Constraint Logic Programming in SWI-Prolog

Markus Triska

Technische Universität Wien

Considering that SWI-Prolog has so far been in continuous development for more than 20 years, its support for Constraint Logic Programming is still comparatively new: In 2004, Jan Wielemaker added support for hProlog-style attributed variables to SWI-Prolog, with help from Bart Demoen. Attributed variables are one way to support libraries for Constraint Logic Programming. Some of these libraries were already included in the then current stable version of SWI-Prolog (5.4.0): CHR, the K. U. Leuven library for Constraint Handling Rules, a constraint solver over finite domains, and `dif/2` and `when/2` constraints.

In 2005, Jan added support for rational numbers and unbounded integers to SWI-Prolog. SWI-Prolog's first constraint library that used rational numbers was Leslie De Koninck's port of Christian Holzbaur's CLP(Q/R) system, included in SWI-Prolog since version 5.6.0.

Also in 2005, Mats Carlsson – the designer and main implementor of SICStus Prolog – generously posted an elegant CLP(FD) formulation of the so-called "Social Golfer Problem" to `comp.lang.prolog`. It used (among others) the constraints `table/2`, `all_distinct/2` and truncated integer division, which were not available in SWI-Prolog's constraint libraries at that time.

To improve SWI-Prolog's compatibility with SICStus Prolog for Mats's program and others like it, I contributed truncated integer division and `table/2` to its finite domain constraint solver. `all_distinct/1` however needed a different domain representation to efficiently achieve strong propagation.

In 2007, I started to work on `library(clpfd)`, a new finite domain constraint solver that is included in SWI-Prolog since version 5.6.40. It supports unbounded integers, and global constraints like `all_distinct/1`, `automaton/3`, `circuit/1` and `global_cardinality/2` for SICStus compatibility. Also, constraint propagation and `labeling/2` *always* terminate, which is of interest for researchers working on termination analysis. I thank Ulrich Neumerkel for testing the library extensively.

The finite domain constraint solver also provides a rudimentary framework to let users implement application-specific constraints. Some users are already using this interface in their own projects.

As the constraint libraries of SWI-Prolog become more advanced, its support for attributed variables improves as well: SWI-Prolog now provides the SICStus-inspired `copy_term/3` and `call_residue_vars/2` predicates, which can be used to reason about constraint programs and are also used by the toplevel to obtain residual constraints.

The robustness of SWI-Prolog makes it a popular choice for teaching and learning Prolog, and its constraint libraries are already being used to teach

constraint logic programming at several universities in Austria, France, Germany, Japan and other countries.

Ongoing and planned additions to SWI-Prolog's constraint libraries include CLP(B) (a constraint solver over Boolean variables) and further global constraints for CLP(FD). Please let us know which features you want most!

1 Links

- CHR (Constraint Handling Rules): http://dtai.cs.kuleuven.be/CHR/
- CHR in SWI-Prolog: `http://www.swi-prolog.org/man/chr.html`
- CLP(FD) in SWI-Prolog: `http://www.swi-prolog.org/man/clpfd.html`
- CLP(Q/R) in SWI-Prolog: `http://www.swi-prolog.org/man/clpqr.html`
- CLP(FD) animations used in Prolog courses at several universities, implemented using SWI-Prolog and easily portable to other systems:
  - N-Queens: `http://www.logic.at/prolog/queens/queens.html`
  - Sudoku: `http://www.logic.at/prolog/sudoku/sudoku.html`
  - Closed Knight's Tour: `http://www.logic.at/prolog/knight/knight.html`