Constraint Handling Rules for SWI-Prolog

Tom Schrijvers*[1], Jan Wielemaker[2], and Bart Demoen[1]

[1] Department of Computer Science, K.U.Leuven, Belgium
[2] Human Computer-Studies Laboratory, University of Amsterdam, The Netherlands

System Description

Until recently SWI-Prolog had no built-in support for constraint solving. The positive experience with porting the K.U.Leuven CHR system [3] from hProlog to XSB Prolog [5], suggested that SWI-Prolog could be extended with CHRs with very little effort.

The K.U.Leuven CHR system relies only on the following non-standard low-level features: attributed variables as in [1], global backtrackable variables and support for cyclic terms. By copying the existing interface and implementation principles from hProlog and XSB, the effort was small indeed and at the same time the compatibility between these systems was realized. With the above three features in place, SWI-Prolog's CHR system was fully operational in no time and all CHR solvers are now available to SWI-Prolog users since SWI-Prolog's release 5.4.0 [6].

Experimental evaluation showed that thanks to the efficient implementation of the low-level features average CHR benchmarks behave better than average Prolog benchmarks when compared to SICStus Prolog [2] with its reference CHR compiler. Since SWI-Prolog's focus on a user friendly environment, the CHR compiler was tightly integrated with the `term_expansion/2` based preprocessor, so as to exploit SWI-Prolog's source-code management and to retain source information. We also added a CHR debugger which hides the underlying generated Prolog code from the user.

To the best of our knowledge, SWI-Prolog's CHR system is the first to support hashtable constraint stores and mode declarations. This results in considerably better performance and e.g. the Union-Find algorithm can be implemented with optimal complexity [4].

Thanks to the initial effort in low-level provisions for CHR, SWI-Prolog now also provides other typical CLP facilities: co-routining (`freeze/1` and `when/2`), a simple bounds-consistency finite domain solver and a port of Christian Holzbaur's CLP($\Re$) solver. It turns out that our minimal and de facto standard extension is adequate to bring full CLP support to a Prolog system.

In the future we intend to improve the portability and standardization of both the CHR system and general CLP solvers. Users should be free to use CLP capabilities in the Prolog system of their choice.

* Research Assistant of the fund for Scientific Research - Flanders (Belgium)(F.W.O. - Vlaanderen)

References

1. Bart Demoen. Dynamic attributes, their hProlog implementation, and a first evaluation. Report CW 350, Department of Computer Science, K.U.Leuven, Leuven, Belgium, oct 2002.
2. Intelligent Systems Laboratory. *SICStus Prolog User's Manual*. PO Box 1263, SE-164 29 Kista, Sweden, October 2003.
3. Tom Schrijvers and Bart Demoen. The K.U.Leuven CHR system: Implementation and application. In *First workshop on constraint handling rules: selected contributions*, pages 1–5, 2004. Published as technical report: Ulmer Informatik-Berichte Nr. 2004-01, ISSN 0939-5091.
4. Tom Schrijvers and Thom Frühwirth. Implementing and Analysing Union-Find in CHR. Report CW 389, K.U.Leuven, Department of Computer Science, jul 2004. Summitted to Theory and Practice of Logic Programming.
5. Tom Schrijvers and David Warren. Constraint handling rules and tabled execution. In *Logic Programming, 20th International Conference, ICLP 2004, Proceedings*, volume 3132 of *LNCS*, pages 120–136. Springer Verlag, 2004.
6. Jan Wielemaker. SWI-Prolog release 5.4.0. http://www.swi-prolog.org/.