A Report on my Visit to ICOT July 7 - August 25, 1991

Johann M. Ph. Schumann
Intellektik
Forschungsgruppe künstliche Intelligenz
Institut für Informatik,
Technische Universität München
Germany

1 Introduction

I had been kindly invited by Dr. Fuchi to stay at ICOT in July and August 1991. My main activities which will be described in more detail below has been the presentation of PARTHEO, the development and implementation of a top-down theorem prover in the language KL1 as well as many interesting discussions with researchers at ICOT about their work and about theorem proving.

Furthermore, I had the possibility to write two Technical Reports [Sch91b, Sch91a] on the work I have done during my stay.

2 Presentation of PARTHEO

Prof. Loveland and I gave a presentation on the Model Elimination Calculus and its (parallel) implementation at a joint working group meeting. Prof. Loveland focused on the calculus and METEORs, a high performance theorem prover developed at Duke University [AL91].

The topic of my presentation was "PARTHEO: A High Performance parallel Theorem Prover" (with [SL90] as a hand-out). PARTHEO is an OR-PARallel THEOrem prover based on the Model Elimination Calculus. It has been implemented on a network of (currently) 16 transputers and an intel Hypercube IPSC/II, both architectures featuring distributed memory and message passing. PARTHEO has been built on top of our sequential theorem prover SETHEO [LSBB90] which uses abstract machine technology, semi-compilation, and which has several built-in methods for pruning the search space. Such an abstract machine is running on each processor node of PARTHEO. OR-parallelism can be exploited by distributing the OR-search tree which is spanned by the formula, among the processors.

Each node of the search-tree and one of its possible continuations comprise a *task*, the basic piece of data which is exchanged between the processors. Each processor executes such tasks, and tries to extend them by applying a Model Elimination Extension or Reduction Steps. If a new node in the OR-tree is encountered, i.e. there is more than one possibility to proceed, new tasks are generated. These tasks are kept in the local memory of the processor, and are transmitted only on *request*. This model, the *task stealing model*, allows for a good load-balance and fast distribution of the work-load without the need of a global control unit. To further reduce the amount of data to transmit, the tasks are encoded and will be reconstructed by recalculation at the receiving processor.

With this architecture, very good speed-up results could be obtained with many examples (e.g. 15.09 with 16 processors (queens example)). Further points of discussions have been methods for pruning the search space, as well as possibilities and problems on generating and using lemmata.

3 KPROP: A Theorem Prover Implemented in KL1

During my stay at ICOT, I had the opportunity to implement a top-down theorem prover in the parallel language KL1. Based on my experience with SETHEO, this prover uses Model Elimination as its calculus, and is capable of proving theorems of full first order *propositional* logic. The restriction to propositional logic made the KL1 program much shorter (it has been my first KL1 program), and also allowed to implement some powerful pruning methods as well as a rather simple exploitation of AND-parallelism.

Similar to the approach in [Let88], and the PTTP [Sti89], I am using a compiling approach: each clause of the input formula (in Conjunctive Normal Form) is compiled into (several) KL1 clauses. Additionally, only a few predicates are needed to complete the theorem prover. The AND-parallel model of KPROP is directly based on the prinicple of KL1 which allows a concurrent execution of the body-goals of a clause. The work of solving the subgoals of each input clause is done in parallel, and synchronisation is accomplished by combining the results ("subgoal solved" or "subgoal not solved").

In order to increase the efficiency of this theorem prover, a powerful pruning method for reducing the size of the search spaces has been implemented, similar to that found in e.g. SETHEO. In regular intervals it is checked, if the current node in the Model Elimination tableau has an ancestor with an identical literal as value. If such a node is found (this check is rather cheap in propositional logic), this branch is considered to be failing, and a backtracking step is made. With this rather straight-forward implementation, very good results could be obtained which compare (or are even better) to other implementations of high performance theorem provers. [Sch91b] describes the theorem prover in detail and shows the results of a number of experiments with commonly used benchmark examples.

The technique of compilation and the usage of Model Elimination Calculus easily fits into the execution scheme of the language KL1. Especially the possibility for rapid prototyping in KL1 and the very useful tools for visualisation (profiler), as well as the kind support of many researchers at ICOT helped me to quickly overcome first difficulties in using KL1 - mainly due to its differences to PROLOG, and due to the lack of a powerful and easy to use parallel debugger.

Additionally to KPROP, I implemented some first prototype of a Model Elimination theorem prover for predicate logic (see also below), but unfortunately, not enough time was left to do further implementation work, and to make measurements.

4 Combining Top-Down and Bottom-Up Theorem Proving

With Dr. Hasegawa and Dr. Fujita I had some discussions about the combination of top-down and bottom-up theorem proving to increase the efficiency of theorem provers. The major source for inefficiency of top-down theorem provers based on the Model Elimination Calculus is that they have to proof the same subgoal over and over again, thus introducing a lot of redundant computation. On the other hand, bottom-up theorem provers lack a "goal directedness".

In his paper [Sti91], Prof. Stickel proposes a Meta-interpretation of the top-down proof procedure by a bottom-up system. This approach is to incorporate the more flexible search strategies, bottom-up systems have. I reported about some work done at the Intellektik Group (Prof. Bibel) in Munich [BLS87]. In this case, data-base technology is employed to do some bottom-up preprocessing steps. Based on this idea, I proposed a simple connection of a top-down theorem prover and MGTP (bottom-up): The MGTP is running as a *preprocessing module* and deduces new single-literal ground clauses ("facts") from the given formula. Hereby the number of allowable levels of resolution is restricted. After that run, the newly generated clauses (after doing some subsumption testing) are added to the original formula, which then is input to the top-down theorem prover. This prover then tries to find a solution, using a depth-first search with iterative deepening.

A number of first experiments which have been carried out by using SETHEO [LSBB90] as the top-down theorem prover showed astonishing results. In some cases, improvements of more than a factor of 5000 could be obtained. This may be due to the following two facts:

1. the preprocessing step can be done in a rather short time, since MGTP is well suited to generate ground atoms.
2. for finding a top-down proof, the search tree has not to be searched as deeply as without the preprocessing step. Even with the additional facts which increase the breadth of the search-tree, in many cases, the search space which has to be explored seems to be smaller. This especially true in cases, where the search space explodes exponentially.

Based on these results, which are described in detail in [Sch91a], I had some discussions with Yuasa-san of Waseda University, who will implement such a combined system in KL1. As the bottom-up theorem prover, MGTP will be used, the top-down theorem prover will be based on the Model Elimination Calculus. It will employ many ideas of KPROP, and the prototype of the unification routine, I have implemented in KL1.

5 Discussion with Researchers

5.1 Overview of FGCS Project

During ~~my~~ stay, I had the possibility to learn about some of the many projects going on at ICOT. Dr. Uchida gave Prof. Loveland and me a very detailed and interesting overview of the Final Stage of the Fifth Generation Project. Of special interest to me has been the presentation of the design of the PIM. The close cooperation at ICOT between hardware designers and developers of the KL1 language seems to me a substantial prerequisite for the design of a powerful parallel system, in contrast to many other approaches, where only a hardware, or only a parallel language exists. Furthermore, the plan of building 5 different PIM's with different architectures and processors, but with the common programming language KL1 opens up best possibilities for a extended study on parallel execution.

I also have been deeply impressed by the large number and variety of different applications. The possibility for a wide-spread usage of the programming language KL1 and the underlying system is largely facilitated by the remote access of the machines at ICOT -- although I think there is still a need for further manuals and tutorials of KL1 written in english.

Furthermore, the combination of PSI or PIM with UNIX/OSF as done at ICOT seems to me essential for a widespread commercial use of this work.

5.2 Theorem Proving

The MGTP theorem prover, developed by Dr. Hasegawa und Dr. Fujita seems very interesting and promising to me. As an extension of SATCHMO, the theorem prover developed at the ECRC, this kind of bottom-up theorem prover seems to be especially appropriate to be executed by KL1. The generation of ground models can be handled very efficiently in KL1 which has pattern matching as its basic method of parameter transfer.

The parallel model of execution of the MGTP developed by Fujita-san shows very good speed-up results which are also due to his very interesting and quite general scheme of distributed load balancing.

Furthermore, a version of MGTP has been developed, which can handle non-ground models. In this case, an explicit unification routine (e.g. from the "meta-libtary") has to be used, since logical variables cannot be directly mapped to KL1-variables. A compiled approach, as well as the usage of vectors in KL1 for the management of variables may increase the efficiency of the unification routines.

5.3 Constraint Satisfaction

In a meeting with Dr. Aiba and Dr. Menju, I have been shown a demonstration of constraint satisfaction algorithms. To me, the paradigm of constraint programming is very interesting, partly because constraint satisfaction and theorem proving are in some kind very similar to each other. I have been very impressed by the speed of the algorithm for solving boolean constraints, which has been developed by Menju-san. Beside for programming, constraint satisfaction seems to be a very valuable tool for hardware verification (or generation of test patterns). Some other, very interesting application could be the combination of constraint satisfaction with theorem proving. Many problems can be solved with the help of constraint satisfaction quite easily, whereas other theorem provers are often incapable of finding a solution. This, e.g. can be seen with many "puzzles" problems. Furthermore, with constraint satisfaction, relations between the variables are detected. Therefore, a whole set of solutions can be obtained, given as a set of equations, whereas a theorem prover has to search for each solution individually.

5.4 Higher Order Type theory

With Dr. Hagiya of RIMS in Kyoto, I had a very interesting discussion about theorem proving and his work on higher order type theory. Especially the application to software reuse is of great interest to me, since at the Intellektik Group, there is a joint project with Siemens about software reuse. In this project, our theorem prover SETHEO will be used to retrieve software models from a library, with the specifications given in first order predicate logic. I thank ICOT for the possibility to make this visit.

Acknowledgements

I would like to thank Dr. Fuchi for his kind invitation to visit ICOT. Dr. Furukawa and Dr. Hasegawa have contributed very much to a extremely stimulating and exciting stay at the 5th laboratory. Thank you very much! Many thanks are due for Fujita-san with whom I had many discussions, and who was very helpful in solving every problem I had, not only in the office, but also in planning my weekend outdoor activities to the Japanese Alps. Special thanks to Iwata-san for organising all the many non-technical aspects and thus making my stay very enjoyable and smooth. Of special value for me has been the invitation to the home of the Furukawas where we enjoyed an excellent Japanese dinner and true Japanese hospitality. This visit to ICOT which also has been my first stay in Japan has been a tremendous possibility not only to learn about the current status of ICOT research, but also to have many discussions and work together with strongly motivated researchers in a very inspiring athmosphere which is highly productive in new ideas in so many areas of computer science.

References

[AL91] O. L. Astrachan and D. W. Loveland. Meteors: High performance theorem provers using model elimination. Technical Report CS-1991-08. Dept. of CS. Duke University. Durham, North Carolina, 1991.

[BLS87] W. Bibel, R. Letz, and J. Schumann. Bottom-up Enhancements of Deductive Systems. In I. Plander, editor. *Artificial Intelligence and Information-Control Systems of Robots 87*, pages 1-9. North-Holland. 1987.

[Let88] R. Letz. Expressing First Order Logic within Horn Clause Logic. Technical Report FKI-96-c-88, Technische Universität München. 1988.

[LSBB90] R. Letz, J. Schumann, S. Bayerl, and W. Bibel. SETHEO: A High-Performance Theorem Prover. Technical report, Technische Universität München, 1990. To appear in Journal of Automated Reasoning.

[Sch91a] Johann Schumann. Combining top-down and bottom-up theorem proving: First experiments. Technical report, ICOT, Tokyo. Japan, 1991.

[Sch91b] Johann Schumann. Kprop: A model elimination theorem prover for propositional logic implemented in kl1. Technical report. ICOT. Tokyo, Japan, 1991.

[SL90] J. Schumann and R. Letz. PARTHEO: a High Performance Parallel Theorem Prover. In *CADE10*. Springer, 1990.

[Sti89] M. E. Stickel. A Prolog Technology Theorem Prover: a new Exposition and Implementation in Prolog. Stanford. 1989.

[Sti91] Mark E. Stickel. Upside-Down Meta-interpretation of the model elimination theorem proving procedure for deduction and abduction. Technical report, 1991.

[STK90] J. Schumann, N. Trapp, and M. van der Koelen. SETHEO/PARTHEO: User's Manual. Technical Report TUM-I 9010. 342/7/90 A, Technische Universität München, SFB 342, 1990.

Curriculum Vitae

Johann M. Ph. Schumann

Feb. 3,1960 born in München, Germany

1986 Diploma in Computer Science, with minor subject: Electrical Engineering, Technische Universität München.

1986-1991 Reseach Assistent at the Intellektik: Forschungsgruppe künstliche Intelligenz, Dept. of Comp. Science, Technische Universität München.

May-June 1988 Visiting Researcher, University of British Columbia, Vancouver, Cananda.

1991 Submission of Doctoral Thesis "Efficient Theorem Provers Based on an Abstract Machine"

Research interests:

Theorem proving, parallelism, neural networks, operating systems

List of Publications:
(unless listed above)

Bayerl, S., Eder, E., Kurfeß, F., Letz, R., and Schumann, J. (1987). An implementation of a Prolog-like theorem prover based on the connection method. In Jorrand, Ph., and Sgurev, V. (ed.) *AIMSA 86*, North Holland.

Bayerl, S., Letz, R., and Schumann, J. (1989). PARTHEO: A Parallel Inference Machine. In Brauer, W., and Freksa, C. (ed.). *Wissensbasierte Systeme*, Informatik Fachberichte 227, Springer.

Bibel, W., Kurfeß, F., Aspetsberger, K., Hintenaus, P., and Schumann, J. (1987). Parallel inference machines. In P. Treleaven, M. V., editor, *Future Parallel Computers*, LNCS 272, pages 185-226. Berlin. Springer.

Ertel, W., Kurfeß, F., Letz, R., Pandolfi, X., and Schumann, J. (1989). PARTHEO: A Parallel Inference Machine. In *PARLE '89*.

Kurfeß, F., Pandolfi, X., Belmesk, Z., Ertel, W., Letz, R., and Schumann, J. (1989). PARTHEO and FP2: Design of a parallel inference machine. In Treleaven, P. (ed). *Parallel Computers: Object-Oriented, Functional and Logical*, chapter 9. Wiley.

Schumann, J., Ertel, W., and Suttner, C. (1989). Learning Heuristics for a Theorem Prover using Backpropagation. In *ÖGAI '89*, Springer.

Schumann, J., Trapp, N., and van der Koelen, M. (1990). SETHEO: Users Manual. Technische Universität München, SFB-Bericht Nr. 342/7/90A.

Schumann, J., Letz, R., and Kurfeß, F. (1990). High Performance Theorem Provers
Efficient implementation and Parallelisation Tutorial on *CADE '90*.

Johann M. Ph. Schumann
Waldeslust 4
8000 München 70

Curriculum Vitae

Febr. 3rd 1960
born in Munich, Germany. Parents: Gabriele and Heiner Schumann.

1966–1970
Elementary School, Munich .

1970-1979
Math. - nat. Rupprecht Gymnasium (Highschool with main subjects in Natural Science and Mathematics), Munich

1979 Abitur.

Fall 1980 - Spring 1986
Study of Computer Science with minor subject Electrical Engineering at the Technical University Munich.

Fall 1982 Vordiplom (Bachelor degree)

Spring 86 Diploma Degree
Diploma Thesis: "Portable Software Monitor and Debugger for Microprocessors with Bootstrap Loader and basic Functions for Multiprocessing".

June 1986 – May 1991
Researcher at the AI Research Group ("Intellektik") at the Technical University Munich.

June 1986 – Nov. 1989
Researcher in Project ESPRIT 415F: "Advanced Information Technology: A VLSI Oriented Approach: Logical Connection Machine".

Jan. 1990 – May 1991
Researcher at the "Sonderforschungsbereich 342"
Subproject A5: "PARIS: Parallelisation of Inference Systems".

May 1991
Submission of PhD Thesis: "Efficient Theorem Provers based on an Abstract Machine".

[illegible]