ACP: System Description for CoCo 2018

Takahito Aoto[1] and Yoshihito Toyama[2]

[1] Faculty of Engineering, Niigata University
aoto@ie.niigata-u.ac.jp
[2] RIEC, Tohoku University
toyama@riec.tohoku.ac.jp

A primary functionality of ACP is proving confluence of term rewriting systems (TRSs). ACP integrates multiple direct criteria for guaranteeing confluence of TRSs. It also incorporates divide–and–conquer criteria by which confluence or non-confluence of TRSs can be inferred from those of their components. Several methods for disproving confluence are also employed. For some criteria, it supports generation of proofs in CPF format that can be certified by certifiers. The internal structure of the prover is kept simple and is mostly inherited from the version 0.11a, which has been described in [2]. No new (non-)confluence criterion for TRSs has been incorporated from the one submitted for CoCo 2017.

This year we have added a new functionality to ACP, namely that of proving unique normal forms w.r.t. conversion (UNC) of TRSs. It incorporates divide–and–conquer criteria for UNC and multiple direct criteria for guaranteeing UNC of TRSs. The list of implemented criteria and methods is reported in [3]. In particular, this includes a *UNC completion method* which is inspired from conditional linearlization technique [4], and a UNC criterion of non-duplicating weight-decreasing joinability [5]. A preliminary implementation for proving confluence of (oriented, type 3) conditional term rewriting systems, is also added.

ACP is written in Standard ML of New Jersey (SML/NJ) and the source code is also available from [1]. It uses a SAT prover such as MiniSAT and an SMT prover YICES as external provers. It internally contains an automated (relative) termination prover for TRSs but external (relative) termination provers can be substituted optionally. Users can specify criteria to be used so that each criterion or any combination of them can be tested. Several levels of verbosity are available for the output so that users can investigate details of the employed approximations for each criterion or can get only the final result of prover's attempt.

References

[1] ACP (Automated Confluence Prover). http://www.ie.riec.tohoku.ac.jp/tools/acp/.

[2] T. Aoto, J. Yoshida, and Y. Toyama. Proving confluence of term rewriting system automatically. In *Proc. of 20th RTA*, volume 5595 of *LNCS*, pages 93–102. Springer-Verlag, 2009.

[3] T. Aoto and Y. Toyama. Automated proofs of unique normal forms w.r.t. conversion for term rewriting systems. Submitted, 2018.

[4] R.C. de Vrijer. Conditional linearization. *Indagationes Mathematicae*, Vol. 10, No. 1, pp. 145–159, 1999.

[5] Y. Toyama and M. Oyamaguchi. Conditional linearization of non-duplicating term rewriting systems. *IEICE Transactions on Information and Systems*, Vol. E84-D, No. 4, pp. 439–447, 2001.