Constraints, Predicates, Functions and a Turtle

Stephan Frank, Martin Grabmüller, Petra Hofstedt and André Metzner

Fakultät IV – Elektrotechnik und Informatik
Technische Universität Berlin, Germany
{sfrank,magr,ph,ame}@cs.tu-berlin.de

This abstract briefly introduces the current work on multiparadigm programming languages and systems at Berlin University of Technology.

Multiparadigm programming languages combine different programming paradigms, such as functional, logic, constraint or imperative ones. The idea of a multiparadigm language is to increase expressiveness and problem-solving power such that the programmer can use a wide range of styles and language features from different paradigms.

In [Hof01] a general approach for the combination of different constraint domains and the cooperation and coordination of constraint solvers is introduced. The mainly theoretically described framework is open for the integration of arbitrary constraint solvers and provides a flexible mechanism for defining cooperation strategies for the solvers. Conditions and properties of such collaborations with regard to soundness, completeness, termination and confluence have been discussed in detail in this work.

A practical implementation and extension of this theoretical framework is Meta-S [FM02,FHM03]. It provides an extensible strategy specification language, including pattern matching and constraint rewriting facilities, and serves as a test-bed for generic and problem-specific (meta-)solving strategies, which are employed to minimize the cooperation overhead incurred.

In [Hof02] we show how considering declarative programming as constraint programming and looking at the language evaluation mechanisms as constraint solvers enables to integrate these particular language solvers into the aforementioned cooperation framework and the smooth integration of constraints into these languages. This allows to build constraint languages customized for a given set of requirements for comfortable modeling and solving of many problems.

A language for such an integration is the functional logic programming language Brooks [Met02] which inherits from the languages Curry and BABEL, but in contrast allows the integration of different narrowing strategies. The smooth integration of constraints into Brooks using the ideas of [Hof02] is a future research topic.

A further work [Gra03] investigates the combination of constraint-based and imperative programming which is called constraint imperative programming. Grabmüller describes the design of the higher-order constraint imperative programming language Turtle which integrates moreover language constructs known from functional programming languages, like higher-order functions and polymorphic modules.

References

[FHM03] Stephan Frank, Petra Hofstedt, and Pierre R. Mai. Meta-S: A strategy-oriented Meta-Solver Framework. In *16th Int'l FLAIRS Conference*, 2003. to appear.

[FM02] Stephan Frank and Pierre R. Mai. Strategies for Cooperating Constraint Solvers. Diploma thesis, Technische Universität Berlin, July 2002.

[Gra03] Martin Grabmüller. Constraint-Imperative Programmierung. Diploma thesis, Technische Universität Berlin, February 2003.

[Hof01] Petra Hofstedt. *Cooperation and Coordination of Constraint Solvers.* PhD thesis, Technische Universität Dresden, 2001. Shaker Verlag, Aachen.

[Hof02] Petra Hofstedt. A general Approach for Building Constraint Languages. In B. McKay and J. Slaney, editors, *AI 2002: Advances in Artificial Intelligence*, volume 2557 of *LNCS*, pages 431–442. Springer-Verlag, 2002.

[Met02] André Metzner. Eine abstrakte Maschine für die (schrittweise) Abarbeitung funktional-logischer Programme. Diploma thesis, Technische Universität Berlin, July 2002.