Rule-based and Constraint Programming

Summer 2013 Project

contact: amira.zaki@uni-ulm.de

Inverse computation of a program is the calculation of the possible inputs given an output. If part of the input is also given, then this is known as partial inversion. Inverse computation is performed by inverse interpreters whilst program inversion is performed by inversion compilers.

Sometimes it is easier to solve a problem in one direction rather the other, like it is easier to disarrange Rubik's cube than to arrange it. Applications to concrete programs include:

- decoding from encoding (Huffman, cryptography)
- inverting (arithmetic) functions
- computing samples for inconsistency/failure in a constraint solver starting backwards from failure
- computing tautologies starting backwards from "true"

We want to compute backwards from a result of a CHR query to one or all possible predecessor states. This process is indeterministic, so the angelic semantics of CHR [2] comes handy. Such an execution is useful for running programs (and their associated functions) backwards, for simulating CHR executions, and for reasoning about them, i.e. to achieve reachability analysis and safety analysis as in model checking for software verification. Reverse execution might also be useful for computing with bidirectional rules in both directions, getting CHR more closer to its most abstract first-order logic semantics.

Preliminary work has been done to invert some common CHR programs. In this project, we wish to compile the various examples and generalize the inversion procedure, then implement an inversion compiler that produces the inverse of an input CHR program. The compiler reads the input CHR program, translates it to a standard notation, performs the inverse then outputs the inverse CHR program.

References

[1] Thom Frühwirth. **Constraint Handling Rules**, Cambridge University Press, August 2009.

[2] Thierry Martinez. **Angelic CHR**. In Jon Sneyers, editor, Proceedings of the 8th Workshop on Constraint Handling Rules (CHR'11), pages 19–31, September 2011. Technical report, GUC.

[3] Richard E. Korf. **Inversion of applicative programs**. Proceedings of the 7th international joint conference on Artificial intelligence (IJCAI'81), pages 10071009, 1981.

[4] Naoki Nishida, Masahiko Sakai and Terutoshi Kato. **Convergent term rewriting systems for inverse computation of injective functions**. Proceedings of the 9th International Workshop on Termination (WST'07), pages 77–81, 2007.

[5] Nachum Dershowitz and Subrata Mitra. **Jeopardy: Inverting Mildly Deep Definitions**. Proceedings of the 10th International Conference on Rewriting Techniques and Applications (RTA'99), pages 16–29, 1999.