

A.E. COSTA PEREIRA *

M.C. MONARD **

R.ALLAN ***

SUMARIO

A Programação Lógica possui características que tornam mais fácil e rápido o desenvolvimento de programas, mas a maioria dos interpretadores e compiladores PROLOG não oferecem as ferramentas necessárias para o processamento de dados numéricos e para o desenvolvimento de pacotes gráficos. Nesta comunicação, mostramos algumas extensões que estamos implementando num interpretador PROLOG a fim de capacitá-lo para este tipo de aplicações.

ABSTRACT

Logic Programming has features which make the development of programs easy and fast but most of PROLOG interpreters and compilers do not provide the necessary tools for processing numerical data and for developing graphic packages. In this communication we show some extensions we are implementing in a PROLOG interpreter to make it suitable for this sort of applications.

* Bacharel em Física (IF-USP,1971); Eng. Eletrônico (EP-USP,1971); MSc(INPE-1975); PhD (Univ. Cornell-USA,1979); Livre Docente (UNESP-Rio Claro, 1983). Inteligência Artificial, Programação Lógica, Sistemas Especialistas.

** MSc(Univ. Southampton-Ingl.,1969); Dr. Informática (PUC-RJ,1980). Estrutura de Dados e Algoritmos, Programação Lógica
USP-ICMSC. Depto. Computação. Av. Dr. Carlos Botelho, 1465. CEP 13560. São Carlos - SP.

*** Bacharel em Computação (USP-São Carlos, 1983); Atualmente Aluno de Mestrado em Ciência da Computação no INPE. Inteligência Artificial, Programação Lógica.

ta estrutura.

No caso do problema de computação gráfica, o uso do "backtracking" de PROLOG torna-se muito interessante. Ao efetuar a expansão na árvore de prova, vai-se desenhando a figura e, no caso de acontecer "backtracking", os pontos traçados podem ser apagados e uma outra figura é tentada.

Não é difícil implementar "turtle graphics" usando o "backtracking" de PROLOG. Para isto é suficiente armazenar na pilha informações de cada segmento de linha que é desenhado e armazenar os valores correntes das variáveis. Quando acontecer "backtracking" estas informações são usadas para apagar estes segmentos de linha.

Achamos que isto é bastante prático para o pessoal ligado a CAI e CAD. Aproveita-se o não determinismo para gerar figuras até chegar a alguma que satisfaz as condições necessárias.

CONCLUSÕES

Com estas duas extensões acreditamos que a programação lógica para processos numéricos e gráficos possa ser facilitada diminuindo o tempo de desenvolvimento de programas e o tempo de execução dos mesmos.

REFERÊNCIAS

- 1 - Boyer, R.S.; Moore, J.S - The Sharing of Structure in Theorem Proving Programs. Machine Intelligence 7. Edimburgh University Press. 1972.
- 2 - Bruynooghe, M. - The Memory Management of PROLOG Implementations. Logic Programming (Eds. Clark K.L.; Tarnul S.A.). Academic Press, New York, NY. pp. 84-98, 1982.
- 3 - Van Emden, M.H. - An Algorithm for Implementing PROLOG Programs. University of Waterloo, 1980.
- 4 - Warren, D.H.D. - Implementing PROLOG-Compiling Logic Programs. DAI Research Report Nº 39-40. University of Edinburh, 1977.
- 5 - Grupo Computação Lógica: Detalhes da Estrutura de Dados e Implementação de Um Interpretador PROLOG. Documento Interno ICMSC-USP, 1985.