# *Editorial: Constraint Handling Rules*

SLIM ABDENNADHER
*Department of Computer Science, German University Cairo, Egypt*
(*e-mail:* Slim.Abdennadher@guc.edu.eg)

THOM FRÜHWIRTH
*Faculty of Computer Science, University of Ulm, Germany*
(*e-mail:* www.informatik.uni-ulm.de/pm/mitarbeiter/fruehwirth/)

CHRISTIAN HOLZBAUR
(*e-mail:* holzbaur@chello.at)

*submitted 1 January 2003; revised 1 January 2003; accepted 1 January 2003*

During the last decade, Constraint Handling Rules (CHR) have become a major specification and implementation language for logical, constraint-based algorithms and intelligent applications (as witnessed for example by several hundred publications available online that mention CHR). Algorithms are often specified using inference rules, rewrite rules, sequents, proof rules, or logical axioms that can be almost directly written in CHR. In recent years, CHR have been successfully used

- as general purpose concurrent constraint language with ask and tell
- as fairly efficient production rule system
- as special kind of theorem prover with constraints
- as system combining forward and backward chaining
- for bottom-up evaluation with integrity constraints
- for top-down evaluation with tabulation
- for combining deduction, abduction, and constraints
- as high-level language for manipulating attributed variables
- for parsing with executable grammars
- for multi-set rewriting and transformations

Applications of CHR range from type systems and time tabling to ray tracing and cancer diagnosis.

CHR is a concurrent committed-choice constraint logic programming language consisting of guarded rules that transform multi-sets of constraints (atomic formulae) into simpler ones until they are solved. CHR make it easy to define constraint reasoning: simplification and propagation as well as incremental solving of constraints. With multi-set formula rewriting and propagation rules, CHR provides a unique combination of features which are essential for non-trivial constraint handling and high-level declarative algorithm design.

Based on first order predicate logic, the clean semantics of CHR facilitates non-trivial program analysis and transformation. About a dozen implementations of CHR exist in Prolog, Haskell, and Java, some with highly optimizing compilers.

CHR is also available as WebCHR for online experimentation with more than 40 constraint solvers and one's own programs. The CHR web-pages can be found at: `http://www.informatik.uni-ulm.de/pm/mitarbeiter/fruehwirth/chr.html`

The following seven papers were accepted from the submissions after at least two rounds of reviewing by at least three reviewers. Revised versions of previously published papers that have not appeared in archival journals were eligible for submission, including those from four workshops on Rule-Based Constraint Reasoning and Programming. The accepted papers range from application to generation, extension and implementation of CHR.

- *FLUX: A Logic Programming Method for Reasoning Agents.*
  Thielscher presents a programming method called FLUX for the design of agents that reason logically about their actions and about sensor information in the presence of incomplete knowledge.
- *Intelligent Search Strategies Based on Adaptive Constraint Handling Rules.*
  Wolf's DJCHR is the first implementation of adaptive rule-based constraint handling which allows the successful application of intelligent search strategies such as random 3-SAT problems.
- *CHR Grammars.*
  Henning develops a powerful engine for constraint-based language processing and supports this in terms of the highly flexible notation of GHR Grammars.
- *A CHR-based Implementation of Known Arc-Consistency.*
  Alberti, Gavanelli, Lamma, Mello and Milano describe the implementation of a solver for a two-sorted CLP language which supports the declarative definition of variable domains as sets and the interleaving of constraint propagation and acquisition of domain values.
- *Automatic Generation of CHR Constraint Solvers.*
  Several methods have been proposed to generate automatically constraint solving rules from constraint specifications. Abdennadher and Rigotti present a framework that extends these previous works by using tabled resolution to generate more expressive rules.
- *Schedulers and Redundancy for a Class of Constraint Propagation Rules.*
  Brand and Apt study the problem of efficient scheduling of rules that naturally arise in the context of rule-based constraint programming. They also analyze how redundant rules can be identified.
- *Optimizing Compilation of Constraint Handling Rules in HAL.*
  Holzbaur, Garcia de la Banda, Stuckey and Duck describe how to improve the compilation of CHR by analyses to determine functional dependencies, set semantics and never stored constraints. The information so derived is used in join ordering, index creation and continuation optimization.

We are thankful to Krzysztof Apt for the idea of this special issue on CHR, to our referees and to Maurice Bruynooghe as the editor in chief of Theory and Practice of Logic Programming.

Slim Abdennadher, Thom Frühwirth, Christian Holzbaur, Guest Editors