CoScart: Confluence Prover in Scala

Karl Gmeiner[1]

UAS Technikum Wien, Vienna, Austria
gmeiner@technikum-wien.at

1 Overview

CoScart is a tool to prove confluence of first-order term rewrite systems and deterministic conditional term rewrite systems automatically. It originates from the project *KaRT*, a collection of Java classes for term rewriting focussing on comparin transformations of conditional term rewrite systems and program transformations for functional programming languages. A first version of KaRT was used to conduct the experiments in [2]. To speed up and simplify development, in particular with focus on implementing CoScart, the whole project was ported to Scala, a functional, object-oriented programming language that compiles to Java Bytecode.

CoScart also comes with an automated termination prover and thus is a stand-alone-tool that does not rely on any other software.

2 Technical Details

The rewrite engine of Scart stores DAGs of terms that are collected in a linked list. This way rewriting is very efficient.

In order to use the Knuth-Bendix method, Scart contains an automatic termination prover (*TeScart*) for first-order TRSs that uses the dependency pairs method in combination with argument filterings with the *some more*-heuristics of [1].

A web interface is planned. New features compared to last year use the latest result of [4] that shows that confluence can be proved via transformations of CTRSs without considering soundness.

Since CoScart is currently a one-man project, there are no sophisticated user interfaces yet, but a web interface is planned.

CoScart proves confluence of (deterministic conditional) TRSs using the following methods: Transformation of [3] from DCTRSs into TRSs, modularity of confluence, Knuth-Bendix, and development-closed critical pairs of left-linear TRSs.

Scart is available at https://github.com/searles/RewriteTool/.

References

[1] N. Hirokawa and A. Middeldorp. Automating the Dependency Pair Method. In *Proc. CADE 2003*, LNAI vol. 2741, pp. 32–46, Springer-Verlag, 2003.

[2] K. Gmeiner and B. Gramlich. Transformations of Conditional Rewrite Systems Revisited. In *Proc. WADT 2008*, LNCS vol. 5486, pp. 166-186, Springer-Verlag, 2009.

[3] K. Gmeiner and N. Nishida. Notes on Structure-Preserving Transformations of Conditional Term Rewrite Systems. In *Proc. WPTE 2014*, OASIcs vol. 40, pp. 3–14, 2014.

[4] K. Gmeiner. Confluence of Conditional Term Rewrite Systems via Transformations. In *Proc. WPTE 2016*, 2016.