INSTITUT FÜR INFORMATIK
Lehr- und Forschungseinheit für
Programmier- und Modellierungssprachen
Oettingenstraße 67, D–80538 München

Ludwig
Maximilians–
Universität
München

SatAM – Satchmo Abstract Machine (System Description)

Alexander von Drach, Tim Geisler, Sven Panne, David Sacher

In: Proc 13th Workshop Logische Programmierung (WLP '98), TU Wien, Oct. 1998
http://www.pms.informatik.uni-muenchen.de/publikationen
Forschungsbericht/Research Report PMS-FB-1998-9, May 1998

SatAM – Satchmo Abstract Machine (System Description)

Alexander von Drach, Tim Geisler, Sven Panne, and David Sacher

Institut für Informatik, Universität München, Oettingenstr. 67, D-80538 München
{vondrach|geisler|panne|sacher}@informatik.uni-muenchen.de

1 Overview

The Satchmo Abstract Machine (SatAM) is a combined bytecode compiler and interpreter system for the efficient generation of Herbrand models for first-order formulas. The compiler translates an input specification given as a set of first-order formulas into bytecode for an abstract machine [4]. The generated bytecode program implements the generation of models using the PUHR tableau method [1] for the given input formulas. The bytecode interpreter executes the bytecode program and generates models. The data structures and the dynamic behavior of the bytecode interpreter can be visualized with the graph visualization tool daVinci [2]. This is extremly useful for debugging the implementation and for explaining its concepts. The compiler is implemented portably in the purely functional language Haskell, the interpreter is implemented portably in C++. The whole system runs on Unix and Microsoft Windows 95/98/NT.

2 The Compiler

The SatAM compiler generates a bytecode file from a set of first-order formulas. These formulas are written in a syntax similar to the one used in the theorem prover Otter [3] with some minor deviations allowing several common and convenient notations for clauses and first-order formulas.

In a first step, the compiler translates its input formulas into range-restricted clauses by Skolemizing and introducing additional literals to non-range-restricted clauses [1]. The resulting range-restricted clauses are compiled into a bytecode program for the abstract machine, which implements the generation of models for these clauses. The structure of these bytecode programs is described briefly in [4]. Bytecode programs are written in a binary file format, which is machine independent and thus portable between different hardware architectures and operating systems.

The compiler offers various command line options, which allow to output the results of all compilation phases.

3 The Interpreter

The SatAM interpreter efficiently executes the bytecode programs and generates models of the formulas processed by the compiler. The interpreter outputs each generated model: it prints the ground atomic formulas that hold in the model.

The interpreter offers various command line options, which enable tracing, single-stepping, viewing the machine state, and printing of several machine statistics.

4 Visualization of the Abstract Machine

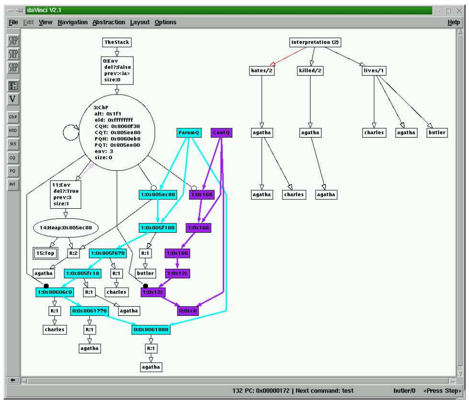

Fig. 1. GUI of SatAM using daVinci.

The interpreter can be connected with the graph visualization system daVinci [2]. This graphical user interface (GUI) to the abstract machine visualizes the current machine state and enables the user to control the machine flow by single-stepping through the abstract-machine code.

It is possible to view all or some selected data structures of the machine in a graph representation with more or less detail (Fig. 1). Furthermore, entries to the machine stack can optionally be visualized as stack frames.

References

1. François Bry and Adnan Yahya. Minimal model generation with positive unit hyper-resolution tableaux. In P. Miglioli, U. Moscato, D. Mundici, and M. Ornaghi, editors, *Proc. TABLEAUX '96*, number 1071 in Springer LNCS, pages 143–159, Terrasini, Palermo, Italy, May 1996. Springer-Verlag.
2. Michael Fröhlich and Mattias Werner. The graph visualization system daVinci – A user interface for applications. Technical Report 5/94, Institut für Informatik, Universiät Bremen, September 1994.
3. William W. McCune. Otter 3.0 reference manual and guide. Technical Report ANL 94/6, Argonne National Laboratory, 1994.
4. Alexander von Drach, Tim Geisler, Sven Panne, and David Sacher. An abstract machine for model generation with PUHR tableaux (extended abstract). In Uwe Egly, editor, *13. Workshop Logische Programmierung, WLP '98*, Technische Universität Wien, Wien, Austria, October 1998.