of Science and Technology

SERC Advanced Research Fellow.

The design, application and implementation of PARLOG, a parallel logic programming language.

CONSULTANCY

1981:
Developed Surgical Audit system for use on Apple II microcomputer for a Consultant Surgeon at Queen Mary's Hospital, London.

1984:
Consultant to Logic Programming Associates Ltd., including developing a DEC-10 Prolog "front end" to LPA's micro-PROLOG and sigma-PROLOG. This was sold as part of the micro-PROLOG and sigma-PROLOG packages.

1984:

Carried out a study for International Computers Ltd. (Study Team on Support of Fifth Generation Languages) on the implementation of PARLOG on the Abstract PROLOG Machine.

1985-86:

Consultant to Hewlett-Packard Laboratories, Bristol, in parallel logic programming.

1985-86:

Consultant to European Computer-Industry Research Centre (ECRC), Munchen, in parallel logic programming.

1986:

Consultant to Honeywell Computer Sciences Center, Minneapolis, in the implementation of PARLOG.

MISCELLANEOUS

Academic prizes

1978:	Imperial College First Year Scholarship
1980:	Imperial College Governors' Prize in Computing Science

Societies

1980 - present:

Voting Member of the Association for Computing Machinery.

Refereeing

1981:	1981 Conference on Functional Programming Languages and Computer Architecture
1984:	Journal of Logic Programming
1985:	International Journal of Parallel Programming
1986:	Third International Logic Programming Conference
1986:	IEEE Symposium on Logic Programming
1986:	Addison-Wesley Publishers, Ltd.

1986:   Journal of Logic Programming

Other activities

Member of program committee for Fourth IEEE Symposium on Logic Programming, 1987.