LAST STEPS TOWARDS AN ULTIMATE PROLOG

A. Colmerauer, H. Kanoui, M. Van Caneghem

Groupe d'Intelligence Artificielle
Faculte des Sciences de Luminy
Universite d'Aix-Marseille II
13288 Marseille Cedex 9,France

ABSTRACT

A portable version of Prolog, an Artificial Intelligence language, is presented. A complete system has been implemented on a micro-computer using a floppy-disk virtual memory. The general methodology of the implementation is discussed in terms of *an* abstract machine (Micromegas) supporting a language (Candide) in which the Prolog system is written.

I INTRODUCTION

Prolog has been developed in our research laboratory in 1973 by A. Colmerauer and P. Roussel [1.2,10], mainly for natural language processing systems. Since then, it has been applied to various A.I. fields. Several Prolog interpreters have been written, and even a compiler by D. Warren for a DEC 10 computer. Practice and theory of Prolog have been also studied and extensively described (see [3,5,6,8]). Amazingly, the influence of Prolog has been until now restricted to Europe, where the language has been largely distributed and used. However, some recent papers in the litterature show a growing interest from the U.S. community [9,11].

II MOTIVATIONS

During the past years, Prolog users clamored for more facilities to write very large programs. For these reasons, we decided to start a new general version required to be usable on large computers as well as on micro-computers. Our first try in that direction was the design of a Prolog system on an { xorciser M6800 micro-computer. This work is reported in [4,7] and gave encouraging results, specially in the use of floppy disks for virtual memory management.

The Prolog system reported here is intended to be the "ultimate" one in the sense that it is specially designed to fullfil the requirements of :

1. Easy transportability.

2. Running large applications thanks to a large memory space dynamically organized as a tree structured hierarchy of sub-worlds to achieve software protection and management of memory space.

3. Versatility by a practical tool to add user-oriented evaluable predicates.

4. Powerful new built-in features such as coroutines, infinite-trees processing [5] , error recovering and user controlled execution thanks to the concept of "block".

5. Interactivity by including clause editing capabilities.

III REALIZATION

We exhibit a complete, self-contained system which includes :

1. A high level programming language, Candide, designed to provide the software tools needed to write the interpreter.

7. A candide compiler which emits code for an abstract machine, Micromegas. featuring the hardware viewpoint of the system.

3. A Micromegas emulator which interprets the Micromegas code at run-time. This is the only part which depends on the host machine on which the Prolog system is implemented.

4. The Prolog interpreter itself, which is the heart of the system, and consists of 50 pages of Candide code.

A complete, fully documented version of this system is operational on Apple II. The implementation on a Solar 16-65 (french minicomputer-equivalent to PDP 11-65) is on progress and will be done by September 81.

The abstract machine Micromegas is embedded into the operating system of the host computer, leaving free access to the built-in resources (e.g. in the Apple II implementation, the U.C.S.D. Pascal system procedures and functions can be reached from the Micromega3 level).

The abstract machine uses cells and words : a cell is the smallest part of memory addressable both by the host computer and by the Micromegas machine. A word is a sequence of contiguous cells containing a large address : in the Apple II implementation, we have 8-bit colls and 24-bit words.

Thin provides an addressing space of 2 megabytes, simulated hy a virtual memory management system Implemented hy software on a floppy disK. The Micromegas machine has also 256 general purpose registers and a user memory which contain", the pro gram and a local variable stack.

The Micromegas instructions perform arithmetic, transfers, comparisons. The operands are cells or words from the virtual memory, the registers and the stack. Other instructions perform conditional, unconditional or computed gate's, procedure invocation and return, external procedure activation, and so on.

IV PROLOG ON APPLE II

This implementation needs :

1. An Apple II with 48 K bytes of memory.

2. Two 143 K bytes floppy disks.

3. The language card.

The Micromegas emulator is embedded into the Apple Pascal system and consists in 10 pages of Pascal code (Initialisation, I/O, and virtual memory management) plus 1b pages of 6502 assembler code (performing the 16 instructions of the Micromegas machine). The Micromegas code for the Prolog interpreter needs 12 K bytes. The virtual memory system in implemented on the two floppies and the main memory paging area occupies 40 256-byte pages. Such a configuration allows up to 50-pages Prolog programs.

V CONCLUSION

The performance of a micro-computer implementation Is slow compared with a large computer one. Nevertheless, the system remains usable thanks to the large virtual memory. A micro-computer implementation can be used hy A. I. students, eleven by researchers for testing and debugging programs, thus performing (at home) on a personnal computer the more tedious and time consuming part of programming work. Naturally, the definitive application will be run on a bigger computer.

Is it really the ultimate Prolog ? Certainly yes for us !

In fact, we have deliberately spent three years to design the system and build the adequate tools to perform our research. This goal is now achieved and we have definitely moved back to the Prolog users family.

ACKNOWLEDGE MENTS

This work has been sponsored by the Centre National do la Recherche Scientifique (Iquipo do Recherche Associated 363) .

REFERENCES

[1] Battani G. and Melon! H.. "Interpret our du langage de programmation Prolog,", internal report. Croupe d'Intelligences Artificial, Universite Aix-Marseilie II. September 19/3.

[2] Colmerauer A., Kanoui P., Pasero P. and Roussel Ph., "Un systeme de communication homme-machine en franca!s", internal report. Croupe d'Intellengence Artificielle. Universite Aix-Marseillo II. June 1973.

[3] Colmerauer A., "Metamorphosis Grammars". Natural language communication with computer. Springer Verlag. 1978.

[4] Colmerauer A.. Kanoui H., and Van Caneghem M.. "Etude et realisation d'un systeme Prolog", Internal report, Groupe d' Intelligence ArtJficielle. Universite Aix-Marseille II. May 1979.

[5] Colmerauer A., "Prolog and infinite trees", to appear in Logic Programming Workshop 1 (K. Clark and S.A. Tarnlund. eds).

[6] Kowalski P. and Van Emdeo M.. "The semantics of predicate logic as a programming language". Journal of the ACM. 23:4. October 1976. 733-742.

[7] Kanoui H. and Van Caneghem M.. "Implementing a very high level language on a very low cost computer", in prne. IF IP 80. October 1980. 349-354.

[0] Kowalski R. , "logic for problem solving,". North Holland. 1979.

[9] Mr. Dermott.The Prolog phenomenon. Sigart Newsletter, n. 72. July 1980). 16-20.

[10] Roussel Ph.. "Manuel d' utilisation Prolog,", internal report, Groupe d' Intelligence Artificielle, Universite Aix-Marse111e II. September 1976.

[11] Comments on Prolog. Sigart Newsletter, n. 73, October 1980. 10-25.