$\mathcal{TOY}$: A CFLP Language and System*

I. Castiñeiras, J. Correas, S. Estévez-Martín, and F. Sáenz-Pérez

June 21, 2012

$\mathcal{TOY}$ is a Constraint Functional Logic Programming (CFLP) language and system [10, 11] that solves goals by means of a demand-driven lazy-narrowing strategy combined with constraint solving. Constraints are integrated as first class citizens in a language that intermixes features of Logic and Functional Programming: Logical variables, relational notation, non-determinism, backtracking, domain variables, functional notation, curried expressions, higher-order functions, patterns, partial applications, lazy evaluation, types, polymorphism and constraint composition. This language relies on strong mathematical foundations [9]. As a system, it is multiplatform (as long as supported by SICStus), open-source, distributed under GPL, has been interfaced to the ACIDE GUI, and enjoys declarative debugging [4] including totality constraints.

$\mathcal{TOY}$ core system supports several constraint domains which are built with SICStus technology: $\mathcal{H}$ for symbolic equations and disequations [2], $\mathcal{R}$ for arithmetic constraints over the real numbers [3], and $\mathcal{FD}$ for finite domain constraints [8]. Furthermore, $\mathcal{TOY}$ enjoys a cooperation mechanism between solvers [7, 6]. The most related CFLP language is Curry and its implementations, as PAKCS [1] which also runs on SICStus. However, this system does not support constraint cooperation.

$\mathcal{TOY}$ has been also interfaced to external constraint solvers: (1) $\mathcal{FD}$ and $\mathcal{FS}$ (finite sets) from ECL^{i}PS^e, (2) $\mathcal{FD}$ from IBM ILOG Solver, and (3) $\mathcal{FD}$ from Gecode. Each of these interfaces has lead to a different implementation (cf. [11]). There are two communication mechanisms with external solvers: For the first interface [6], ECL^{i}PS^e runs as a separate process and a simple communication protocol is built based on standard pipes. For the las two interfaces [5], the C++ interface as provided by each external solver is used to build a single application. In these systems, both batch (for pure CP applications) and online (for model reasoning) constraint solving are allowed.

Current trends for future work include improving performance for tackling real-life problems which are amenable to be modeled with constraints and their combinations in different domains.

*This work has been partially supported by the Spanish projects STAMP (TIN2008-06622-C03-01) (first, second and fourth authors), DOVES (TIN-2008-05624) (third author), Prometidos-CM (S2009TIC-1465), and GPD (UCM-BSCH-GR35/10-A-910502).

References

[1] S. Antoy and M. Hanus. Compiling Multi-Paradigm Declarative Programs into Prolog. In *FroCoS*, volume 1794 of *LNCS*, pages 171–185. Springer, 2000.

[2] P. Arenas-Sánchez, A. Gil-Luezas, and F. López-Fraguas. Combining Lazy Narrowing with Disequality Constraints. In *PLILP'94*, volume 844 of *LNCS*, pages 385–399. Springer, 1994.

[3] P. Arenas-Sánchez, M.T. Hortalá-González, F. López-Fraguas, and E. Ullán. Real constraints within a functional logic language. In Paqui Lucio, Maurizio Martelli, and Marisa Navarro, editors, *APPIA-GULP-PRODE*, pages 451–464, 1996.

[4] R. Caballero. A declarative debugger of incorrect answers for constraint functional-logic programs. In *WCFLP '05: Proceedings of the 2005 ACM SIGPLAN workshop on Curry and functional logic programming*, pages 8–13, New York, NY, USA, 2005. ACM Press.

[5] I. Castiñeiras and F. Sáenz-Pérez. Improving the Performance of FD Constraint Solving in a CFLP System. In *FLOPS'12*, volume 7294 of *LNCS*, pages 88–103. Springer, 2012.

[6] S. Estévez-Martín, J. Correas Fernández, and F. Sáenz-Pérez. Extending the TOY System with the ECLIPSE Solver over Sets of Integers. In *FLOPS'12*, volume 7294 of *LNCS*, pages 120–135. Springer, 2012.

[7] S. Estévez-Martín, A.J. Fernández, M.T. Hortalá-González, M. Rodríguez-Artalejo, F. Sáenz-Pérez, and R. del Vado Vírseda. On the Cooperation of the Constraint Domains $\mathcal{H}$, $\mathcal{R}$ and $\mathcal{FD}$ in $CFLP$. *TPLP*, 9:415–527, 2009.

[8] A.J. Fernández, M.T. Hortalá-González, F. Sáenz-Pérez, and R. Vado Vírseda. Constraint functional logic programming over finite domains. *TPLP*, 7(5):537–582, 2007.

[9] F.J. López-Fraguas, M. Rodríguez-Artalejo, and R. del Vado Vírseda. A new generic scheme for functional logic programming with constraints. *Higher-Order and Symbolic Computation*, 20(1/2):73–122, 2007.

[10] F.J. López-Fraguas and J. Sánchez-Hernández. $\mathcal{TOY}$: A Multiparadigm Declarative System. In *RTA*, volume 1631 of *LNCS*, pages 244–247. Springer, 1999.

[11] $\mathcal{TOY}$ Web Site. `http://toy.sourceforge.net/`.