Abstract Interpretation in Pisa

From 1987 -:
The roots (and my roots)

November 1987: I was looking for a subject....

Giorgio gave me 2 papers:

  Abstract Interpretation or Partial Evaluation?
PARTIAL EVALUATION AS A MEANS FOR INFERRING DATA STRUCTURES IN AN APPLICATION LANGUAGE: A THEORY AND IMPLEMENTATION IN THE CASE OF PROLOG

R. Jan Komenda
Software Systems Research Center
Linkoping University
S-581 83 Linkoping, Sweden

ABSTRACT

An semantics of the Prolog programming language is introduced. We use this to specify the semantics of the purpose of the work is to provide an implementation of Prolog inferences. An operational meaning of this specification is in terms of partial evaluative strategies based on the principle of partial evaluation.

Transformations which account for partial evaluation of data structure propagation and opening (which also provides redundant data structure propagation) are formally introduced and proved. To prove reasoning about programs. The so introduced transformations provide a basis for inferring data structures in an application language. The specification is then roughly related to research in rule-based systems and logic.

An efficient well-integrated partial evaluation system is available in Oz (a Lisp programming environment for Prolog).

1.0 INTRODUCTION

It is very likely that a large part of future programming will be programming in increasingly higher level languages, in such languages more attention will be paid to efficient problem solving, whereas less efficiency need not reflect the requirements of an efficient computation. In such circumstances program transformations must play the central role in making the efficiency realistic. It is also felt that the tools should be interactive. One reason is that they are to support the user in the incrementally interactive activities of programming. The user can use the data structure propagator in the Oz system for inferring data structures in an application language.

The rest of the paper is organized as follows. First, an informal introduction of partial evaluation and partial evaluations of the Oz system according to the specification. Finally, a brief discussion of inferring data structures in rule-based systems follows.

2.0 PARTIAL EVALUATION: AN INTRODUCTION

An informal introduction to partial evaluation is presented here and illustrated with a simple example.

The goal of partial evaluation is to transform programs into more efficient ones. The improvement is obtained at the expense of the generality of the programs. The restrictions on generality are usually introduced by setting...
The choice....
The beginning....

Giorgio was of course already interested in semantics and correctness of symbolic interpreters!!!


**Introduction**

The semantics of programming languages has received a good deal of consideration because it is an essential part of the definition of a programming language and provides a sound basis for interpreter design. The interest in this field has been emphasized because of its relationship to proving properties of programs. Some recent results (Boyer and Moore [1,2], Burstall [3] and Popov [4]) have shown that interpreters can be extended to cope with the task of proving properties of programs. This task requires the (symbolic) interpreter to be able to deal with symbolic values (i.e. expressions containing quantified variable symbols) and to make use of induction rules.

We are concerned with the problem of defining general methods for generating symbolic interpreters for programming languages. Any such method must depend on a description of the programming language semantics providing a characterization of the language in terms of a suitable symbolic logic. In the paper we will introduce a calculus (Term Equation Language) and its symbolic interpreter. TEL has a straightforward logical interpretation. Programming language semantics is given by means of a set of TEL axioms which provide through the TEL interpreter a symbolic interpreter for the programming language.
...and Barbuti?

- This is where I have understood correctness of static semantics!
- ......simple and clean!
The beginning...


- The big (theological) deal: bottom-up or top-down?
The great days!!

- In Pisa in the 90s:
  - The best group in semantics of Logic Programming!
  - Strong mathematical bases!
  - Challenging projects!
  - International environment
  - Enthusiasm!
The Cousots exist!

ICLP’91: a joint workshop on Abstract Interpretation in Logic Programming ... in Paris!
and then: WSA, SAS, etc.
The Abstract Interpretation slice

- Dependence analysis
- Modelling Prolog Control
- CLP & Numeric domains
- Bottom-up Analysis
- Compositional Analysis
- Abstract Domain Theory
- Abstract Debugging
- Verification and Types
- Sharing & Freeness
- Groundness
- Numeric domains

Timeline:
- 1989
- 1992
- 1993
- 1994
- 1995
- 1996
- 1997
- 1998
- 1999
- .......
Abstract Interpretation and People

Giorgio Levi
Roberto Barbuti
Roberto Giacobazzi
  Michael Codish
  Michael Maher
Roberto Bagnara
  Saumya Debray
Enea Zaffanella
  Francesco Ranzato
Francesca Scozzari
  Agostino Cortesi
Catuscia Palamidessi
  Moreno Falaschi
  Gilberto Filè
  Giuliana Vitiello
Maria Chiara Meo
  Marco Comini
  Paolo Volpe
Roberta Gori
  Fausto Spoto
Gianluca Amato
Francesca Levi
Sergio Maffeis
  Patricia M. Hill
  Francois Fages
Spreading ideas and people
The chance of being in Pisa